Diagnosing Island Supplemental Material

Jose Guadalupe Hernandez

2023-02-01

Contents

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       x86_64-pc-linux-gnu
## arch           x86_64
## 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.

```r
# 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
```

```r
library(PupillometryR)
```

```
## Loading required package: rlang
```

```r
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.
```

```r
# default variables

MODEL = c('EA','IS','NMIS')
EXPERIMENTS = c('BASE-EXPERIMENTS/','MI50/','MI5000/')
SCHEME = c('TRUNCATION','TOURNAMENT','LEXICASE')
DIAGNOSTIC = c('EXPLOITATION_RATE', 'ORDERED_EXPLOITATION', 'CONTRADICTORY_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 = "", collap
    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 = "", collap
    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 = "", coll
    mi5000_over_time = rbind(mi5000_over_time, read.csv(mi5000_dir, header = TRUE, stringsAsFacto

  }
}
```

```
## [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 = "", c
    base_best = rbind(base_best, read.csv(base_dir, header = TRUE, stringsAsFactors = F

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

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

```
## [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 = N
    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 = N
    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 =
    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

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

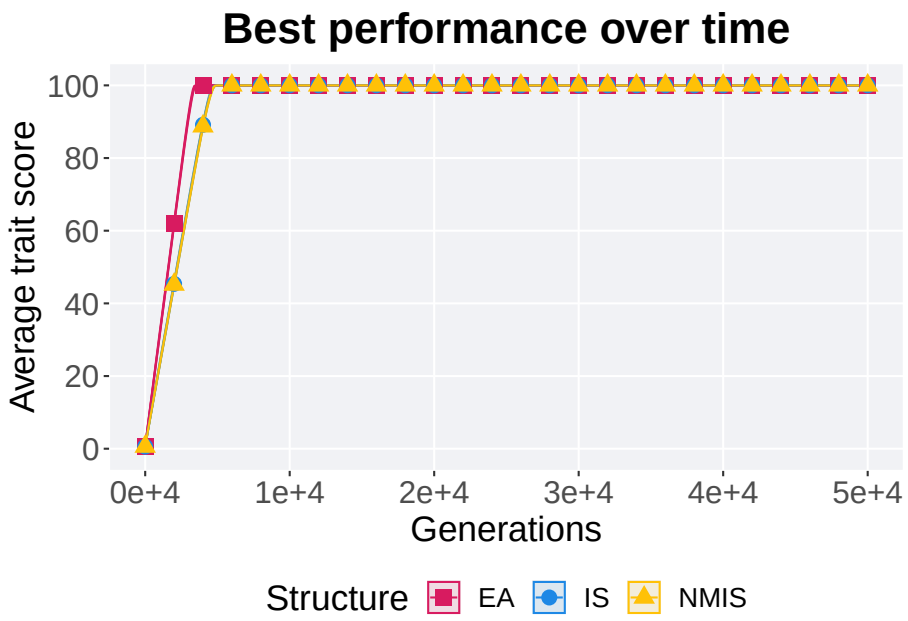
Here we analyze how the different population structures affect truncation selection (size 8) on the exploitation rate diagnostic.

2.2.1 Performance over time

```
lines = filter(base_over_time, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TRUNCA
  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",
    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
```

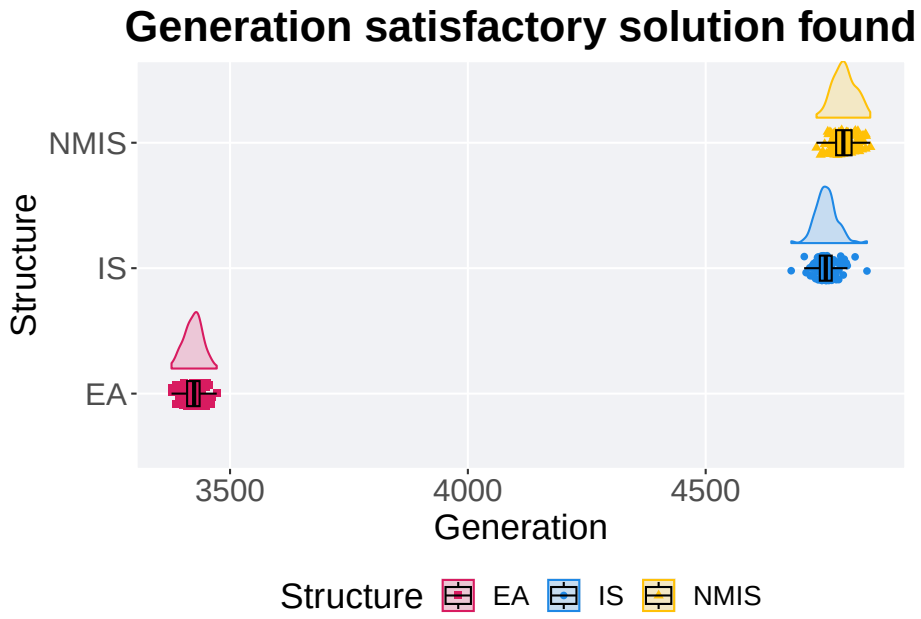
```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
```



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 = Str
  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` == 'TRU
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
```

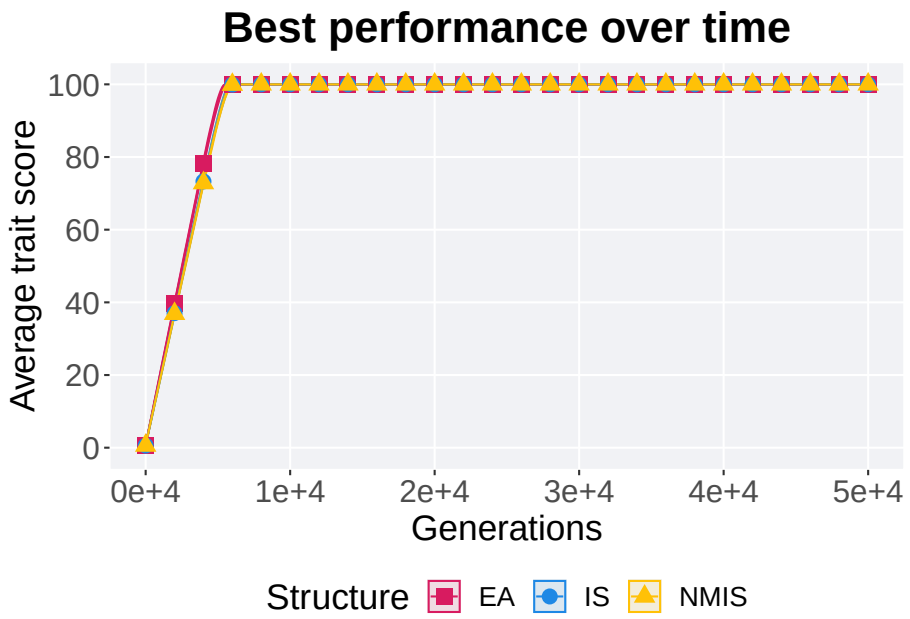
2.3 Tournament selection

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` == 'TOURNA
  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
```

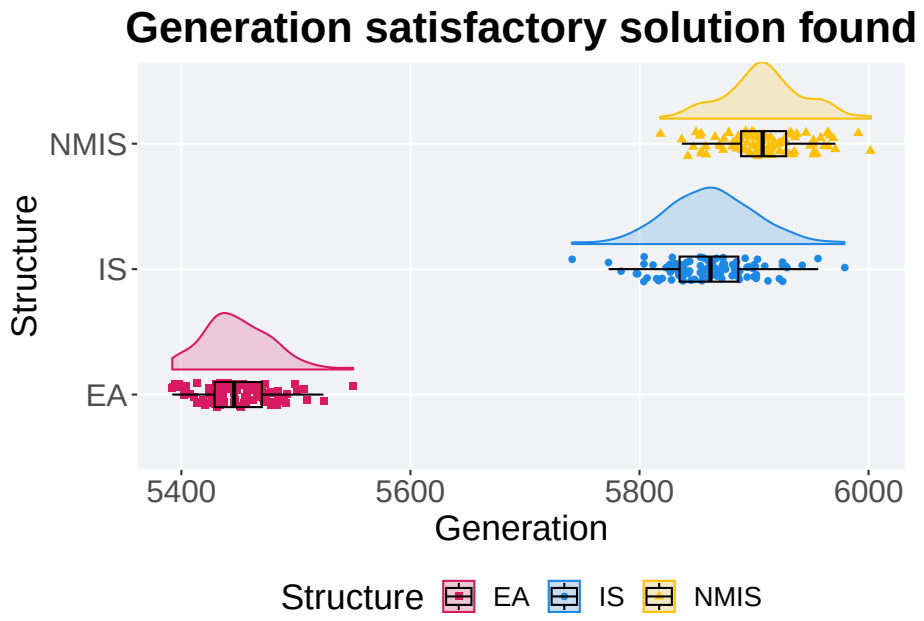


2.3.2 Generation satisfactory solution found

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, s
  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="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 = "bonferr
                     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
```

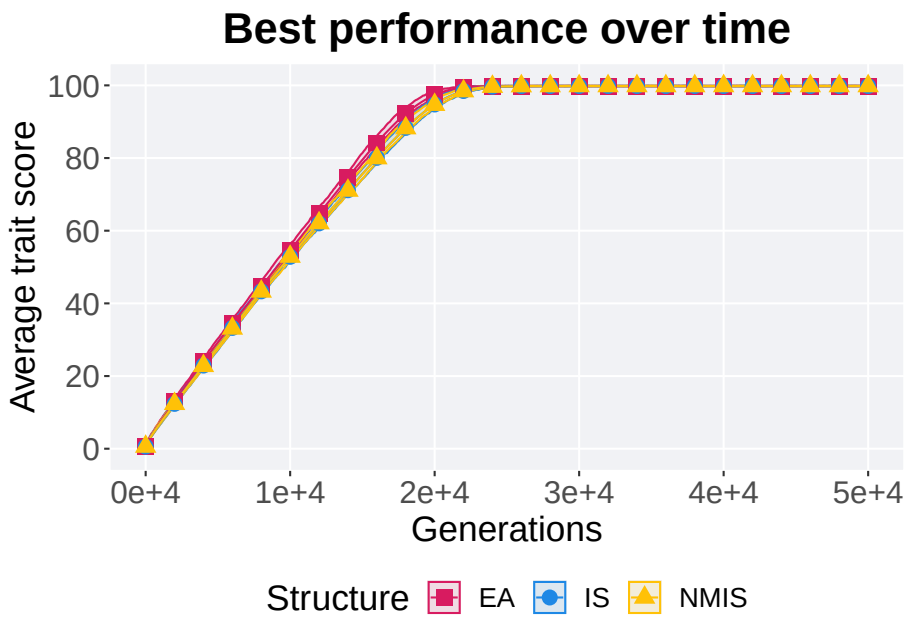
2.4 Lexicase selection

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(
```

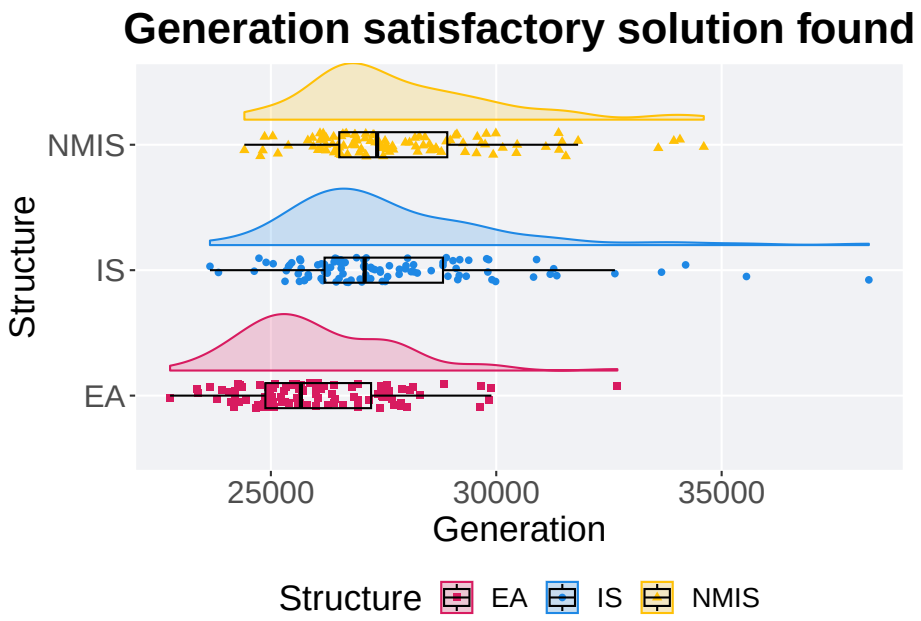
```
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  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, s
  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="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 = "bonferr
                     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

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)
```

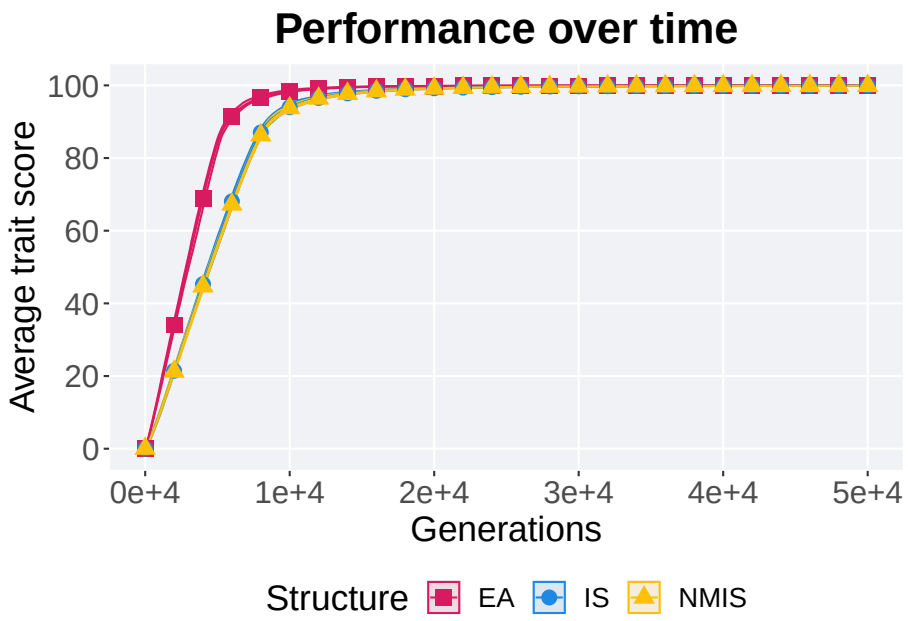
3.2 Truncation selection

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,
```

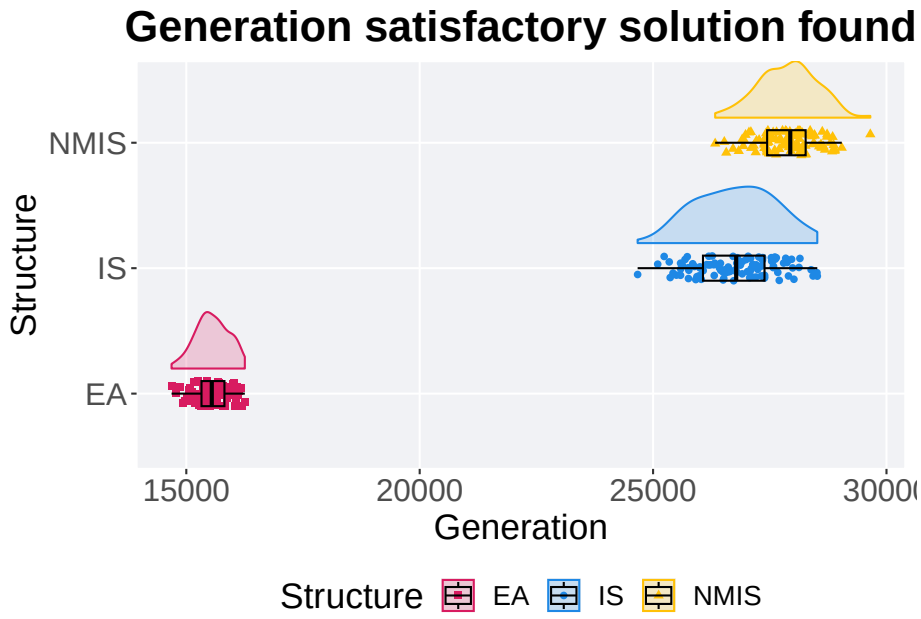
```
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0,
  scale_y_continuous(
    name="Average trait score",
    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 = Str
  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` == '
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
```

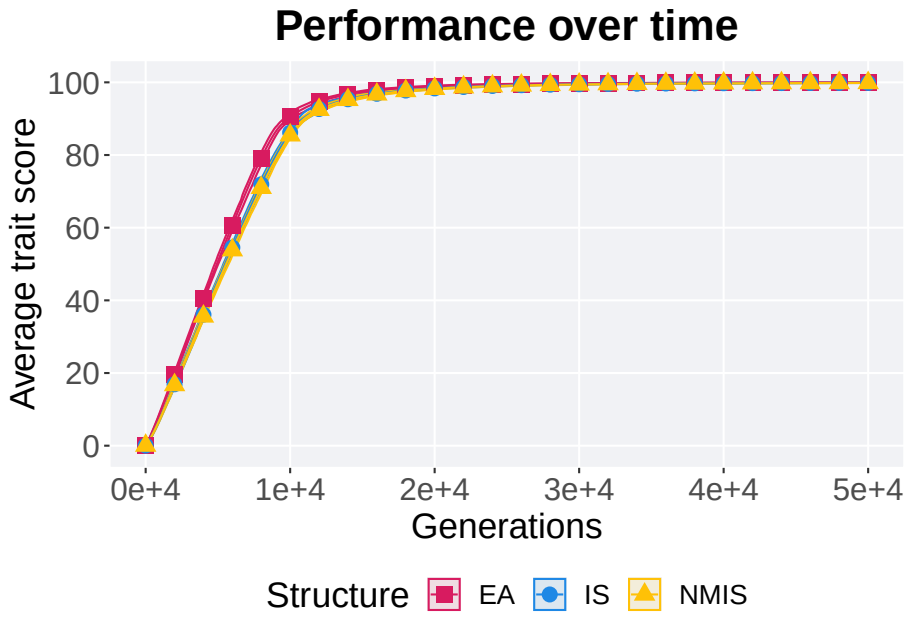
3.3 Tournament selection

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` == 'TOU
  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
```

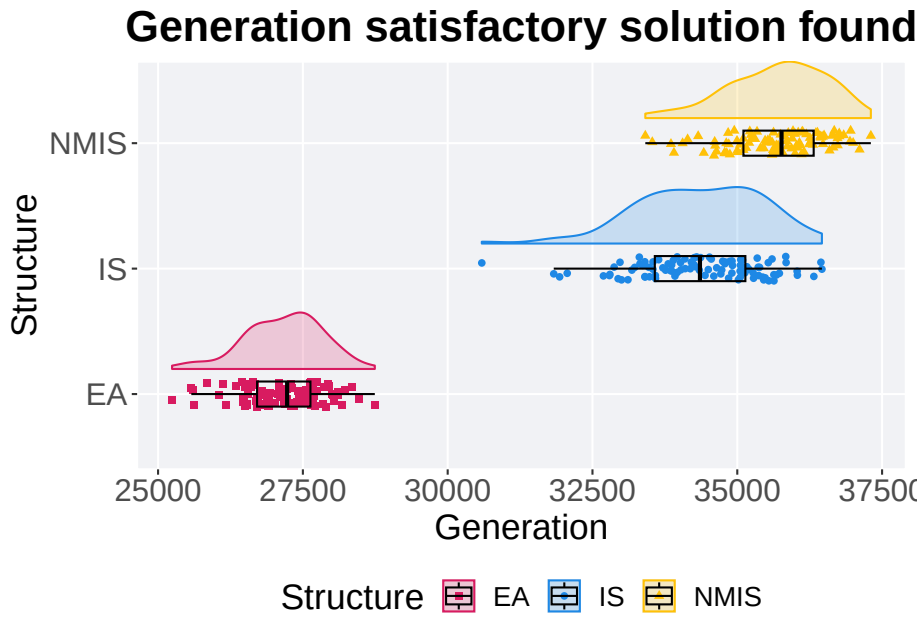


3.3.2 Generation satisfactory solution found

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

```
filter(base_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TOURNAM
  ggplot(., aes(x = Structure, y = Generations , color = Structure, fill = Structure, s
  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="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 = "bonferr
                     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
```

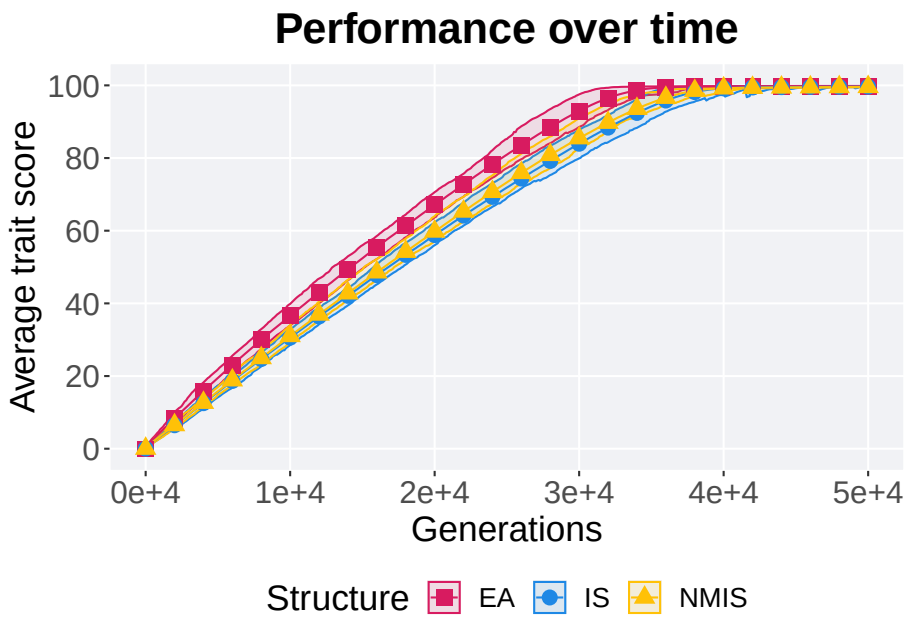
3.4 Lexicase selection

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\nSchem
  group_by(Structure, Generations) %>%
  dplyr::summarise(
```

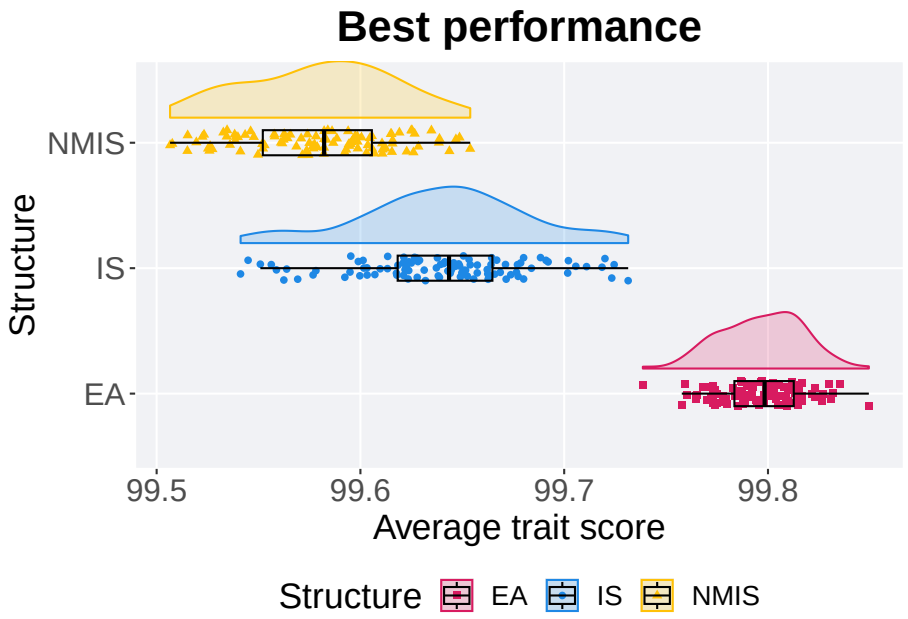
```
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  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 = 
  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(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 0.0291
## 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(VAL ~ Structure, data = performance)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  VAL 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.

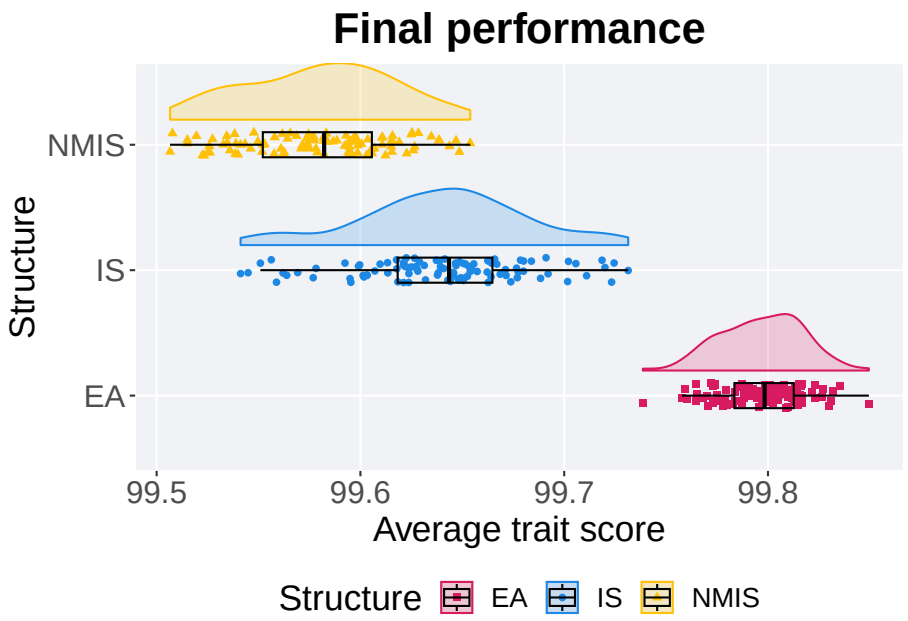
```
pairwise.wilcox.test(x = performance$VAL, g = performance$Structure, p.adjust.method =
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VAL and performance$Structure
##
##      EA     IS
## IS   <2e-16 -
## NMIS <2e-16 <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` == '
  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 = 
  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
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

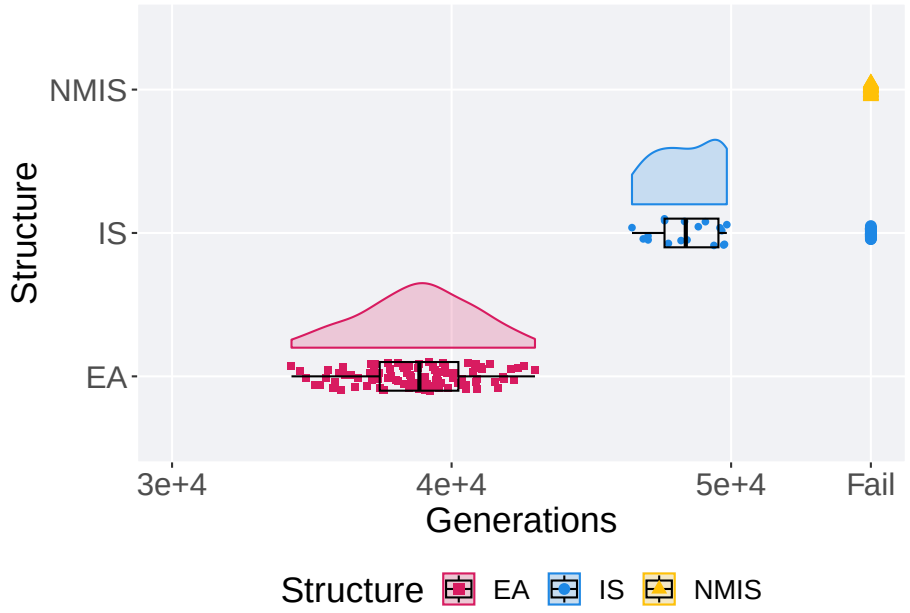
```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$pop_fit_max and performance$Structure
##
##      EA     IS
## IS   <2e-16 -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

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`
lex_fail$Generations = 55000
lex_fail$Structure <- factor(lex_fail$Structure, levels = MODEL)

filter(base_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEXICAS
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, sh
    geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha =
  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, fi
  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 = "bonferr
                     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

Contradictory objectives results

Here we present the results for the **satisfactory trait corverage** 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.

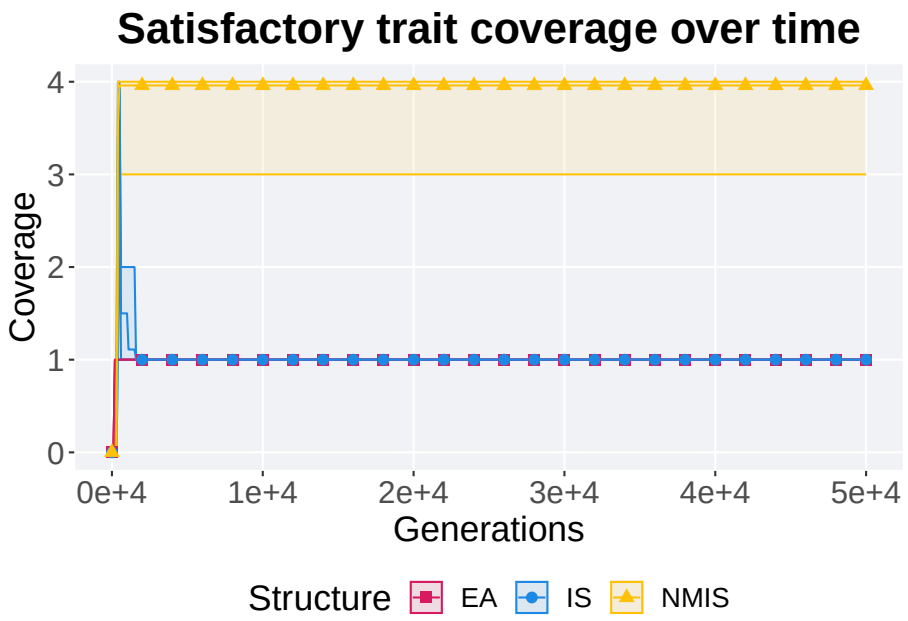
4.2.1.1 Coverage over time

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.
```

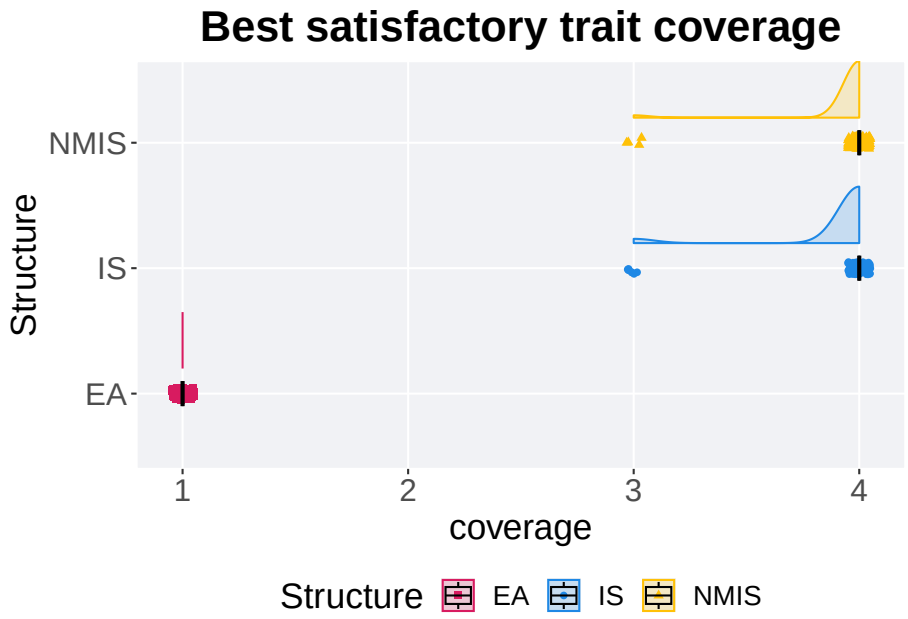
```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
  scale_y_continuous(
    name="Coverage"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

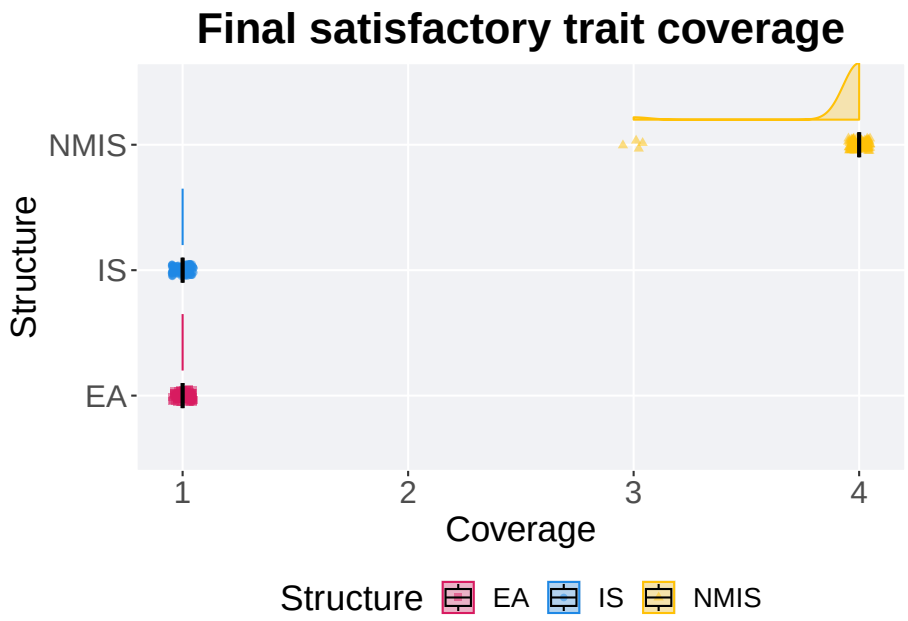
```
pairwise.wilcox.test(x = coverage$VAL, g = coverage$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
##  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 == 'CONTRADICTORY_OBJECTIVES' & `Selectio
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 = "bonferr
                     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.

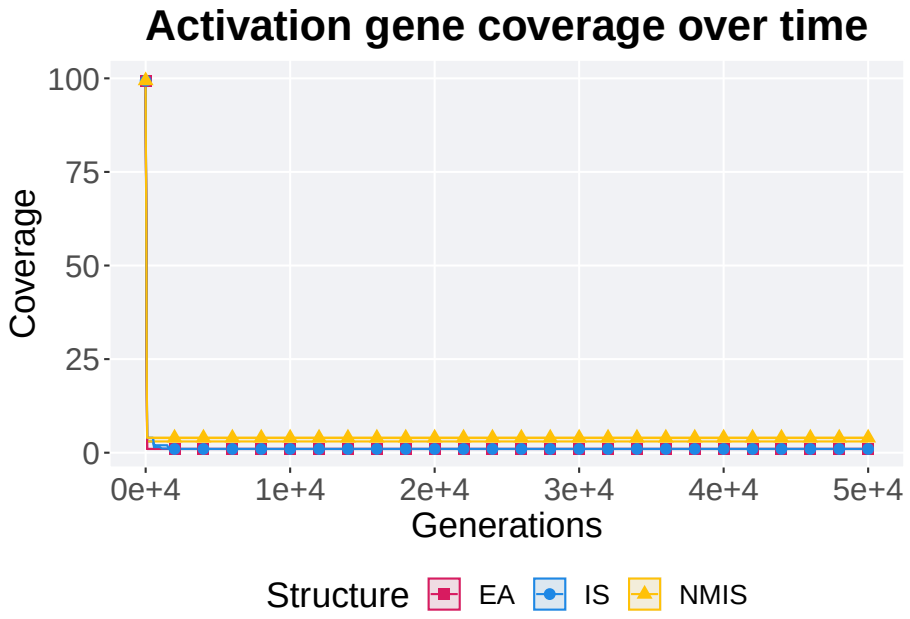
4.2.2.1 Coverage over time

Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == '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.
```

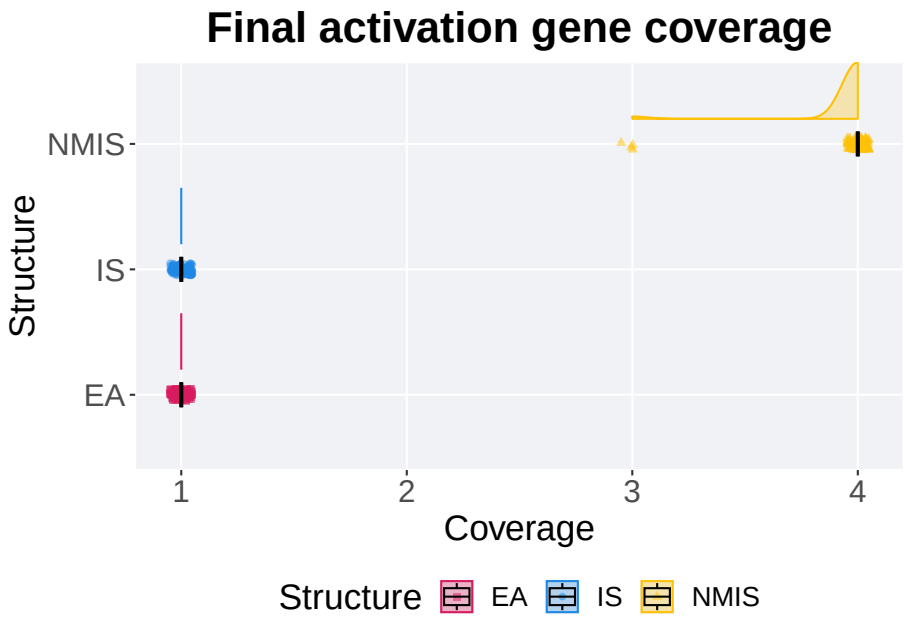
```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
  scale_y_continuous(
    name="Coverage"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



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 == 'CONTRADICTORY_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 == 'CONTRADICTORY_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.

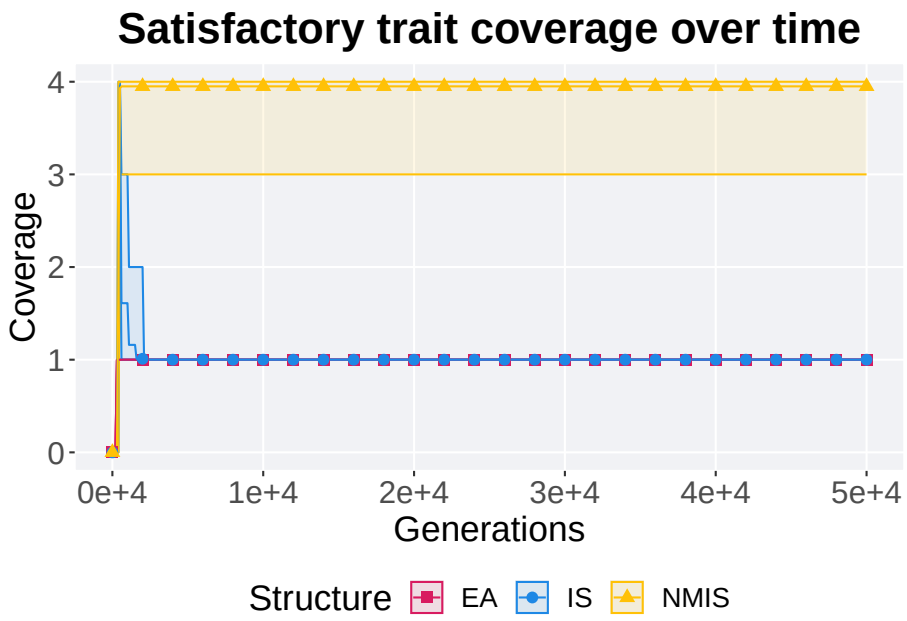
4.3.1.1 Coverage over time

Satisfactory trait coverage over time.

```
lines = filter(base_over_time, Diagnostic == 'CONTRADICTORY_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.
```

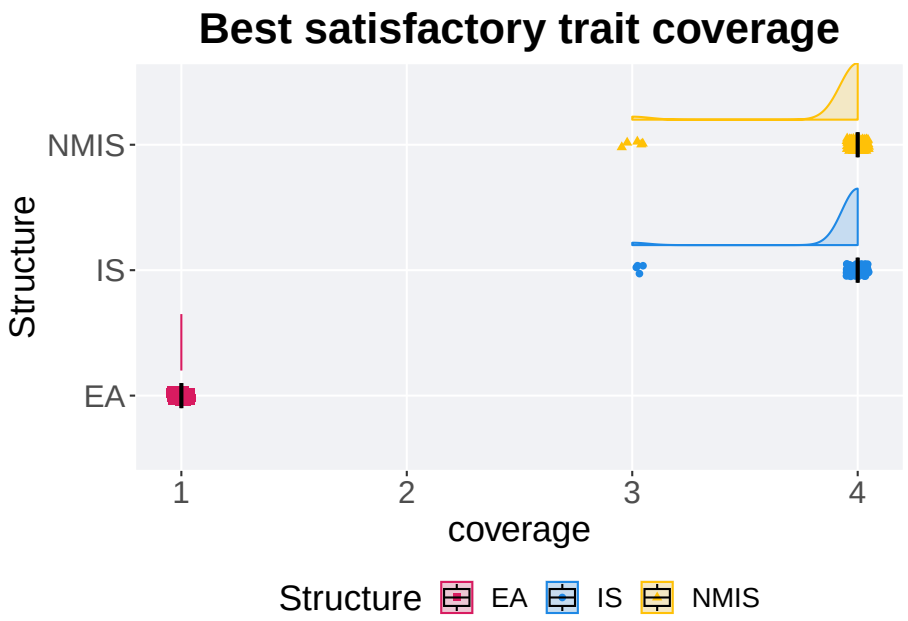
```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    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 == 'CONTRADICTORY_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 =
  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()
```



4.3.1.2.1 Stats

Summary statistics for the best satisfactory trait coverage.

```
### best
coverage = filter(base_best, Diagnostic == 'CONTRADICTORY_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.

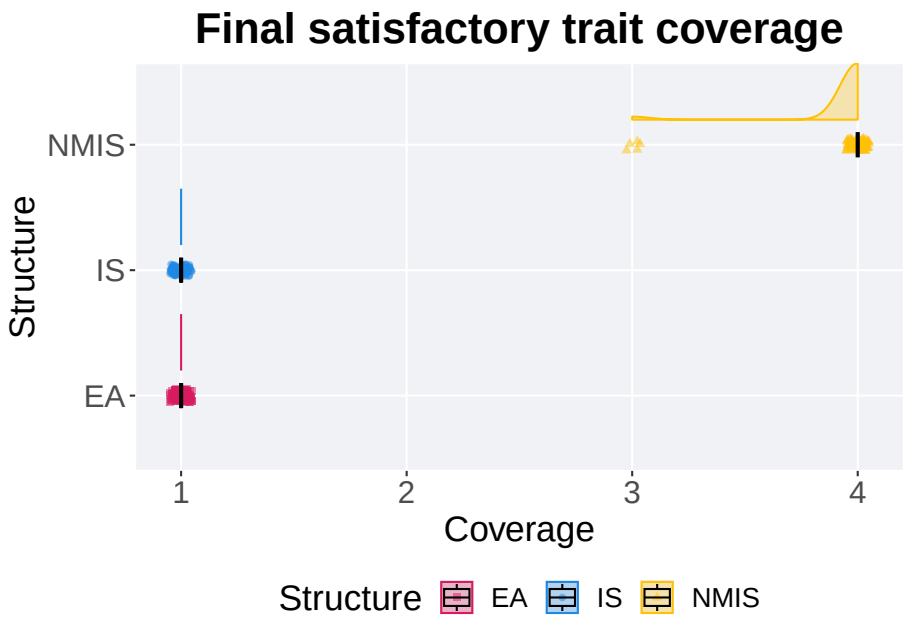
```
pairwise.wilcox.test(x = coverage$VAL, g = coverage$Structure, p.adjust.method = "bonfe
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
##  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` =
  ggplot(., aes(x = Structure, y = pop_sat_cov, color = Structure, fill = Structure, sh
  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 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 == 'CONTRADICTORY_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
                     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.

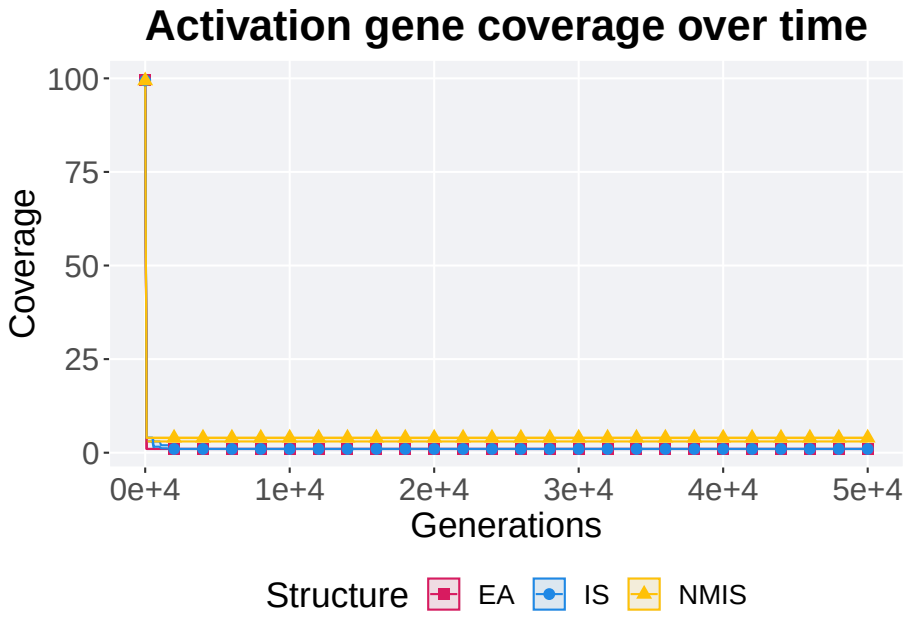
4.3.2.1 Coverage over time

Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\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.
```

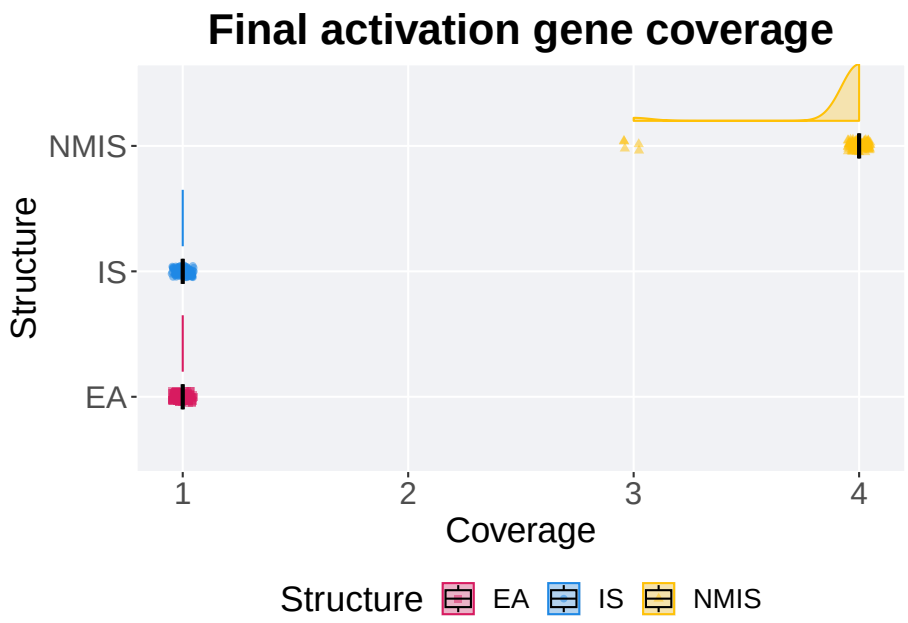
```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    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 == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` =
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, sh
  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 == 'CONTRADICTORY_OBJECTIVES' & `Selectio
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 = "bonferr
                     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.

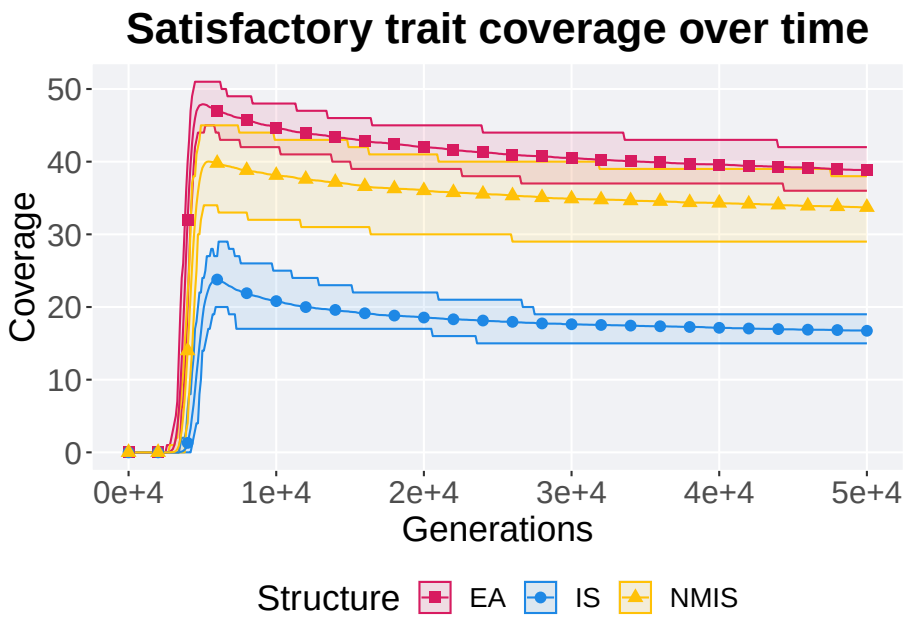
4.4.1.1 Coverage over time

Satisfactory trait coverage over time.

```
lines = filter(base_over_time, Diagnostic == '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.
```

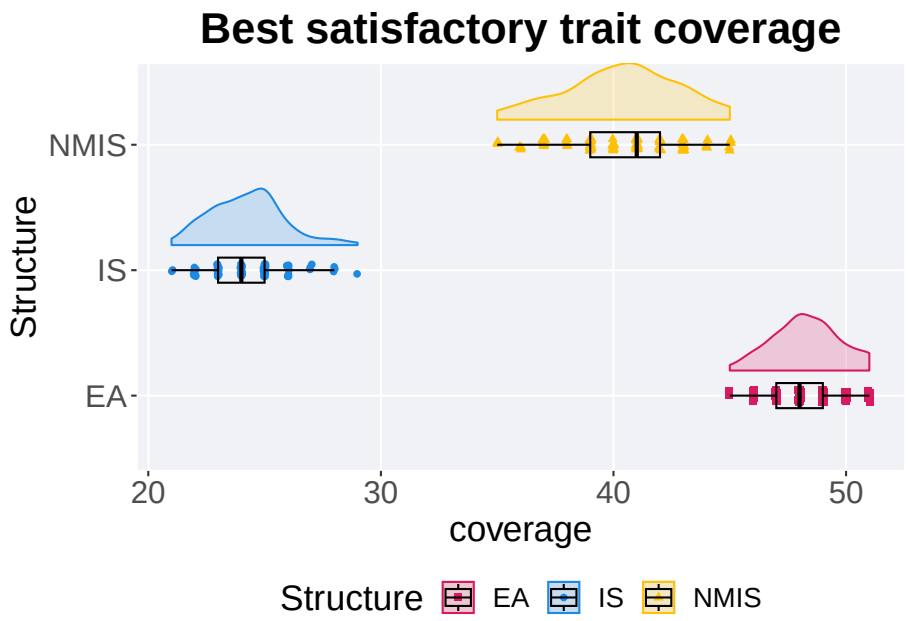
```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
  scale_y_continuous(
    name="Coverage"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



4.4.1.2 Best coverage throughout

Best satisfactory trait coverage throughout 50,000 generations.

```
### best satisfactory trait coverage throughout
filter(base_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()
```



4.4.1.2.1 Stats

Summary statistics for the best satisfactory trait coverage.

```
### best
coverage = filter(base_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  48.3    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')
```

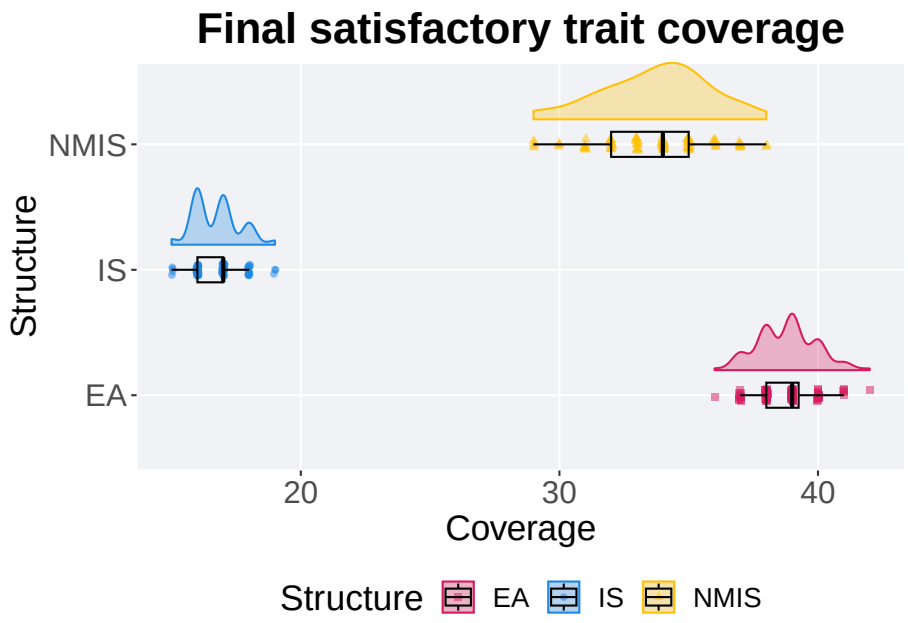
```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$VAL and coverage$Structure
##
##      EA     NMIS
## NMIS <2e-16 -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

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` == 'LEXICAS
  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()
```



4.4.1.3.1 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' & `Selectio
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 = "bonferr
                     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.

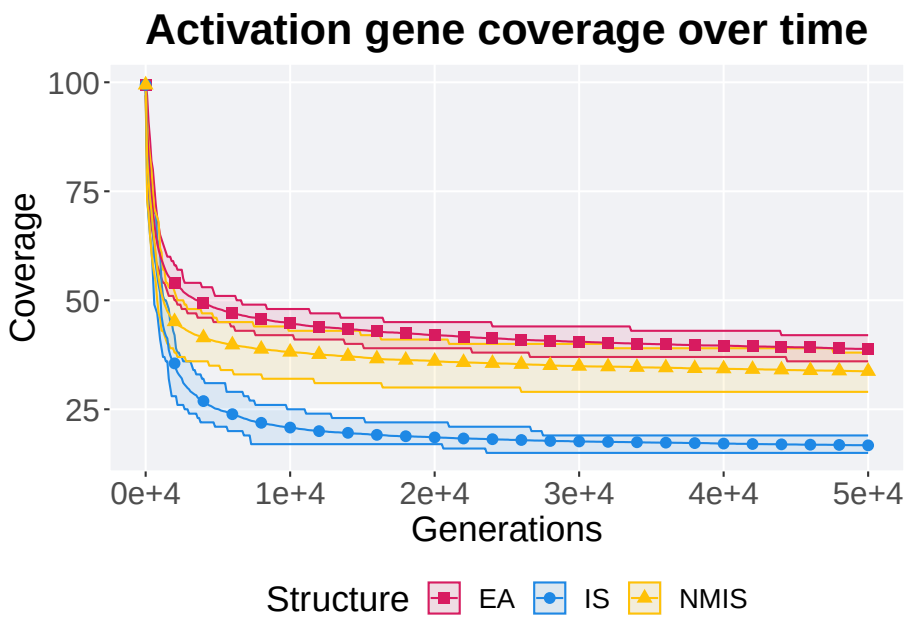
4.4.2.1 Coverage over time

Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == '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
```

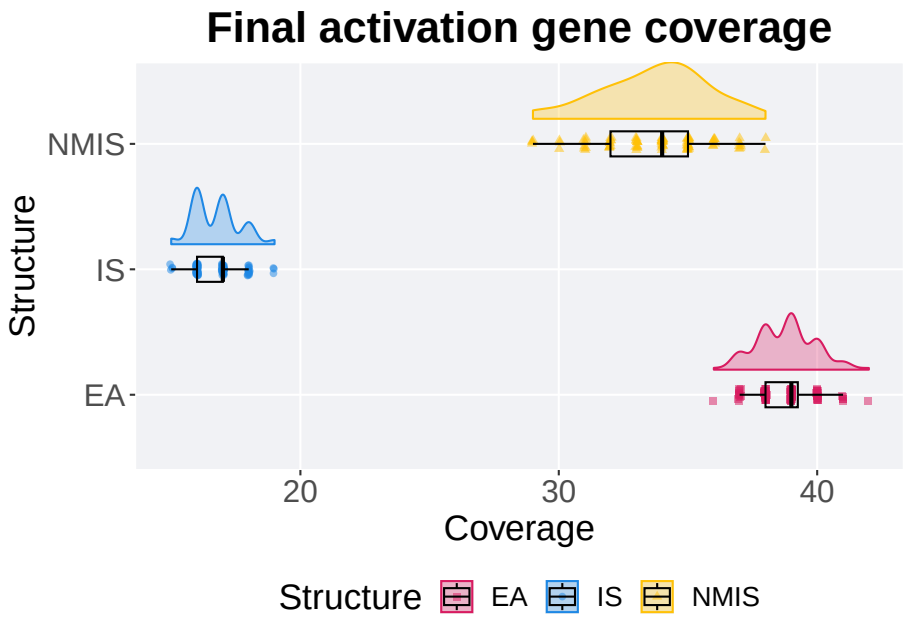
```
## `.groups` argument.
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
  scale_y_continuous(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



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 == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'LEXICAS
  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()
```



4.4.2.2.1 Stats

Summary statistics for activation gene coverage.

```
coverage = filter(base_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme`
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

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)
```

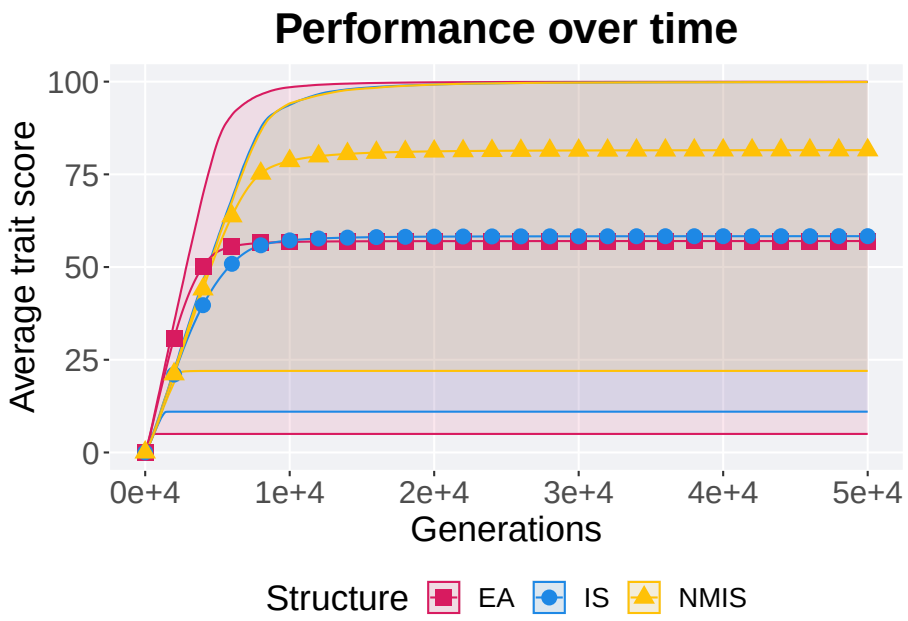
5.2 Truncation selection

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

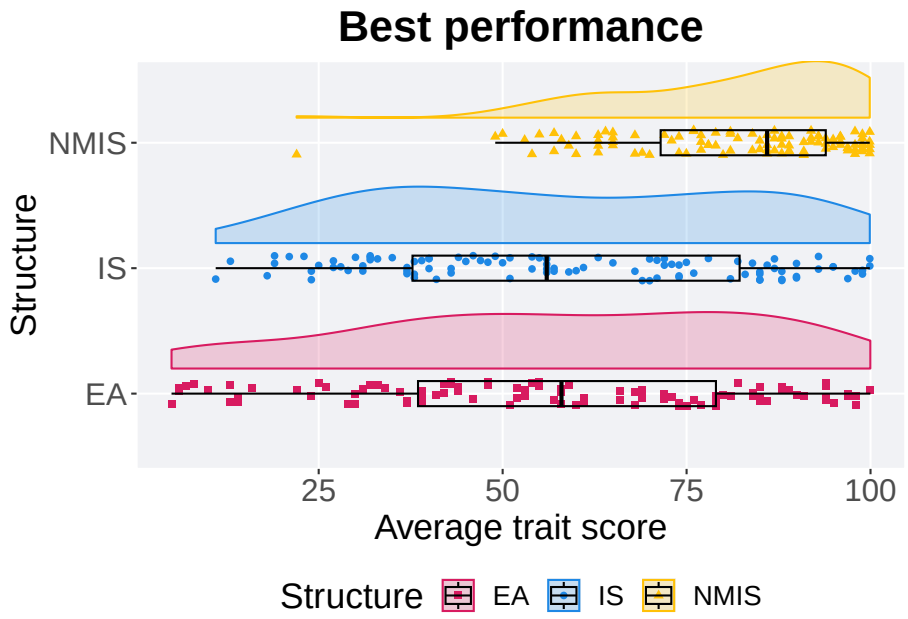
```
lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nSche
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = 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(VAL ~ 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.

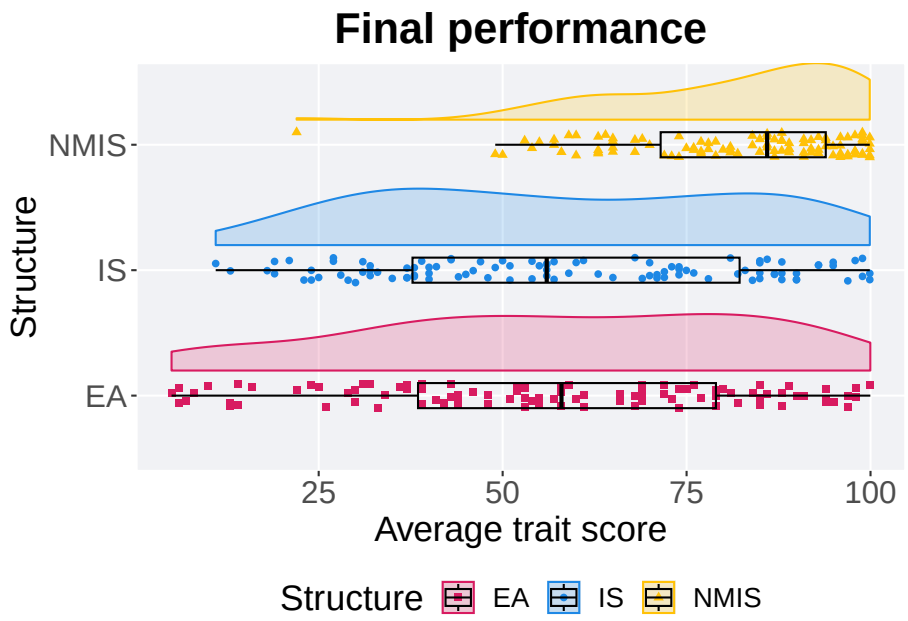
```
pairwise.wilcox.test(x = performance$VAL, g = performance$Structure, p.adjust.method = "bonferron
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
##  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 = Struct
  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' & `Selectio
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.

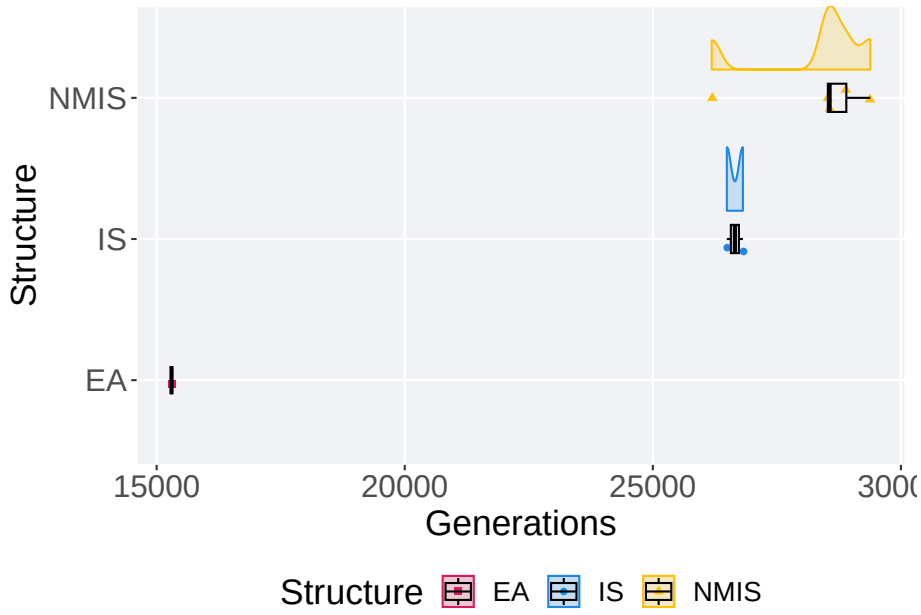
```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.method = "b
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
##  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'& Ger
  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.

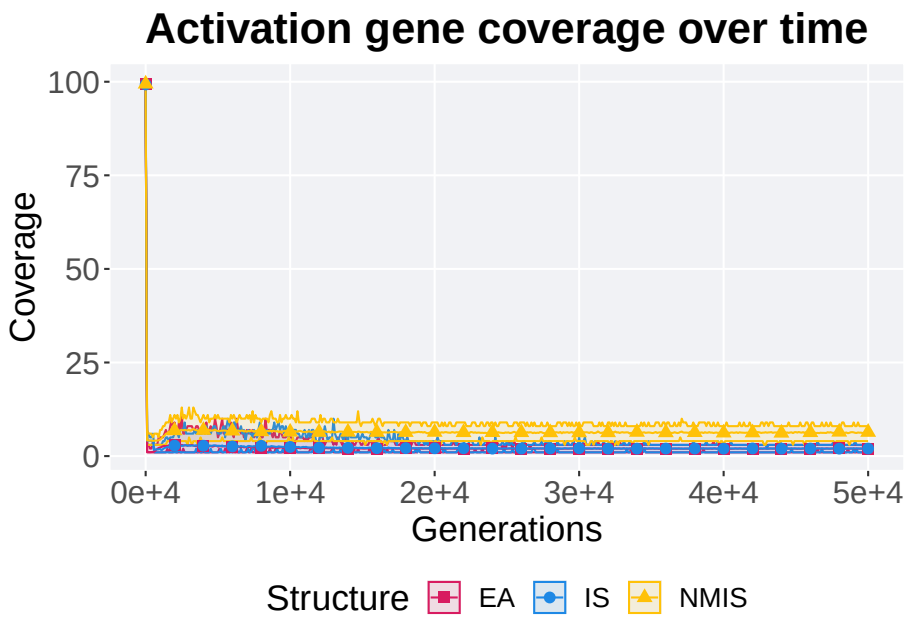
5.2.3.1 Coverage over time

Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TR
  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.
```

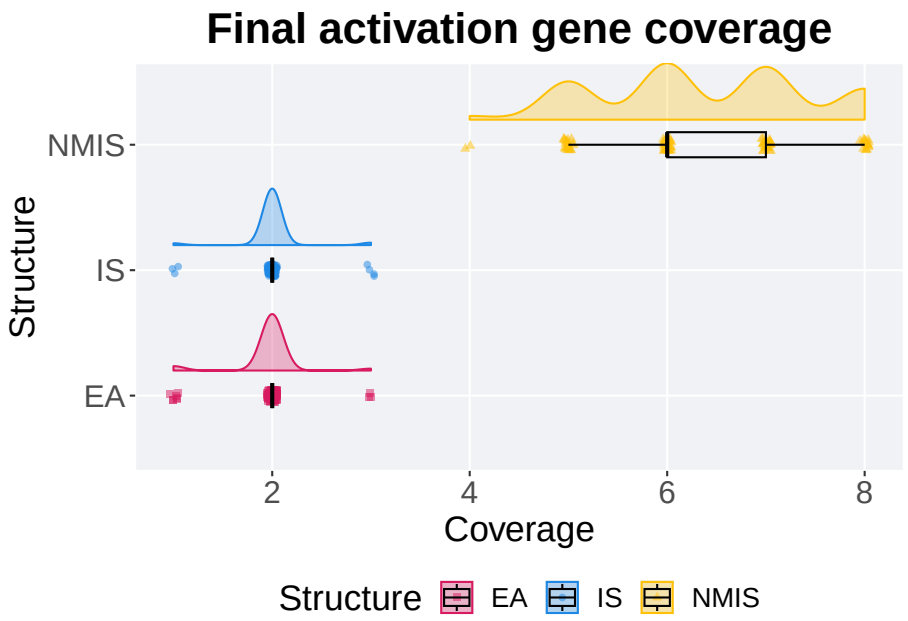
```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    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, sl
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = (
  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()
```



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

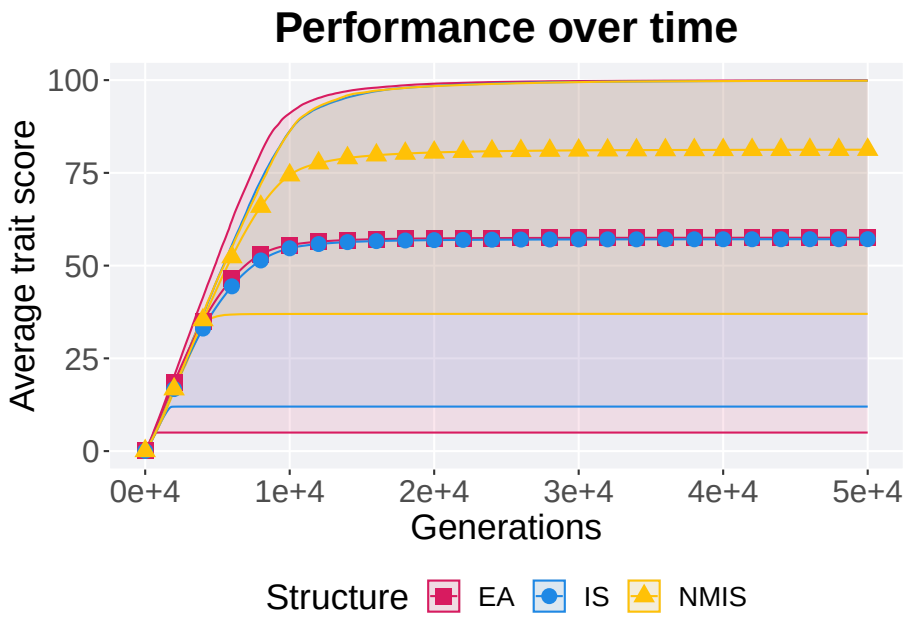
Here we analyze how the different population structures affect tournament selection (size 8) on the contradictory objectives diagnostic.

5.3.1 Performance

5.3.1.1 Performance over time

```
lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\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 = 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
```

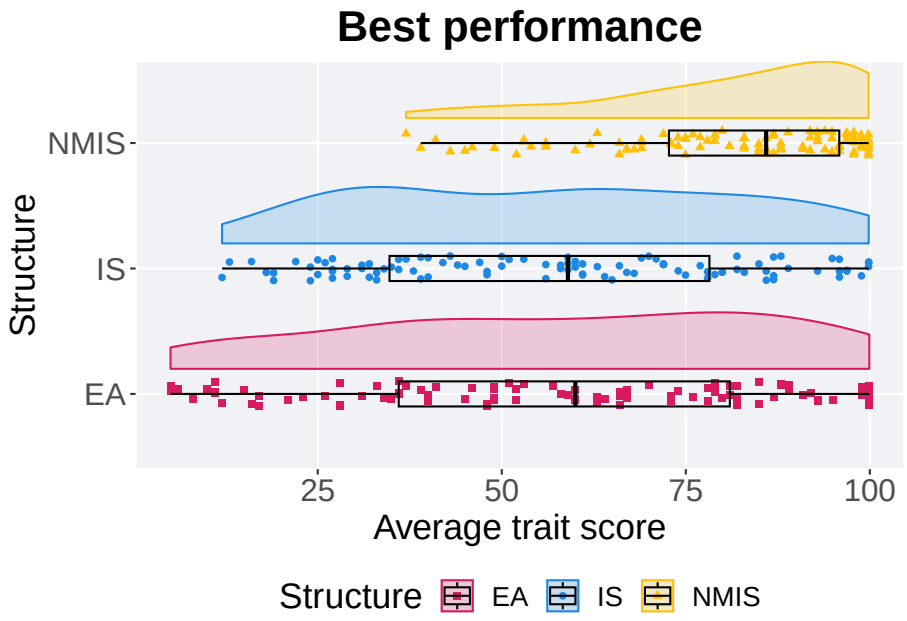


5.3.1.2 Best performance

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\nScl
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     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(VAL ~ Structure, data = performance)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  VAL 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$VAL, g = performance$Structure, p.adjust.method = "bonferron
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

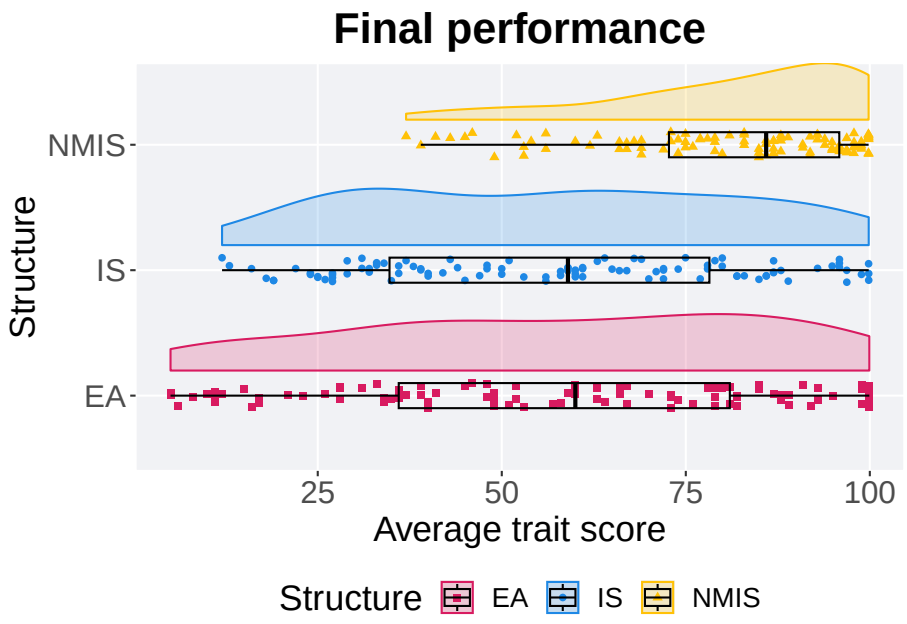
```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VAL and performance$Structure
##
##      EA      IS
## IS   1       -
## NMIS 5.9e-09 5.3e-11
##
## P value adjustment method: bonferroni
```

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 = Struct
  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' & `Selectio
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')
```

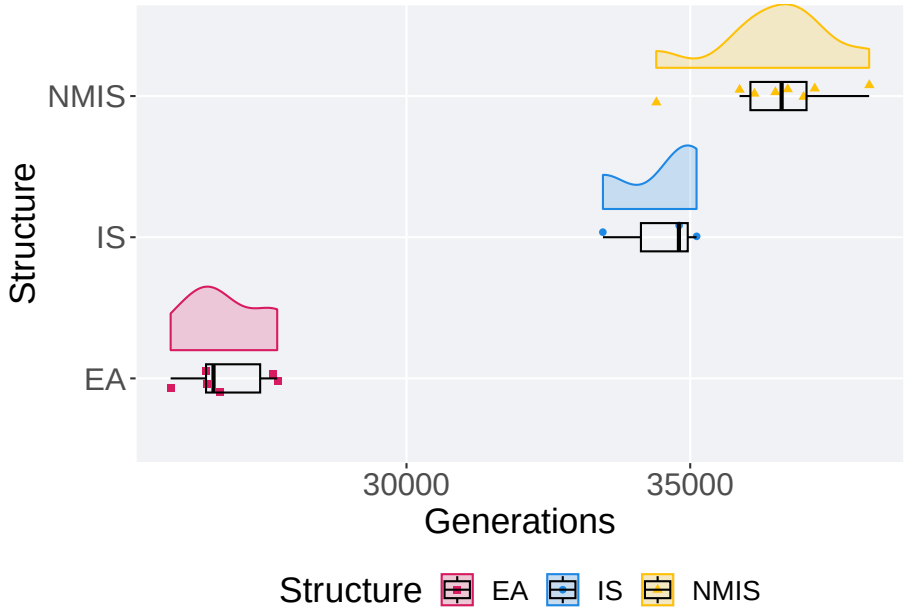
```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$pop_fit_max and performance$Structure
##
##      EA      IS
## IS   1       -
## NMIS 5.9e-09 5.3e-11
##
## P value adjustment method: bonferroni
```

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 = 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()
```



5.3.2.1 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.

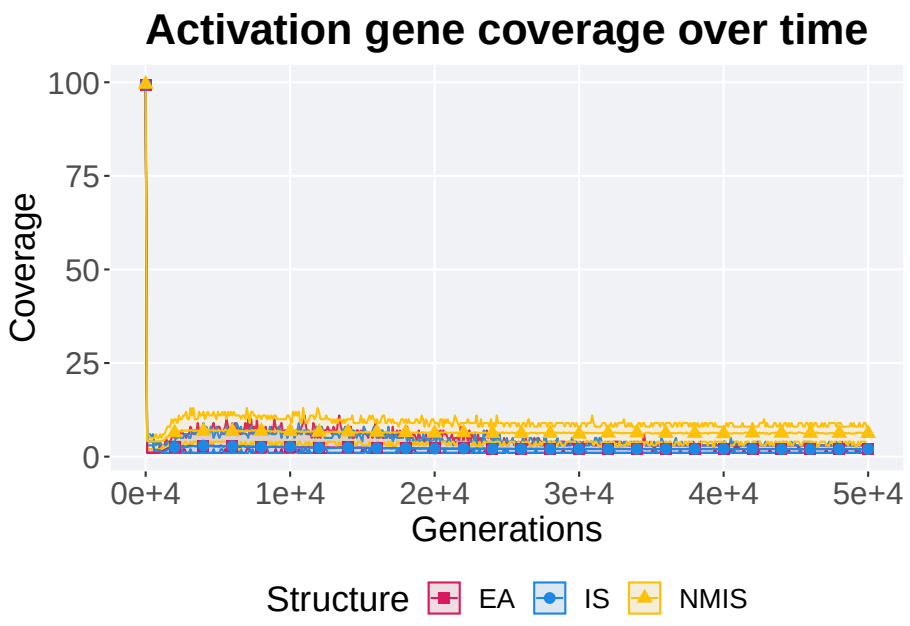
5.3.3.1 Coverage over time

Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TO
  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
```

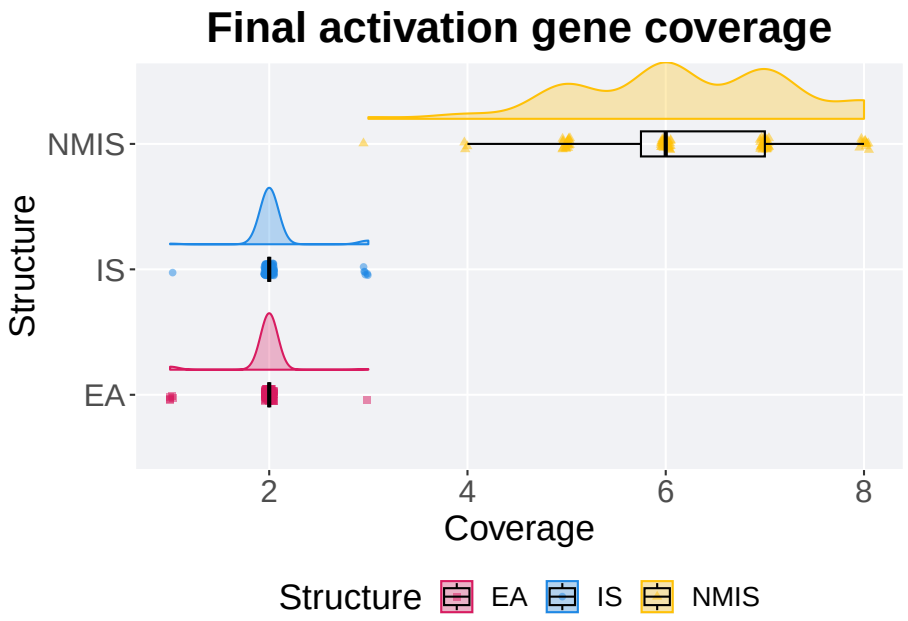
```
## `.groups` argument.
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
  scale_y_continuous(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



5.3.3.2 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 = 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()
```



5.3.3.2.1 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  2.05     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')
```

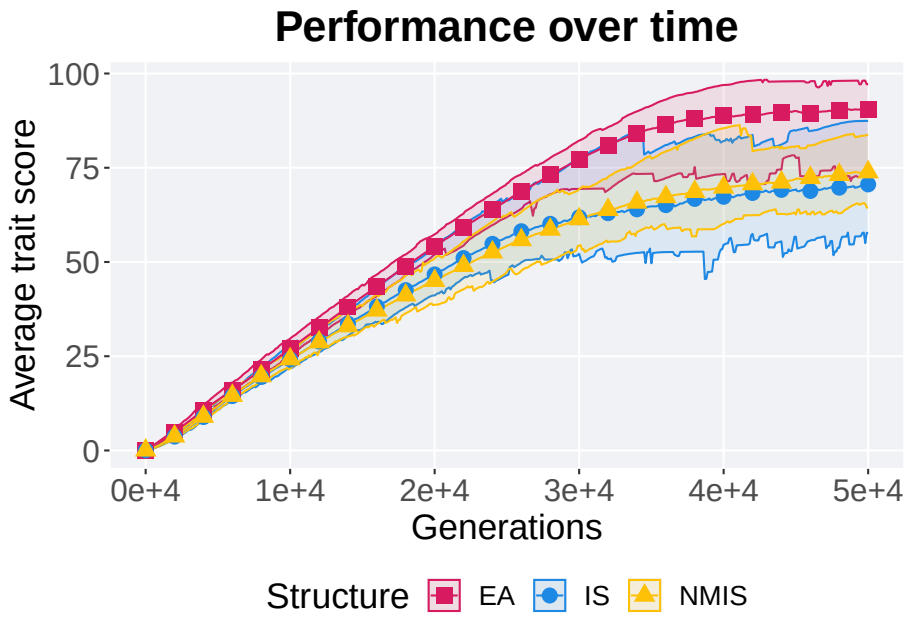
```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_act_cov and coverage$Structure
##
##       EA     IS
## IS   0.019  -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

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

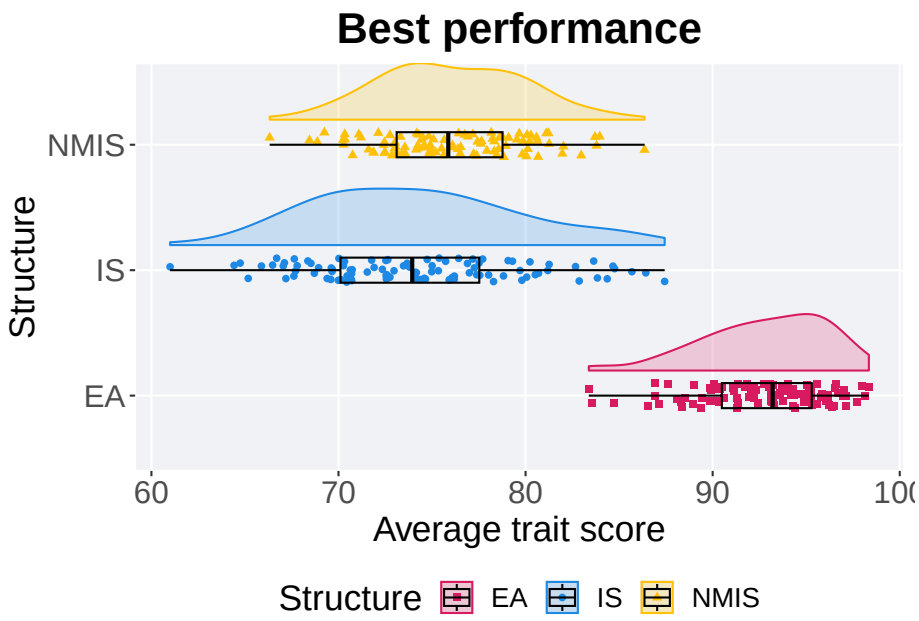
```
lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'LE
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
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 = 
  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(VAL ~ Structure, data = performance)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  VAL 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.

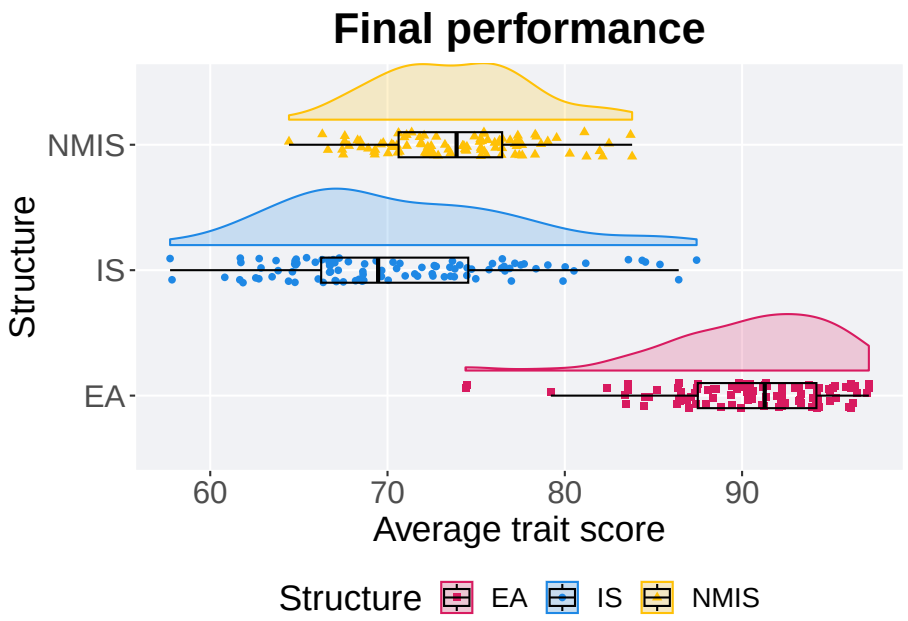
```
pairwise.wilcox.test(x = performance$VAL, g = performance$Structure, p.adjust.method =
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VAL and performance$Structure
##
##      EA     NMIS
## NMIS <2e-16 -
## IS   <2e-16 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 = 
  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
                     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

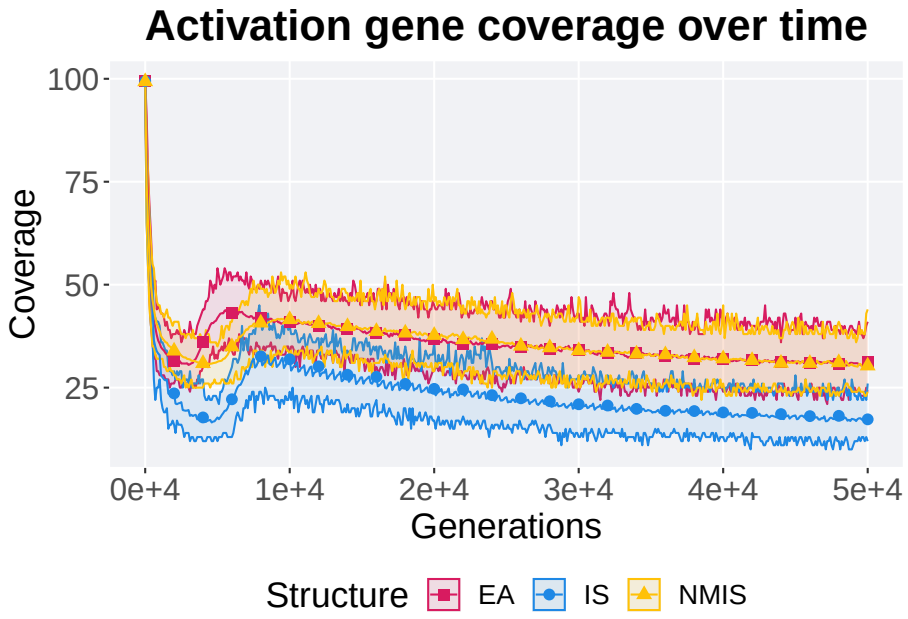
Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nSche
  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
```

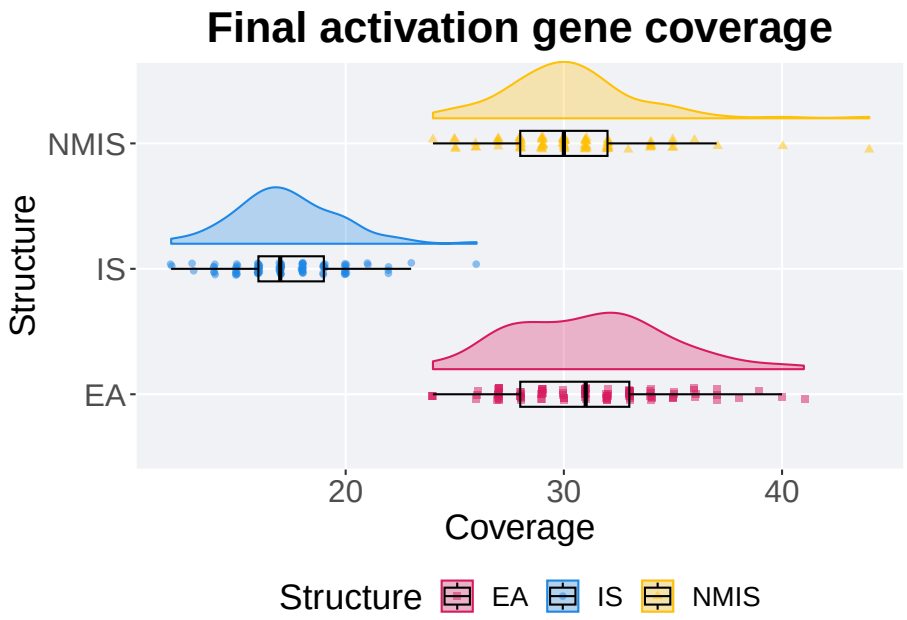


5.4.2.2 End of 50,000 generations

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 = 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()
```



5.4.2.2.1 Stats

Summary statistics for activation gene coverage.

```
coverage = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nS
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 = "bonferr
                     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
```