

Supplemental Material for Adaptive phenotypic
plasticity stabilizes evolution in fluctuating
environments

Alexander Lalejini, Austin J. Ferguson, Nkrumah A. Grant, and Charles Ofria

2021-05-22

Contents

1	Introduction	5
1.1	About our supplemental material	5
1.2	Contributing authors	5
1.3	Research overview	6
2	Data availability	7
2.1	Source code	7
2.2	Experimental results	7
3	Compile and run experiments locally	9
3.1	Docker	9
4	Validation experiment	11
4.1	Overview	11
4.2	Analysis dependencies	12
4.3	Setup	12
4.4	Evolution of phenotypic plasticity	14
5	Evolutionary change	17
5.1	Overview	17
5.2	Analysis dependencies	17
5.3	Setup	18
5.4	The evolution of phenotypic plasticity	20
5.5	Average generation	22
5.6	Coalescence event count	25
5.7	Phenotypic volatility along the dominant lineage	33
5.8	Phenotypic fidelity	39
5.9	Mutation count	43
5.10	Genotypic fidelity	50
5.11	Characterizing variation along dominant lineages	54
5.12	Manuscript figures	66
6	Evolution and maintenance of novel traits	69
6.1	Overview	69

6.2	Analysis dependencies	70
6.3	Setup	71
6.4	The evolution of phenotypic plasticity	73
6.5	Final novel task count (dominant genotype)	75
6.6	Novel task count (final population)	79
6.7	Novel task discovery (lineage)	82
6.8	Novel task discovery (population)	85
6.9	Novel task discovery frequency (lineage)	88
6.10	Novel tasks gained (lineage)	92
6.11	Novel task loss (lineage)	94
6.12	Frequency of novel task loss (lineage)	98
6.13	How many instances of novel trait loss co-occurred with changes in base phenotype?	101
6.14	Manuscript figures	105
6.15	Combined panel	105
7	Accumulation of deleterious instructions	111
7.1	Overview	111
7.2	Analysis dependencies	111
7.3	Setup	112
7.4	Evolution of phenotypic plasticity	115
7.5	Poison instruction execution	117
7.6	Characterizing mutations that increase poison instruction execution	128
7.7	What fraction of poison execution increases occur in unexpressed phenotype (as cryptic variation)?	146
7.8	Manuscript figures	147
8	Regulation in Avida	149
8.1	Overview	149
8.2	Analysis dependencies	149
8.3	Setup	150
8.4	How many instructions do plastic genomes toggle depending on environmental context?	151
8.5	What is the distribution of toggled sequence sizes?	152
9	Evolutionary change (variable length genomes)	155
9.1	Overview	155
9.2	Analysis dependencies	155
9.3	Setup	156
9.4	Evolution of phenotypic plasticity	157
9.5	Genome length	159
9.6	Average generation	162
9.7	Coalescence events	163
9.8	Phenotypic volatility along the dominant lineage	168
9.9	Mutation accumulation along the dominant lineage	171

Chapter 1

Introduction

This is the supplemental material for our work entitled, *Adaptive phenotypic plasticity stabilizes evolution in fluctuating environments*. Preprint forthcoming.

1.1 About our supplemental material

This supplemental material is hosted on GitHub using GitHub pages. The source code and configuration files used to generate this supplemental material can be found in this GitHub repository. We compiled our data analyses and supplemental documentation into this nifty web-accessible book using bookdown.

Our supplemental material includes the following:

- Data availability (Section 2)
- Guide for running our experiments (Section 3)
- Experiment analyses (including source code):
 - Validating the evolution of phenotypic plasticity (Section TODO)
 - Effect of adaptive phenotypic plasticity on evolutionary change (Section TODO)
 - * Results with variable-length genomes (Section TODO)
 - Effect of adaptive phenotypic plasticity on the evolution and maintenance of novel traits (Section TODO)
 - Effect of adaptive phenotypic plasticity on the accumulation of deleterious instructions (Section TODO)
 - Exploring how regulation is encoded in genomes in Avida (Section TODO)

1.2 Contributing authors

- Alexander Lalejini

- Austin J. Ferguson
- Nkrumah A. Grant
- Charles Ofria

1.3 Research overview

Abstract: > Fluctuating environmental conditions are ubiquitous in natural systems, and populations have evolved various strategies to cope with such fluctuations. The particular mechanisms that evolve profoundly influence subsequent evolutionary dynamics. One such mechanism is phenotypic plasticity, which is the ability of a single genotype to produce alternate phenotypes in an environmentally dependent context. Here, we use digital organisms (self-replicating computer programs) to investigate how adaptive phenotypic plasticity alters evolutionary dynamics and influences evolutionary outcomes in cyclically changing environments. Specifically, we examined the evolutionary histories of both plastic populations and non-plastic populations to ask: (1) Does adaptive plasticity promote or constrain evolutionary change? (2) Are plastic populations better able to evolve and then maintain novel traits? And (3), how does adaptive plasticity affect the potential for maladaptive traits to accumulate in evolving genomes? We find that populations with adaptive phenotypic plasticity undergo less evolutionary change than non-plastic populations, which must rely on genetic variation from de novo mutations to continuously readapt to environmental fluctuations. Indeed, the non-plastic populations undergo more frequent selective sweeps and accumulate many more genetic changes. We find that the repeated selected sweeps in non-plastic populations drive the loss of beneficial traits via deleterious hitchhiking, whereas phenotypic plasticity can stabilize populations against environmental fluctuations. This stabilization allows plastic populations to more easily retain novel adaptive traits than their non-plastic counterparts. In general, the evolution of adaptive phenotypic plasticity shifted evolutionary dynamics to be more similar to that of populations evolving in a static environment than to non-plastic populations evolving in an identical fluctuating environment. All natural environments subject populations to some form of change; our findings suggest that the stabilizing effect of phenotypic plasticity plays an important role in subsequent adaptive evolution.

Chapter 2

Data availability

2.1 Source code

The source code for this work is publicly accessible on GitHub: <https://github.com/amlalejini/evolutionary-consequences-of-plasticity>

2.2 Experimental results

The data from our experiments are available online in our OSF repository (Lalejini and Ferguson, 2021) at <https://osf.io/sav2c/>.

Chapter 3

Compile and run experiments locally

Here, we provide a brief guide to compiling and running our experiments using our Docker image.

Please file an issue on GitHub if something is unclear or does not work.

3.1 Docker

You can use the Dockerfile in our repository to build a docker image locally, or you can pull the latest docker image from DockerHub using

```
docker pull amlalejini/evolutionary-consequences-of-plasticity
```

This will pull down a docker image with:

- all of the requisite dependencies installed/downloaded
- all experiment source code
- the minimal set of raw data needed to compile the supplemental material
- a build of our supplemental material (which will also run all of our analyses)

To run the container interactively:

```
docker run -it --entrypoint bash amlalejini/evolutionary-consequences-of-plasticity
```

You can exit the container at any point with `ctrl-d`.

Inside the container, you should be able to navigate to `/opt/evolutionary-consequences-of-plasticity`:

```
cd /opt/evolutionary-consequences-of-plasticity
```

To run Avida, you'll need to `cd` into the `avida` directory and run `./build_avida`.

All of the Avida configuration files necessary for re-running our experiments can be found here: <https://github.com/amlalejini/evolutionary-consequences-of-plasticity/tree/master/experiments>.

For example, the configuration files for our evolutionary change experiment are here: <https://github.com/amlalejini/evolutionary-consequences-of-plasticity/tree/master/experiments/2021-02-08-evo-dynamics/hpcc/config>.

Chapter 4

Validation experiment

In this experiment, we validate that (1) we observe the evolution of phenotypic plasticity in a changing environment when digital organisms have access to sensory instructions (capable of differentiating environmental states) and (2) that adaptive phenotypic plasticity does not evolve when populations lack access to sensory instructions.

4.1 Overview

```
total_updates <- 200000
replicates <- 100

all_traits <- c("not","nand","and","ornot","or","andnot")
traits_set_a <- c("not", "and", "or")
traits_set_b <- c("nand", "ornot", "andnot")

# Relative location of data.
working_directory <- "experiments/2021-01-07-validation/analysis/" # << For bookdown
# working_directory <- "./" # << For local analysis
```

We evolved populations of digital organisms under four conditions:

1. A fluctuating environment with access to sensory instructions
2. A fluctuating environment without access to sensory instructions (i.e., sensory instructions are no-operations)
3. A constant environment with access to sensory instructions
4. A constant environment without access to sensory instructions

In fluctuating environments, we alternate between rewarding and punishing different sets of computational tasks. In one environment, we reward tasks not,

and, or and punish tasks nand, ornot, andnot. In the alternative environment, we reward tasks nand, ornot, andnot and punish tasks not, and, or. In constant environments, we reward all tasks (not, nand, and, ornot, or, andnot).

For each replicate of each condition, we extract the dominant (i.e., most numerous) genotype at the end of the run to analyze further. We expect to observe the evolution of adaptive phenotypic plasticity in only the first experimental condition. In conditions without sensors, plasticity in any form should be unable to evolve.

4.2 Analysis dependencies

Load all required R libraries.

```
library(ggplot2)
library(tidyverse)
library(cowplot)
source("https://gist.githubusercontent.com/benmarwick/2a1bb0133ff568cbe28d/raw/fb53bd9")
```

These analyses were conducted/knitted with the following computing environment:

```
print(version)

##
## platform      x86_64-pc-linux-gnu
## arch          x86_64
## os            linux-gnu
## system        x86_64, linux-gnu
## status
## major         4
## minor         1.0
## year          2021
## month         05
## day           18
## svn rev       80317
## language      R
## version.string R version 4.1.0 (2021-05-18)
## nickname      Camp Pontanezen
```

4.3 Setup

```
data_loc <- paste0(working_directory, "data/aggregate.csv")
data <- read.csv(data_loc, na.strings="NONE")

data$DISABLE_REACTION_SENSORS <- as.factor(data$DISABLE_REACTION_SENSORS)
```

```

data$chg_env <- as.factor(data$chg_env)
data$dom_plastic_odd_even <- as.factor(data$dom_plastic_odd_even)
data$sensors <- data$DISABLE_REACTION_SENSORS == "0"
data$is_plastic <- data$dom_plastic_odd_even == "True"

env_label_fun <- function(chg_env) {
  if (chg_env) {
    return("Fluctuating")
  } else {
    return("Constant")
  }
}

sensors_label_fun <- function(has_sensors) {
  if (has_sensors) {
    return("Sensors")
  } else {
    return("No sensors")
  }
}

# Count observed plasticity for each condition (I'm sure there's a 'tidier' way to do this..)
observed_plasticity <- data.frame(
  environment=character(),
  sensors=character(),
  plastic=integer(),
  nonplastic=integer(),
  plastic_adaptive=integer(),
  plastic_optimal=integer(),
  plastic_nonadaptive=integer()
)

for (env_chg in levels(data$chg_env)) {
  for (disabled_sensors in levels(data$DISABLE_REACTION_SENSORS)) {
    cond_data <- filter(data, chg_env == env_chg & data$DISABLE_REACTION_SENSORS == disabled_sensors)
    environment_label <- env_label_fun(env_chg)
    sensors_label <- sensors_label_fun(disabled_sensors == "0")

    observed_plasticity <- observed_plasticity %>% add_row(
      environment=environment_label,
      sensors=sensors_label,
      plastic=nrow(filter(cond_data, is_plastic==TRUE)),
      nonplastic=nrow(filter(cond_data, is_plastic==FALSE)),
      plastic_adaptive=nrow(filter(cond_data, dom_adaptive_plasticity=="True")),
      plastic_optimal=nrow(filter(cond_data, dom_optimal_plastic=="True")),
      plastic_nonadaptive=nrow(filter(cond_data, is_plastic==TRUE & dom_adaptive_plasticity=="False"))
    )
  }
}

```

```

    )
  }
}

observed_plasticity <- pivot_longer(
  observed_plasticity,
  cols=c("plastic", "plastic_adaptive", "plastic_optimal", "plastic_nonadaptive", "nonplastic"),
  names_to="phenotype",
  values_to="phenotype_cnt"
)

##### misc #####
# Configure our default graphing theme
theme_set(theme_cowplot())

```

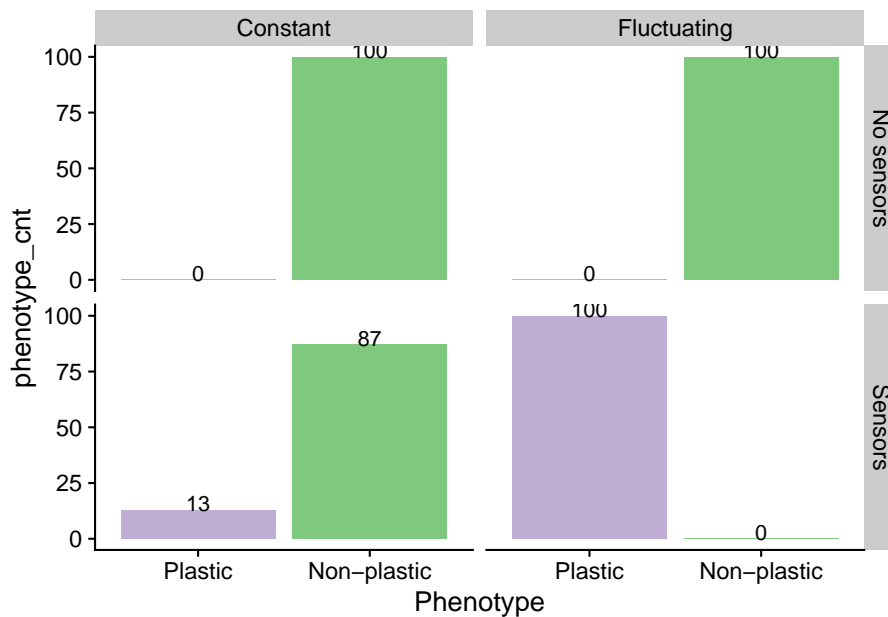
4.4 Evolution of phenotypic plasticity

For each experimental condition, do we observe the evolution of phenotypic plasticity? To test for phenotypic plasticity, we culture digital organisms in both environments from the fluctuating condition (including organisms evolved in a constant environment). Any plasticity that we observe from digital organisms evolved under constant conditions is cryptic variation (as these organisms were never exposed to these culturing environments).

```

ggplot(filter(observed_plasticity, phenotype %in% c("plastic", "nonplastic")), aes(x=phenotype, y=phenotype_cnt)) +
  geom_bar(
    stat="identity",
    position=position_dodge(0.9)
  ) +
  geom_text(
    stat="identity",
    mapping=aes(label=phenotype_cnt),
    vjust=0.05
  ) +
  scale_fill_brewer(palette="Accent") +
  scale_x_discrete(
    name="Phenotype",
    limits=c("plastic", "nonplastic"),
    labels=c("Plastic", "Non-plastic")
  ) +
  facet_grid(sensors~environment) +
  theme(
    legend.position="none"
  )

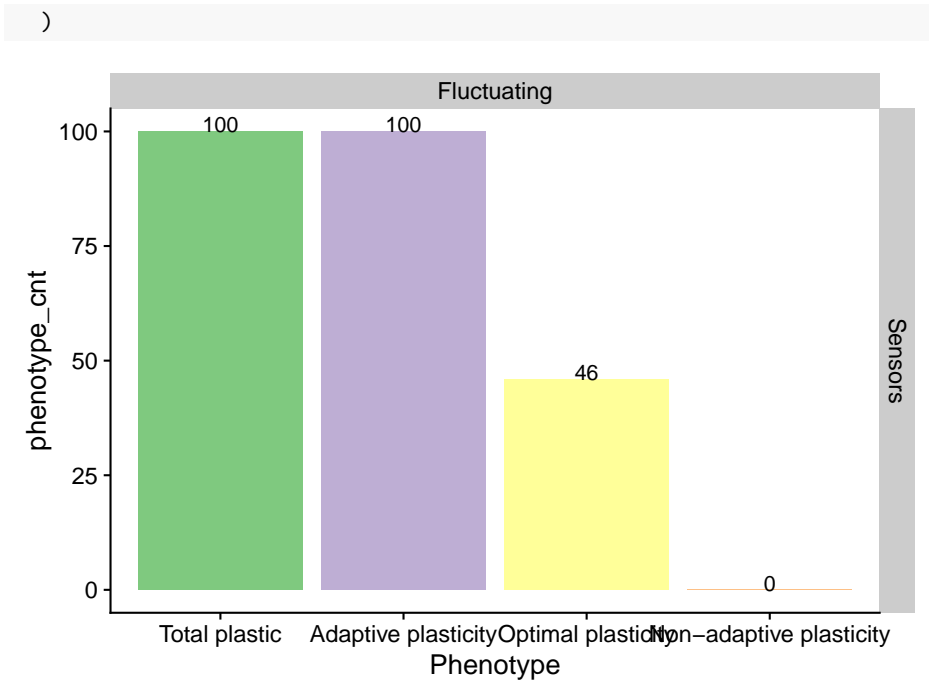
```



Indeed, we do not observe the evolution of phenotypic plasticity in any replicates in which digital organisms do not have access to sensory instructions. We do observe the evolution of plasticity (not necessarily adaptive plasticity) in both constant and fluctuating environments where sensors are enabled.

To what extent is the observed phenotypic plasticity adaptive?

```
ggplot(filter(observed_plasticity, environment=="Fluctuating" & sensors == "Sensors" & phenotype
  geom_bar(
    stat="identity",
    position=position_dodge(0.9)
  ) +
  geom_text(
    stat="identity",
    mapping=aes(label=phenotype_cnt),
    vjust=0.05
  ) +
  scale_fill_brewer(palette="Accent") +
  scale_x_discrete(
    name="Phenotype",
    limits=c("plastic", "plastic_adaptive", "plastic_optimal", "plastic_nonadaptive"),
    labels=c("Total plastic", "Adaptive plasticity", "Optimal plasticity", "Non-adaptive plasticity")
  ) +
  facet_grid(sensors~environment) +
  theme(
    legend.position="none"
```



Chapter 5

Evolutionary change

The effect of adaptive phenotypic plasticity on evolutionary change.

5.1 Overview

```
total_updates <- 200000
replicates <- 100
alpha <- 0.05

all_traits <- c("not", "nand", "and", "ornot", "or", "andnot")
traits_set_a <- c("not", "and", "or")
traits_set_b <- c("nand", "ornot", "andnot")

# Relative location of data.
working_directory <- "experiments/2021-02-08-evo-dynamics/analysis/" # << For bookdown
# working_directory <- "./" # << For local analysis
```

5.2 Analysis dependencies

Load all required R libraries.

```
library(ggplot2)
library(rstatix)
library(ggsignif)
library(scales)
library(tidyverse)
library(cowplot)
library(RColorBrewer)
library(Hmisc)
```

```
library(boot)
source("https://gist.githubusercontent.com/benmarwick/2a1bb0133ff568cbe28d/raw/fb53bd9
```

These analyses were conducted/knitted with the following computing environment:

```
print(version)

##
## platform      x86_64-pc-linux-gnu
## arch          x86_64
## os            linux-gnu
## system        x86_64, linux-gnu
## status
## major         4
## minor         1.0
## year          2021
## month         05
## day           18
## svn rev       80317
## language      R
## version.string R version 4.1.0 (2021-05-18)
## nickname      Camp Pontanezen
```

5.3 Setup

```
summary_data_loc <- paste0(working_directory, "data/aggregate.csv")
summary_data <- read.csv(summary_data_loc, na.strings="NONE")

summary_data$DISABLE_REACTION_SENSORS <- as.factor(summary_data$DISABLE_REACTION_SENSORS)
summary_data$chg_env <- summary_data$chg_env == "True"
summary_data$dominant_plastic_odd_even <- as.factor(summary_data$dominant_plastic_odd_even)
summary_data$sensors <- summary_data$DISABLE_REACTION_SENSORS == "0"
summary_data$is_plastic <- summary_data$dominant_plastic_odd_even == "True"

env_label_fun <- function(chg_env) {
  if (chg_env) {
    return("Fluctuating")
  } else {
    return("Constant")
  }
}

sensors_label_fun <- function(has_sensors) {
  if (has_sensors) {
```

```

    return("Sensors")
  } else {
    return("No sensors")
  }
}

# note that this labeler makes assumptions about how we set up our experiment
condition_label_fun <- function(has_sensors, env_chg) {
  if (has_sensors && env_chg) {
    return("PLASTIC")
  } else if (env_chg) {
    return("NON-PLASTIC")
  } else {
    return("STATIC")
  }
}

summary_data$env_label <- mapply(
  env_label_fun,
  summary_data$chg_env
)
summary_data$sensors_label <- mapply(
  sensors_label_fun,
  summary_data$sensors
)
summary_data$condition <- mapply(
  condition_label_fun,
  summary_data$sensors,
  summary_data$chg_env
)

condition_order = c(
  "STATIC",
  "NON-PLASTIC",
  "PLASTIC"
)

pairwise_comparisons <- list(
  c("STATIC", "NON-PLASTIC"),
  c("STATIC", "PLASTIC"),
  c("PLASTIC", "NON-PLASTIC")
)

p_label <- function(p_value) {
  threshold = 0.0001
  if (p_value < threshold) {

```

```

    return(paste0("p < ", threshold))
  } else {
    return(paste0("p = ", p_value))
  }
}

# *really* inefficient way to identify outliers
is_outlier <- function(value, cond, data, column) {
  cond_data <- filter(data, condition==cond)
  q1 <- summary(cond_data[,column])[["1st Qu."]]
  q3 <- summary(cond_data[,column])[["3rd Qu."]]
  H <- 1.5 * IQR(cond_data[,column])
  return( (value < (q1-H)) || (value > (q3+H)) )
}

##### misc #####
# Configure our default graphing theme
theme_set(theme_cowplot())
# Palette
cb_palette <- "Paired"
# Create a directory to store plots
dir.create(paste0(working_directory, "plots"), showWarnings=FALSE)
# Define sample mean function
samplemean <- function(x, d) {
  return(mean(x[d]))
}

```

5.4 The evolution of phenotypic plasticity

For sensor-enabled populations in fluctuating environments, we only transferred populations containing an optimally plastic genotype to phase-two.

```

summary_data_grouped = dplyr::group_by(summary_data, condition)
summary_data_group_counts = dplyr::summarize(summary_data_grouped, n=dplyr::n())

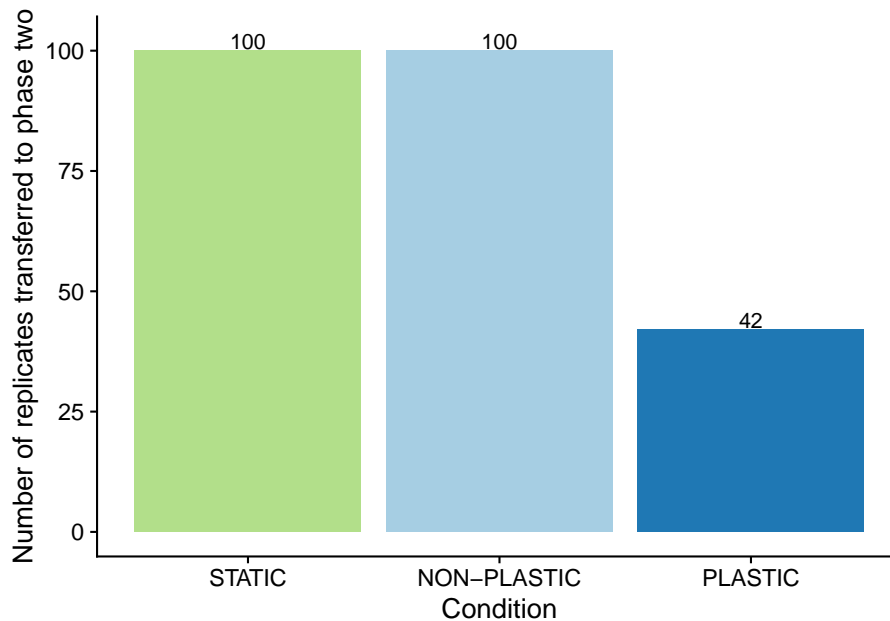
ggplot(summary_data_group_counts, aes(x=condition, y=n, fill=condition)) +
  geom_col(position=position_dodge(0.9)) +
  geom_text(aes(label=n, y=n+2)) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +

```

```

scale_color_brewer(
  palette=cb_palette
) +
ylab("Number of replicates transferred to phase two") +
theme(
  legend.position="none"
)

```



We can confirm our expectation that the dominant genotypes in non-plastic conditions are not phenotypically plastic.

```

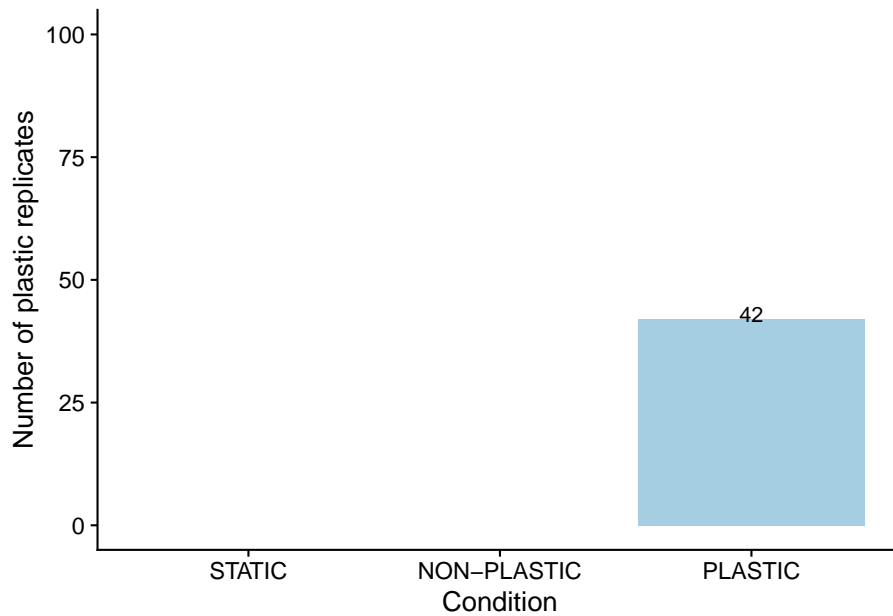
summary_data_grouped = dplyr::group_by(summary_data, condition, is_plastic)
summary_data_group_counts = dplyr::summarize(summary_data_grouped, n=dplyr::n())
ggplot(filter(summary_data_group_counts, is_plastic), aes(x=condition, y=n, fill=condition)) +
  geom_col(
    position=position_dodge(0.9)
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(

```

```

palette=cb_palette
) +
geom_text(aes(label=n, y=n+1)) +
ylab("Number of plastic replicates") +
ylim(0, 100) +
theme(
  legend.position="none"
)

```



5.5 Average generation

```

ggplot(summary_data, aes(x=condition, y=time_average_generation, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(

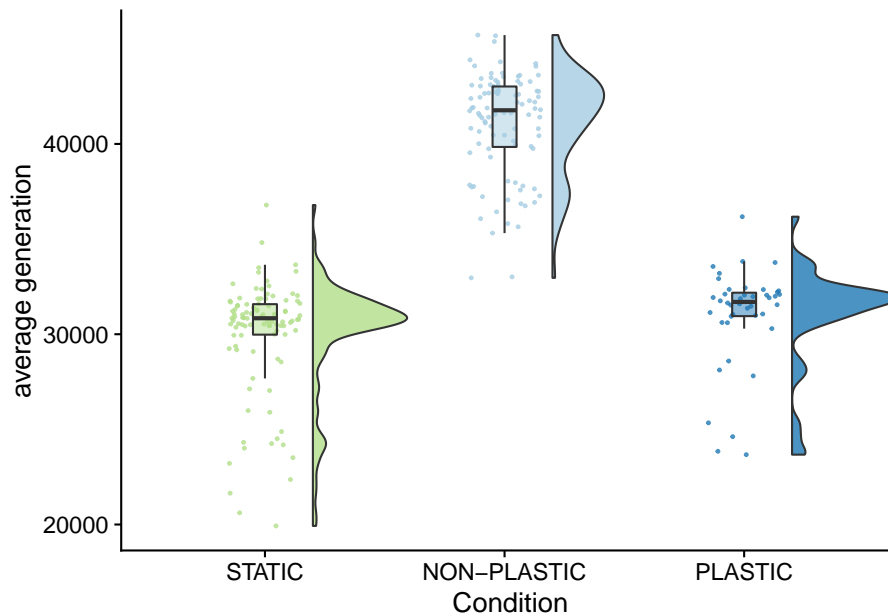
```

```

width = .1,
outlier.shape = NA,
alpha = 0.5
) +
scale_x_discrete(
  name="Condition",
  limits=condition_order
) +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
# coord_flip() +
ylab("average generation") +
theme(
  legend.position="none"
) +
ggsave(paste0(working_directory, "plots/", "average-generation.png"))

```

Saving 6.5 x 4.5 in image



```

kruskal.test(
  formula=time_average_generation~condition,

```

```

data=summary_data
)

##
##  Kruskal-Wallis rank sum test
##
## data:  time_average_generation by condition
## Kruskal-Wallis chi-squared = 177.33, df = 2, p-value < 2.2e-16
pairwise.wilcox.test(
  x=summary_data$time_average_generation,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  summary_data$time_average_generation and summary_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      0.004
##
## P value adjustment method: bonferroni
paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$time_average_generation)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$time_average_generation)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$time_average_generation)
  )
)

## [1] "PLASTIC median: 31697.65; STATIC median: 30839.75; NON-PLASTIC median: 41768.65"
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"

```



```

for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=time_average_generation~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

## [1] "STATIC<-->NON-PLASTIC: W=9982"
## [1] "STATIC<-->PLASTIC: W=2818"
## [1] "PLASTIC<-->NON-PLASTIC: W=4186"

summary_data %>%
  group_by(condition) %>%
  summarise(mean=mean(time_average_generation), sd=sd(time_average_generation))

## # A tibble: 3 x 3
##   condition    mean    sd
##   <chr>      <dbl> <dbl>
## 1 NON-PLASTIC 41090. 2702.
## 2 PLASTIC    31016. 2615.
## 3 STATIC    30002. 3011.

```

5.6 Coalescence event count

The number of times the most recent common ancestor changes gives us the number of selective sweeps that occur during the experiment.

```

# Compute manual labels for geom_signif
stat.test <- summary_data %>%
  wilcox_test(phylo_mrca_changes ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="condition", step.increase=1)
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior in g
stat.test$manual_position <- log10(stat.test$y.position) * c(1.0, 1.0, 1.03)
stat.test$label <- map(p_label, stat.test$p.adj)

summary_data$is_outlier <- map(
  is_outlier,
  summary_data$phylo_mrca_changes,
  summary_data$condition,

```

```

MoreArgs=list(data=summary_data, column="phylo_mrca_changes")
)

coalescence_events_fig <- ggplot(
  summary_data,
  aes(x=condition, y=phylo_mrca_changes, fill=condition)
) +
  geom_flat_violin(
    # data=filter(summary_data, is_outlier==FALSE),
    scale="width",
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order,
    labels=condition_order,
    breaks=condition_order
  ) +
  scale_y_continuous(
    name="Coalescence event count (log scale)",
    trans=pseudo_log_trans(sigma = 1, base = 10),
    breaks=c(0, 10, 100, 1000, 10000),
    limits=c(-1, 35000)
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  labs(
    subtitle=paste0(
      "Kruskal-Wallis, ",

```

```

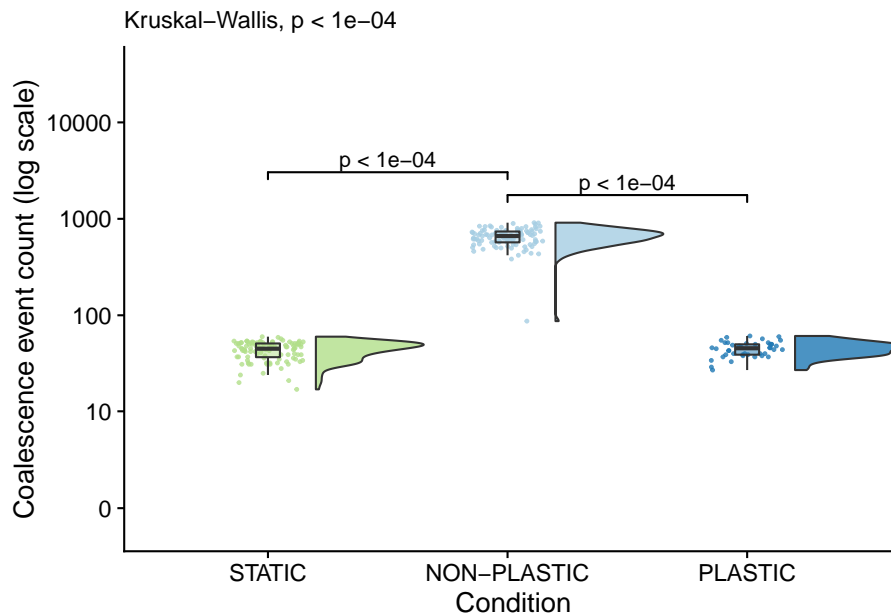
    p_label(signif(kruskal.test(formula=phylo_mrca_changes~condition, data=summary_data)$p.value
  )
) +
ggsignif::geom_signif(
  data=filter(stat.test, p.adj <= alpha),
  aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
  manual=TRUE,
  inherit.aes=FALSE
) +
# coord_flip() +
theme(
  legend.position="none"
) +
ggsave(
  paste0(working_directory, "plots/", "selective-sweeps.pdf"),
  width=5,
  height=5
)

```

```

## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
coalescence_events_fig

```



```

kruskal.test(
  formula=phylo_mrca_changes~condition,
  data=summary_data
)

```

```

)

##
## Kruskal-Wallis rank sum test
##
## data: phylo_mrca_changes by condition
## Kruskal-Wallis chi-squared = 175.46, df = 2, p-value < 2.2e-16
pairwise.wilcox.test(
  x=summary_data$phylo_mrca_changes,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$phylo_mrca_changes and summary_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      1
##
## P value adjustment method: bonferroni
paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$phylo_mrca_changes)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$phylo_mrca_changes)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$phylo_mrca_changes)
  )
)

## [1] "PLASTIC median: 45.5; STATIC median: 45; NON-PLASTIC median: 663.5"
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"

```

```
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=phylo_mrca_changes~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}
```

```
## [1] "STATIC<-->NON-PLASTIC: W=10000"
## [1] "STATIC<-->PLASTIC: W=2215"
## [1] "PLASTIC<-->NON-PLASTIC: W=4200"
```

5.6.1 Average number of generations between coalescence events

```
# Compute frequency of coalescence events
summary_data$generations_per_mrca_change <- summary_data$time_average_generation / summary_data$phylo_mrca_change

# Compute manual labels for geom_signif
stat.test <- summary_data %>%
  wilcox_test(generations_per_mrca_change ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="condition")
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior in g
stat.test$manual_position <- stat.test$y.position
stat.test$label <- mapply(p_label, stat.test$p.adj)

summary_data$is_outlier <- mapply(
  is_outlier,
  summary_data$generations_per_mrca_change,
  summary_data$condition,
  MoreArgs=list(data=summary_data, column="generations_per_mrca_change")
)

coalescence_events_freq_fig <- ggplot(
  summary_data,
  aes(x=condition, y=generations_per_mrca_change, fill=condition)
) +
  geom_flat_violin(
    # data=filter(summary_data, is_outlier==FALSE),
```

```

    scale="width",
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order,
    labels=condition_order
  ) +
  scale_y_continuous(
    name="Avg. generations between coalescence events",
    limits=c(0, 2000),
    breaks=seq(0, 2000, 500)
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  # coord_flip() +
  labs(
    subtitle=paste0(
      "Kruskal-Wallis, ",
      p_label(signif(kruskal.test(formula=generations_per_mrca_change~condition, data=
    )
  ) +
  ggsignif::geom_signif(
    data=filter(stat.test, p.adj <= alpha),
    aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
    manual=TRUE,
    inherit.aes=FALSE
  ) +
  theme(

```

```

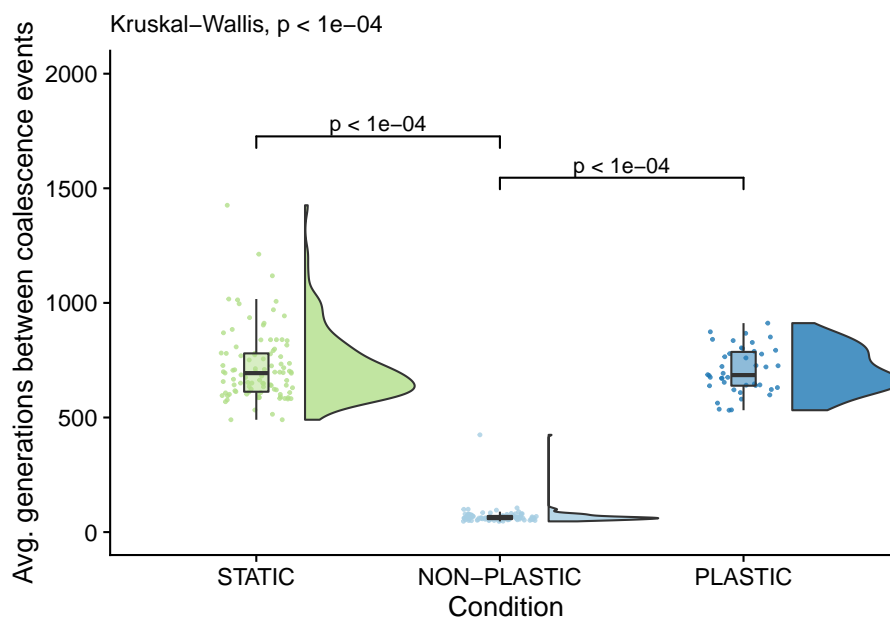
    legend.position="none"
  ) +
  ggsave(
    paste0(working_directory, "plots/", "generations-between-selective-sweeps.png"),
    width=5,
    height=5
  )

```

```

## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
coalescence_events_freq_fig

```



```

kruskal.test(
  formula=generations_per_mrca_change~condition,
  data=summary_data
)

```

```

##
## Kruskal-Wallis rank sum test
##
## data: generations_per_mrca_change by condition
## Kruskal-Wallis chi-squared = 175.33, df = 2, p-value < 2.2e-16
pairwise.wilcox.test(
  x=summary_data$generations_per_mrca_change,
  g=summary_data$condition,
  p.adjust.method="bonferroni",

```

```

)

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$generations_per_mrca_change and summary_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      1
##
## P value adjustment method: bonferroni
paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$generations_per_mrca_change)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$generations_per_mrca_change)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$generations_per_mrca_change)
  )
)

## [1] "PLASTIC median: 685.001780758557; STATIC median: 693.676265008576; NON-PLASTIC
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=generations_per_mrca_change~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

## [1] "STATIC<-->NON-PLASTIC: W=0"

```


5.7. PHENOTYPIC VOLATILITY ALONG THE DOMINANT LINEAGE33

```
## [1] "STATIC<-->PLASTIC: W=2151"  
## [1] "PLASTIC<-->NON-PLASTIC: W=0"
```

5.7 Phenotypic volatility along the dominant lineage

```
# Compute manual labels for geom_signif  
stat.test <- summary_data %>%  
  wilcox_test(dominant_lineage_trait_volatility ~ condition) %>%  
  adjust_pvalue(method = "bonferroni") %>%  
  add_significance() %>%  
  add_xy_position(x="condition", step.increase=1)  
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior in g  
stat.test$manual_position <- log10(stat.test$y.position) * c(1.0,1.0,1.03)  
stat.test$label <- mapply(p_label,stat.test$p.adj)  
  
summary_data$is_outlier <- mapply(  
  is_outlier,  
  summary_data$dominant_lineage_trait_volatility,  
  summary_data$condition,  
  MoreArgs=list(data=summary_data, column="dominant_lineage_trait_volatility")  
)  
  
phenotypic_volatility_fig <- ggplot(  
  summary_data,  
  aes(x=condition, y=dominant_lineage_trait_volatility, fill=condition)  
) +  
  geom_flat_violin(  
    # data=filter(summary_data,is_outlier==FALSE),  
    scale="width",  
    position = position_nudge(x = .2, y = 0),  
    alpha = .8  
) +  
  geom_point(  
    mapping=aes(color=condition),  
    position = position_jitter(width = .15),  
    size = .5,  
    alpha = 0.8  
) +  
  geom_boxplot(  
    width = .1,  
    outlier.shape = NA,  
    alpha = 0.5  
) +
```

```

scale_x_discrete(
  name="Condition",
  limits=condition_order,
  labels=condition_order
) +
scale_y_continuous(
  name="Phenotypic volatility (log scale)",
  trans=pseudo_log_trans(sigma = 1, base = 10),
  breaks=c(0, 10, 100, 1000, 10000),
  limits=c(-1, 35000)
) +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
labs(
  subtitle=paste0(
    "Kruskal-Wallis, ",
    p_label(signif(kruskal.test(formula=dominant_lineage_trait_volatility~condition,
  )
) +
ggsignif::geom_signif(
  data=filter(stat.test, p.adj<=alpha),
  aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
  manual=TRUE,
  inherit.aes=FALSE
) +
# coord_flip() +
theme(
  legend.position="none"
) +
ggsave(
  paste0(working_directory, "plots/", "phenotypic-volatility.pdf"),
  width=5,
  height=5
)

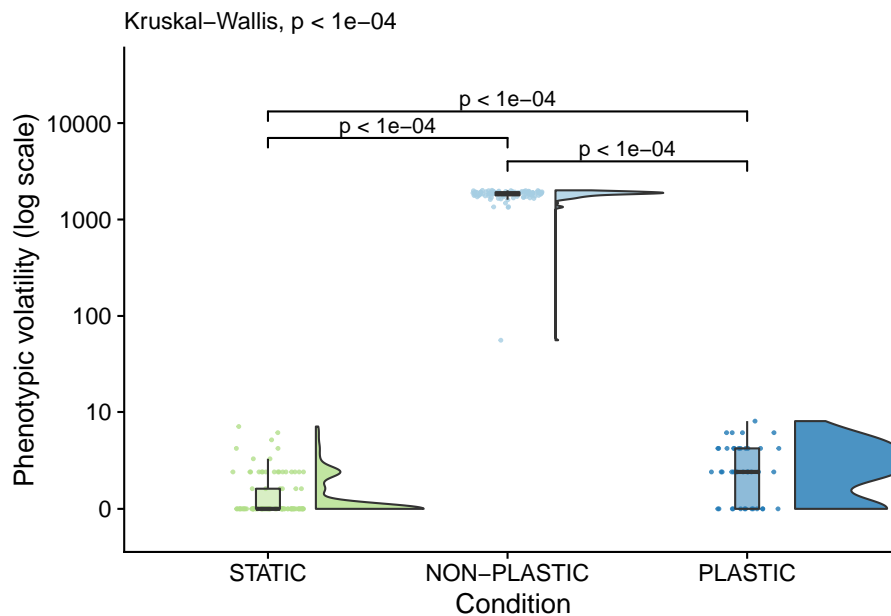
```

```

## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
phenotypic_volatility_fig

```

5.7. PHENOTYPIC VOLATILITY ALONG THE DOMINANT LINEAGE35



```
kruskal.test(
  formula=dominant_lineage_trait_volatility~condition,
  data=summary_data
)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_trait_volatility by condition
## Kruskal-Wallis chi-squared = 190.78, df = 2, p-value < 2.2e-16
```

```
pairwise.wilcox.test(
  x=summary_data$dominant_lineage_trait_volatility,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$dominant_lineage_trait_volatility and summary_data$condition
##
##      NON-PLASTIC PLASTIC
## PLASTIC < 2e-16      -
## STATIC  < 2e-16      8.7e-07
##
```

```
## P value adjustment method: bonferroni
```

```
paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$dominant_lineage_trait_volatility,
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$dominant_lineage_trait_volatility,
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$dominant_lineage_trait_volatility,
  )
)
```

```
## [1] "PLASTIC median: 2; STATIC median: 0; NON-PLASTIC median: 1868"
```

```
print("Wilcox rank sum test statistics:")
```

```
## [1] "Wilcox rank sum test statistics:"
```

```
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=dominant_lineage_trait_volatility~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}
```

```
## [1] "STATIC<-->NON-PLASTIC: W=10000"
```

```
## [1] "STATIC<-->PLASTIC: W=3116.5"
```

```
## [1] "PLASTIC<-->NON-PLASTIC: W=4200"
```

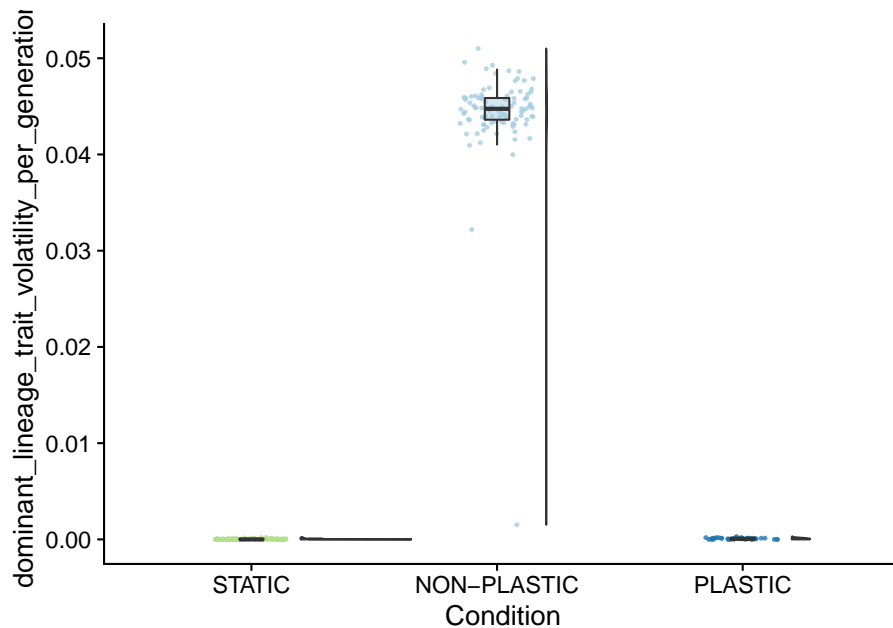
5.7.1 Phenotypic volatility normalized by generations elapsed

```
summary_data$dominant_lineage_trait_volatility_per_generation <- summary_data$dominant_lineage_trait_volatility /
  (summary_data$generations_elapsed - 1)

ggplot(summary_data, aes(x=condition, y=dominant_lineage_trait_volatility_per_generation)) +
  geom_flat_violin()
```

5.7. PHENOTYPIC VOLATILITY ALONG THE DOMINANT LINEAGE37

```
    position = position_nudge(x = .2, y = 0),
    alpha = .8
) +
geom_point(
  mapping=aes(color=condition),
  position = position_jitter(width = .15),
  size = .5,
  alpha = 0.8
) +
geom_boxplot(
  width = .1,
  outlier.shape = NA,
  alpha = 0.5
) +
scale_x_discrete(
  name="Condition",
  limits=condition_order
) +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
# coord_flip() +
theme(
  legend.position="none"
)
```



```
kruskal.test(
  formula=dominant_lineage_trait_volatility_per_generation~condition,
  data=summary_data
)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  dominant_lineage_trait_volatility_per_generation by condition
## Kruskal-Wallis chi-squared = 189.62, df = 2, p-value < 2.2e-16
```

```
pairwise.wilcox.test(
  x=summary_data$dominant_lineage_trait_volatility_per_generation,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  summary_data$dominant_lineage_trait_volatility_per_generation and summary_da
##
##           NON-PLASTIC PLASTIC
## PLASTIC < 2e-16      -
## STATIC  < 2e-16      4.2e-06
##
```

```
## P value adjustment method: bonferroni
```

```
paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$dominant_lineage_trait_volatility_per_generation
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$dominant_lineage_trait_volatility_per_generation
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$dominant_lineage_trait_volatility_per_generation
  )
)
```

```
## [1] "PLASTIC median: 6.33339279717772e-05; STATIC median: 0; NON-PLASTIC median: 0.04474401456"
```

```
print("Wilcox rank sum test statistics:")
```

```
## [1] "Wilcox rank sum test statistics:"
```

```
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=dominant_lineage_trait_volatility_per_generation~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}
```

```
## [1] "STATIC<-->NON-PLASTIC: W=10000"
```

```
## [1] "STATIC<-->PLASTIC: W=3061.5"
```

```
## [1] "PLASTIC<-->NON-PLASTIC: W=4200"
```

5.8 Phenotypic fidelity

Frequency that an offspring's genotype is identical to a parent genotype (along the dominant lineage).

```
summary_data$dominant_lineage_trait_fidelity <- (summary_data$dominant_generation_born - summary_data$dominant_generation_death) / summary_data$dominant_generation_age

# Compute manual labels for geom_signif
```

```

stat.test <- summary_data %>%
  wilcox_test(dominant_lineage_trait_fidelity ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="condition",step.increase=1.5)
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior)
stat.test$manual_position <- stat.test$y.position * c(1.0,1.0,1.0005)
stat.test$label <- mapply(p_label,stat.test$p.adj)

summary_data$is_outlier <- mapply(
  is_outlier,
  summary_data$dominant_lineage_trait_fidelity,
  summary_data$condition,
  MoreArgs=list(data=summary_data, column="dominant_lineage_trait_fidelity")
)

phenotypic_fidelity_fig <- ggplot(
  summary_data,
  aes(x=condition, y=dominant_lineage_trait_fidelity, fill=condition)
) +
  geom_flat_violin(
    data=filter(summary_data,is_outlier==FALSE),
    scale="width",
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order,
    labels=condition_order
  ) +
  scale_y_continuous(

```



```

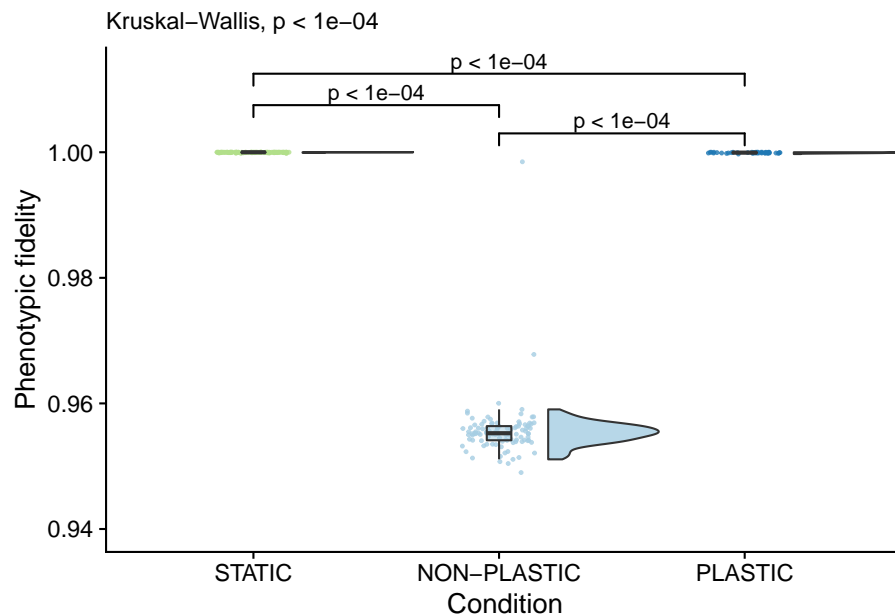
    name="Phenotypic fidelity",
    limits=c(0.94, 1.013),
    breaks=c(0.94, 0.96, 0.98, 1.0) #seq(0.94, 1.0, 0.01)
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  # coord_flip() +
  labs(
    subtitle=paste0(
      "Kruskal-Wallis, ",
      p_label(signif(kruskal.test(formula=dominant_lineage_trait_fidelity~condition, data=summary
    )
  ) +
  ggsignif::geom_signif(
    data=filter(stat.test, p.adj <= alpha),
    aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
    manual=TRUE,
    inherit.aes=FALSE
  ) +
  theme(
    legend.position="none"
  ) +
  ggsave(
    paste0(working_directory, "plots/", "phenotypic-fidelity.pdf"),
    width=5,
    height=5
  )

```

```

## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
phenotypic_fidelity_fig

```



```
kruskal.test(
  formula=dominant_lineage_trait_fidelity~condition,
  data=summary_data
)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_trait_fidelity by condition
## Kruskal-Wallis chi-squared = 189.62, df = 2, p-value < 2.2e-16
```

```
pairwise.wilcox.test(
  x=summary_data$dominant_lineage_trait_fidelity,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$dominant_lineage_trait_fidelity and summary_data$condition
##
##      NON-PLASTIC PLASTIC
## PLASTIC < 2e-16      -
## STATIC  < 2e-16      4.2e-06
##
```

```
## P value adjustment method: bonferroni
```

```
paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$dominant_lineage_trait_fidelity)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$dominant_lineage_trait_fidelity)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$dominant_lineage_trait_fidelity)
  )
)
```

```
## [1] "PLASTIC median: 0.999936666072028; STATIC median: 1; NON-PLASTIC median: 0.95525598543618"
```

```
print("Wilcox rank sum test statistics:")
```

```
## [1] "Wilcox rank sum test statistics:"
```

```
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=dominant_lineage_trait_fidelity~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}
```

```
## [1] "STATIC<-->NON-PLASTIC: W=0"
```

```
## [1] "STATIC<-->PLASTIC: W=1138.5"
```

```
## [1] "PLASTIC<-->NON-PLASTIC: W=0"
```

5.9 Mutation count

```
# Compute manual labels for geom_signif
stat.test <- summary_data %>%
  wilcox_test(dominant_lineage_total_mut_cnt ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
```

```

    add_xy_position(x="condition",step.increase=1)
    # Tweak y.position manually to account for scaled axis (edge case that triggers bad be
stat.test$manual_position <- log10(stat.test$y.position) * c(1.0,1.0,1.03) # c(1.0,1.
stat.test$label <- mapply(p_label,stat.test$p.adj)

summary_data$is_outlier <- mapply(
  is_outlier,
  summary_data$dominant_lineage_total_mut_cnt,
  summary_data$condition,
  MoreArgs=list(data=summary_data, column="dominant_lineage_total_mut_cnt")
)

mutation_count_fig <- ggplot(
  summary_data,
  aes(x=condition, y=dominant_lineage_total_mut_cnt, fill=condition)
) +
  geom_flat_violin(
    # data=filter(summary_data, !is_outlier),
    scale="width",
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order,
    labels=condition_order
  ) +
  scale_y_continuous(
    name="Mutation count (log scale)",
    trans=pseudo_log_trans(sigma = 1, base = 10),
    breaks=c(0, 10, 100, 1000, 10000),
    limits=c(-1, 35000)
  ) +
  scale_fill_brewer(

```

```

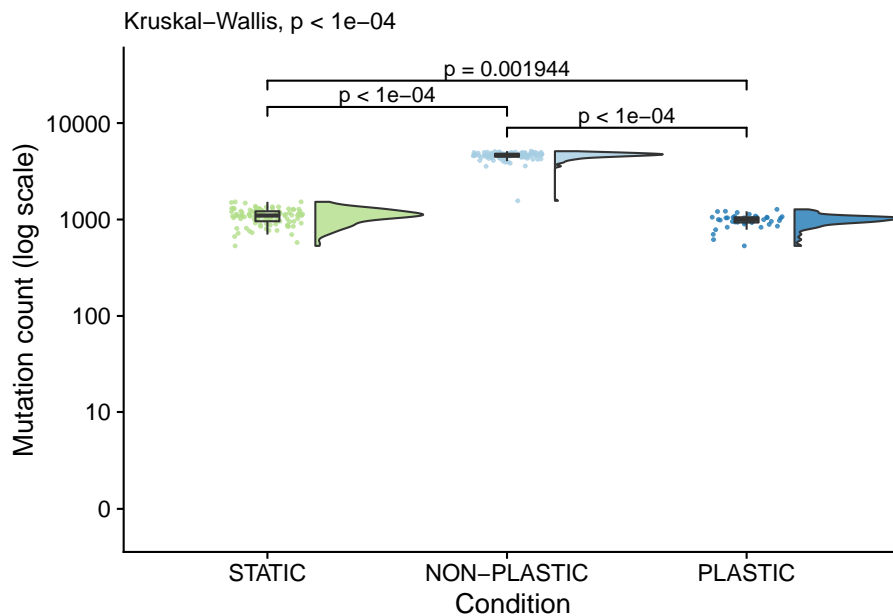
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  # coord_flip() +
  labs(
    subtitle=paste0(
      "Kruskal-Wallis, ",
      p_label(signif(kruskal.test(formula=dominant_lineage_total_mut_cnt~condition, data=summary_
    )
  ) +
  ggsignif::geom_signif(
    data=filter(stat.test, p.adj <= alpha),
    aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
    manual=TRUE,
    inherit.aes=FALSE
  ) +
  theme(
    legend.position="none"
  ) +
  ggsave(
    paste0(working_directory, "plots/", "mutation-accumulation.pdf"),
    width=5,
    height=4
  )

```

```

## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
mutation_count_fig

```



```
kruskal.test(
  formula=dominant_lineage_total_mut_cnt~condition,
  data=summary_data
)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_total_mut_cnt by condition
## Kruskal-Wallis chi-squared = 179.33, df = 2, p-value < 2.2e-16
```

```
pairwise.wilcox.test(
  x=summary_data$dominant_lineage_total_mut_cnt,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$dominant_lineage_total_mut_cnt and summary_data$condition
##
##      NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      0.0019
##
```

```
## P value adjustment method: bonferroni
```

```
paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$dominant_lineage_total_mut_cnt)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$dominant_lineage_total_mut_cnt)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$dominant_lineage_total_mut_cnt)
  )
)
```

```
## [1] "PLASTIC median: 998.5; STATIC median: 1100; NON-PLASTIC median: 4657.5"
```

```
print("Wilcox rank sum test statistics:")
```

```
## [1] "Wilcox rank sum test statistics:"
```

```
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=dominant_lineage_total_mut_cnt~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}
```

```
## [1] "STATIC<-->NON-PLASTIC: W=10000"
```

```
## [1] "STATIC<-->PLASTIC: W=1336.5"
```

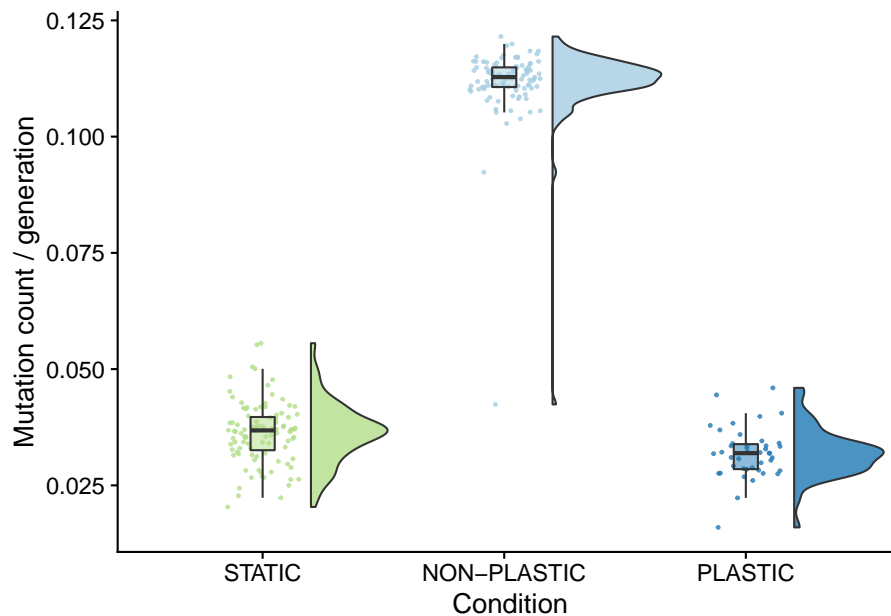
```
## [1] "PLASTIC<-->NON-PLASTIC: W=4200"
```

5.9.1 Mutation count normalized by generations elapsed

```
summary_data$mutations_per_generation <- summary_data$dominant_lineage_total_mut_cnt / summary_data$generations_elapsed

ggplot(summary_data, aes(x=condition, y=mutations_per_generation, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  )
```

```
) +  
geom_point(  
  mapping=aes(color=condition),  
  position = position_jitter(width = .15),  
  size = .5,  
  alpha = 0.8  
) +  
geom_boxplot(  
  width = .1,  
  outlier.shape = NA,  
  alpha = 0.5  
) +  
scale_x_discrete(  
  name="Condition",  
  limits=condition_order  
) +  
ylab("Mutation count / generation") +  
scale_fill_brewer(  
  palette=cb_palette  
) +  
scale_color_brewer(  
  palette=cb_palette  
) +  
# coord_flip() +  
theme(  
  legend.position="none"  
)
```

```
kruskal.test(
  formula=mutations_per_generation~condition,
  data=summary_data
)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: mutations_per_generation by condition
## Kruskal-Wallis chi-squared = 180.11, df = 2, p-value < 2.2e-16
```

```
pairwise.wilcox.test(
  x=summary_data$mutations_per_generation,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$mutations_per_generation and summary_data$condition
##
##      NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      2e-04
##
```

```
## P value adjustment method: bonferroni

paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$mutations_per_generation
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$mutations_per_generation
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$mutations_per_generation
  )
)

## [1] "PLASTIC median: 0.0319267181456982; STATIC median: 0.0368157192941933; NON-PLA"
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"

for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=mutations_per_generation~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

## [1] "STATIC<-->NON-PLASTIC: W=9987"
## [1] "STATIC<-->PLASTIC: W=1206"
## [1] "PLASTIC<-->NON-PLASTIC: W=4198"
```

5.10 Genotypic fidelity

The frequency that an offspring's genotype is the same as a parent's genotype.

```
summary_data$dominant_lineage_genotypic_fidelity <- (summary_data$dominant_generation_1
# Compute manual labels for geom_signif
stat.test <- summary_data %>%
```

```

wilcox_test(dominant_lineage_genotypic_fidelity ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="condition",step.increase=0.2)
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior in g
stat.test$manual_position <- stat.test$y.position * c(1.0,1.0,1.0)
stat.test$label <- mapply(p_label,stat.test$p.adj)

genotypic_fidelity_fig <- ggplot(
  summary_data,
  aes(x=condition, y=dominant_lineage_genotypic_fidelity, fill=condition)
) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order,
    labels=condition_order
  ) +
  scale_y_continuous(
    name="Genotypic fidelity",
    limits=c(0.85, 1.01),
    breaks=c(0.85, 0.90, 0.95, 1.0) #seq(0.85, 1.0, 0.02)
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  # coord_flip() +
  labs(

```

```

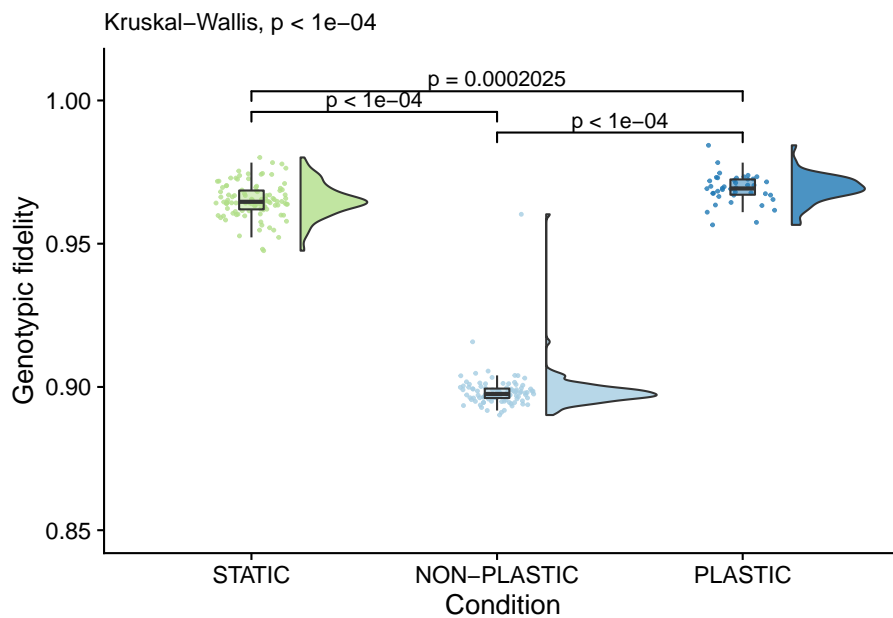
subtitle=paste0(
  "Kruskal-Wallis, ",
  p_label(signif(kruskal.test(formula=dominant_lineage_genotypic_fidelity~condition)
)
) +
ggsignif::geom_signif(
  data=filter(stat.test, p.adj <= alpha),
  aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
  manual=TRUE,
  inherit.aes=FALSE
) +
theme(
  legend.position="none"
) +
ggsave(
  paste0(working_directory, "plots/", "genotypic-fidelity.pdf"),
  width=5,
  height=4
)

```

```

## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
genotypic_fidelity_fig

```



```

kruskal.test(
  formula=dominant_lineage_genotypic_fidelity~condition,

```

```

    data=summary_data
)

##
## Kruskal-Wallis rank sum test
##
## data:  dominant_lineage_genotypic_fidelity by condition
## Kruskal-Wallis chi-squared = 179.86, df = 2, p-value < 2.2e-16
pairwise.wilcox.test(
  x=summary_data$dominant_lineage_genotypic_fidelity,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  summary_data$dominant_lineage_genotypic_fidelity and summary_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      2e-04
##
## P value adjustment method: bonferroni
paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$dominant_lineage_genotypic_fidelity)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$dominant_lineage_genotypic_fidelity)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$dominant_lineage_genotypic_fidelity)
  )
)

## [1] "PLASTIC median: 0.969286906891951; STATIC median: 0.964620594632577; NON-PLASTIC median:
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"

```

```

for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=dominant_lineage_genotypic_fidelity~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

```

```

## [1] "STATIC<-->NON-PLASTIC: W=18"
## [1] "STATIC<-->PLASTIC: W=2992"
## [1] "PLASTIC<-->NON-PLASTIC: W=2"

```

5.11 Characterizing variation along dominant lineages

5.11.1 Mutational instability

```

summary_data$frac_phenotype_changing_mut_steps <- summary_data$dominant_lineage_num_mutations /
summary_data$frac_phenotype_stable_mut_steps <- 1 - summary_data$frac_phenotype_changing_mut_steps

# Compute manual labels for geom_signif
stat.test <- summary_data %>%
  wilcox_test(frac_phenotype_changing_mut_steps ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="condition", step.increase=0.2)
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior)
stat.test$manual_position <- stat.test$y.position ## c(1.0,1.0,1.0)
stat.test$label <- mapply(p_label, stat.test$p.adj)

ggplot(summary_data, aes(x=condition, y=frac_phenotype_changing_mut_steps, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  )

```

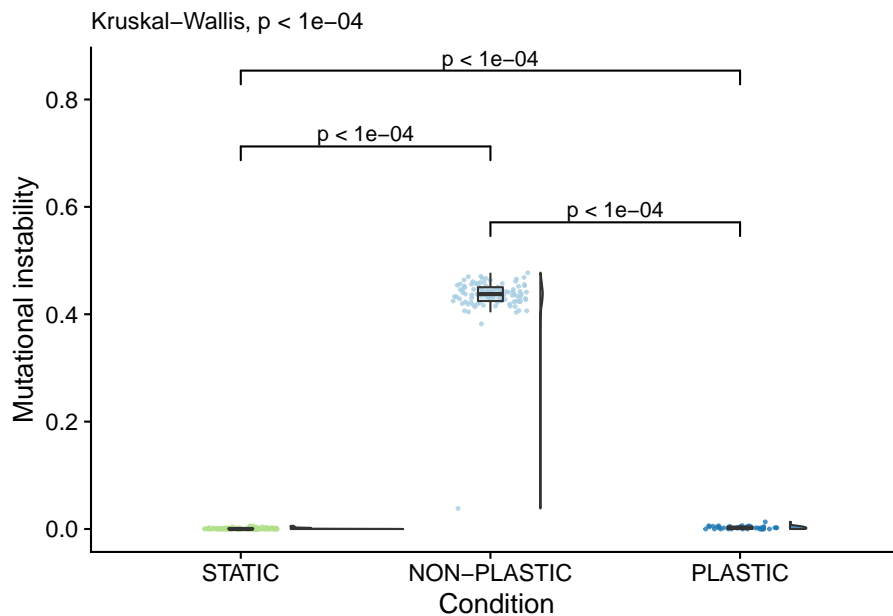
```

) +
geom_boxplot(
  width = .1,
  outlier.shape = NA,
  alpha = 0.5
) +
scale_x_discrete(
  name="Condition",
  limits=condition_order
) +
ylab("Mutational instability") +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
# coord_flip() +
labs(
  subtitle=paste0(
    "Kruskal-Wallis, ",
    p_label(signif(kruskal.test(formula=frac_phenotype_changing_mut_steps~condition, data=summa
  )
) +
ggsignif::geom_signif(
  data=filter(stat.test, p.adj <= alpha),
  aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
  manual=TRUE,
  inherit.aes=FALSE
) +
theme(
  legend.position="none"
) +
ggsave(paste0(working_directory, "plots/", "frac_phenotype_changing_mutational_steps.png"))

```

```
## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
```

```
## Saving 6.5 x 4.5 in image
```



```
kruskal.test(
  formula=frac_phenotype_changing_mut_steps~condition,
  data=summary_data
)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  frac_phenotype_changing_mut_steps by condition
## Kruskal-Wallis chi-squared = 191.23, df = 2, p-value < 2.2e-16
```

```
pairwise.wilcox.test(
  x=summary_data$frac_phenotype_changing_mut_steps,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  summary_data$frac_phenotype_changing_mut_steps and summary_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC < 2e-16      -
## STATIC  < 2e-16      2.3e-07
##
```


5.11. CHARACTERIZING VARIATION ALONG DOMINANT LINEAGES⁵⁷

```
## P value adjustment method: bonferroni
```

```
paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$frac_phenotype_changing_mut_steps)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$frac_phenotype_changing_mut_steps)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$frac_phenotype_changing_mut_steps)
  )
)
```

```
## [1] "PLASTIC median: 0.00224941742616098; STATIC median: 0; NON-PLASTIC median: 0.437583018324"
```

```
print("Wilcox rank sum test statistics:")
```

```
## [1] "Wilcox rank sum test statistics:"
```

```
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=frac_phenotype_changing_mut_steps~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}
```

```
## [1] "STATIC<-->NON-PLASTIC: W=10000"
```

```
## [1] "STATIC<-->PLASTIC: W=3172"
```

```
## [1] "PLASTIC<-->NON-PLASTIC: W=4200"
```

5.11.2 Mutational stability

```
# Compute manual labels for geom_signif
stat.test <- summary_data %>%
  wilcox_test(frac_phenotype_stable_mut_steps ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="condition", step.increase=0.75)
```

```

# Tweak y.position manually to account for scaled axis (edge case that triggers bad be
stat.test$manual_position <- stat.test$y.position ** c(1.0,1.0,1.0)
stat.test$label <- mapapply(p_label,stat.test$p.adj)

summary_data$is_outlier <- mapapply(
  is_outlier,
  summary_data$dominant_lineage_trait_volatility,
  summary_data$condition,
  MoreArgs=list(data=summary_data, column="dominant_lineage_trait_volatility")
)

mutational_stability_fig <- ggplot(
  summary_data,
  aes(x=condition, y=frac_phenotype_stable_mut_steps, fill=condition)
) +
  geom_flat_violin(
    # data=filter(summary_data,is_outlier==FALSE),
    scale="width",
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  scale_y_continuous(
    name="Mutational stability",
    limits=c(0.5, 1.15),
    breaks=c(0.5, 0.75, 1.0)
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(

```

5.11. CHARACTERIZING VARIATION ALONG DOMINANT LINEAGES59

```

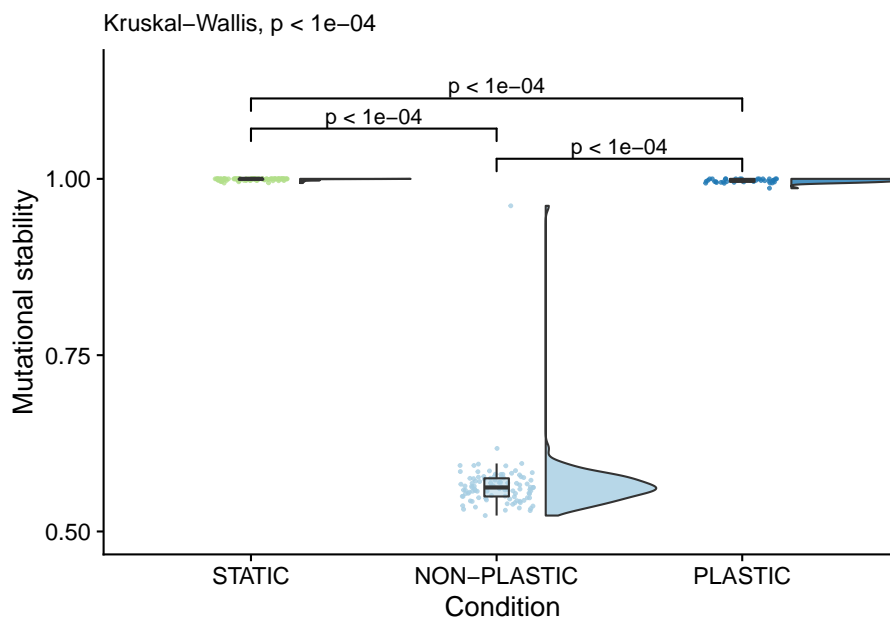
palette=cb_palette
) +
labs(
  subtitle=paste0(
    "Kruskal-Wallis, ",
    p_label(signif(kruskal.test(formula=frac_phenotype_stable_mut_steps~condition, data=summary
  )
) +
ggsignif::geom_signif(
  data=filter(stat.test, p.adj <= alpha),
  aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
  manual=TRUE,
  inherit.aes=FALSE
) +
theme(
  legend.position="none"
)

```

```

## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
mutational_stability_fig

```



```

kruskal.test(
  formula=frac_phenotype_stable_mut_steps~condition,
  data=summary_data
)

```

```

##
## Kruskal-Wallis rank sum test
##
## data: frac_phenotype_stable_mut_steps by condition
## Kruskal-Wallis chi-squared = 191.23, df = 2, p-value < 2.2e-16

pairwise.wilcox.test(
  x=summary_data$frac_phenotype_stable_mut_steps,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$frac_phenotype_stable_mut_steps and summary_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC < 2e-16      -
## STATIC  < 2e-16      2.3e-07
##
## P value adjustment method: bonferroni

paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$frac_phenotype_stable_mut_steps
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$frac_phenotype_stable_mut_steps
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$frac_phenotype_stable_mut_steps
  )
)

## [1] "PLASTIC median: 0.997750582573839; STATIC median: 1; NON-PLASTIC median: 0.5624"
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)

```

5.11. CHARACTERIZING VARIATION ALONG DOMINANT LINEAGES61

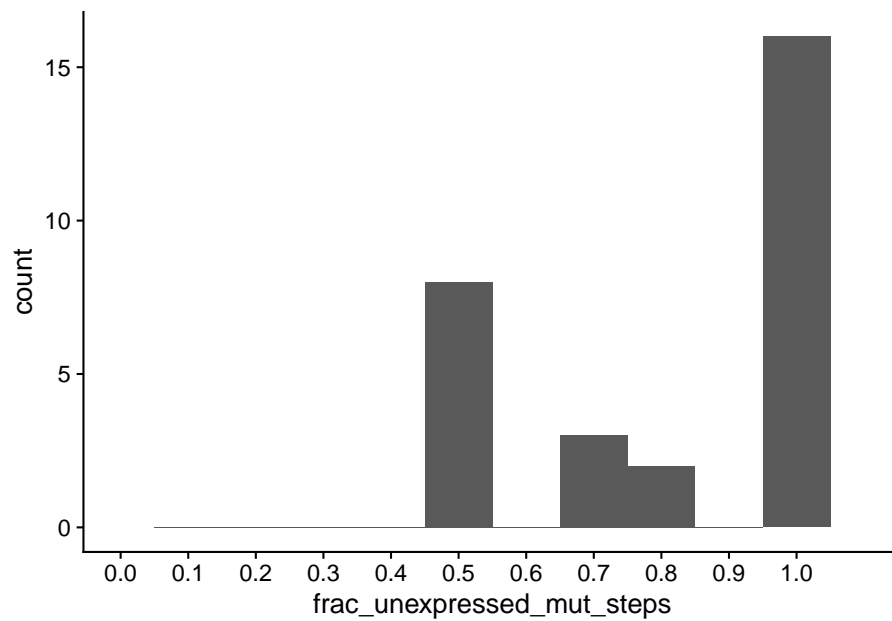
```
wt <- wilcox.test(  
  formula=frac_phenotype_stable_mut_steps~condition,  
  data=pair_data,  
  exact=FALSE,  
  paired=FALSE  
)  
print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))  
}
```

```
## [1] "STATIC<-->NON-PLASTIC: W=0"  
## [1] "STATIC<-->PLASTIC: W=1028"  
## [1] "PLASTIC<-->NON-PLASTIC: W=0"
```

5.11.3 For PLASTIC populations, what fraction of

```
summary_data$frac_unexpressed_mut_steps <- summary_data$dominant_lineage_num_mut_steps_that_change_aggr  
summary_data$frac_expressed_mut_steps <- summary_data$dominant_lineage_num_mut_steps_that_change_aggr  
  
ggplot(filter(summary_data, condition=="PLASTIC" & dominant_lineage_num_mut_steps_that_change_aggr>0))  
  geom_histogram(binwidth=0.1) +  
  scale_x_continuous(  
    limits=c(0, 1.1),  
    breaks=seq(0, 1.0, 0.1)  
  ) +  
  theme(  
    legend.position="none"  
  )
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



```
print(paste0("PLASTIC - Mean with bootstrapped 95% CI"))
```

```
## [1] "PLASTIC - Mean with bootstrapped 95% CI"
```

```
bo <- boot(filter(summary_data, condition=="PLASTIC" & dominant_lineage_num_mut_steps_1),
  statistic = samplemean, R = 10000)
print(bo)
```

```
##
```

```
## ORDINARY NONPARAMETRIC BOOTSTRAP
```

```
##
```

```
##
```

```
## Call:
```

```
## boot(data = filter(summary_data, condition == "PLASTIC" & dominant_lineage_num_mut_steps_1),
##       statistic = samplemean, R = 10000)
```

```
##
```

```
##
```

```
## Bootstrap Statistics :
```

```
##      original      bias    std. error
```

```
## t1* 0.8247126 -0.000387931 0.04027409
```

```
print(boot.ci(bo, conf=0.95, type="perc"))
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
```

```
## Based on 10000 bootstrap replicates
```

```
##
```

```
## CALL :
```

```
## boot.ci(boot.out = bo, conf = 0.95, type = "perc")
```

5.11. CHARACTERIZING VARIATION ALONG DOMINANT LINEAGES⁶³

```
##
## Intervals :
## Level      Percentile
## 95%      ( 0.7443,  0.8994 )
## Calculations and Intervals on Original Scale

plastic_summary_data <- filter(summary_data, condition=="PLASTIC")
aggregate_frac_mut_steps_that_change_unexpressed_phenotype <- sum(plastic_summary_data$dominant_lineage_num_mut_steps_that_change_unexpressed_phenotype)

## [1] 83

sum(plastic_summary_data$dominant_lineage_num_mut_steps_that_change_aggregate_phenotype)

## [1] 102

aggregate_frac_mut_steps_that_change_unexpressed_phenotype

## [1] 0.8137255

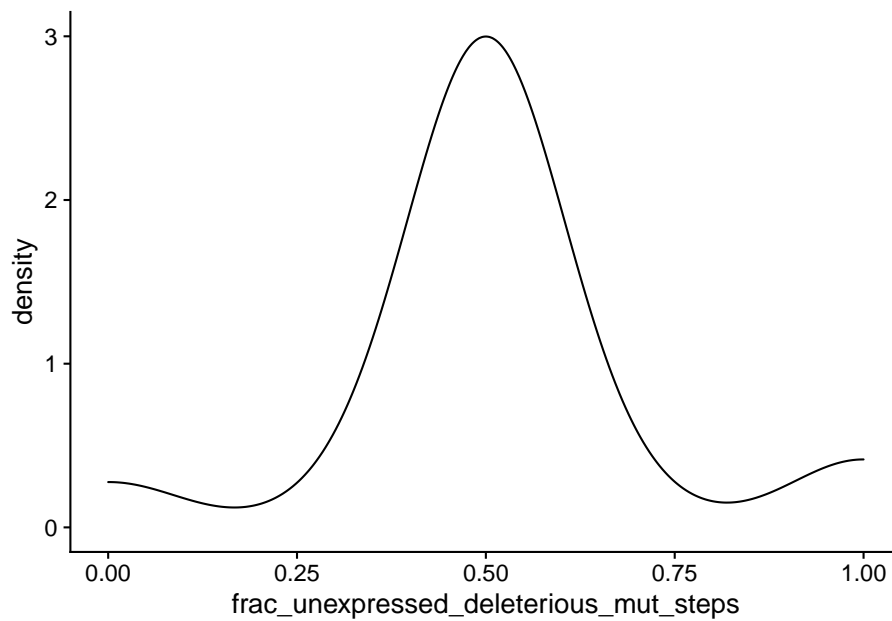
83 / 102 (0.8137255)
```

5.11.4 For PLASTIC populations, what fraction of mutations that affect the unexpressed phenotype are deleterious versus beneficial?

```
aggregate_frac_unexpressed_deleterious_mut_steps <- sum(plastic_summary_data$dominant_lineage_num_mut_steps_that_change_unexpressed_phenotype)
aggregate_frac_unexpressed_beneficial_mut_steps <- sum(plastic_summary_data$dominant_lineage_num_mut_steps_that_change_unexpressed_phenotype)
```

5.11.4.1 Deleterious mutations

```
summary_data$frac_unexpressed_deleterious_mut_steps <- summary_data$dominant_lineage_num_mut_steps_that_change_unexpressed_phenotype / summary_data$dominant_lineage_num_mut_steps_that_change_unexpressed_phenotype
ggplot(
  filter(summary_data, condition=="PLASTIC" & dominant_lineage_num_mut_steps_that_change_unexpressed_phenotype > 0)
  aes(x=frac_unexpressed_deleterious_mut_steps)
) +
  geom_density() +
  theme(
    legend.position="none"
  )
```



```
bo <- boot(filter(summary_data, condition=="PLASTIC" & dominant_lineage_num_mut_steps_1)
print(bo)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = filter(summary_data, condition == "PLASTIC" & dominant_lineage_num_mut_steps_1)
##       0)$frac_unexpressed_deleterious_mut_steps, statistic = samplemean,
##       R = 10000)
##
##
## Bootstrap Statistics :
##      original      bias    std. error
## t1*  0.5172414  0.0002813793  0.03907746
print(boot.ci(bo, conf=0.95, type="perc"))

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bo, conf = 0.95, type = "perc")
##
## Intervals :
```

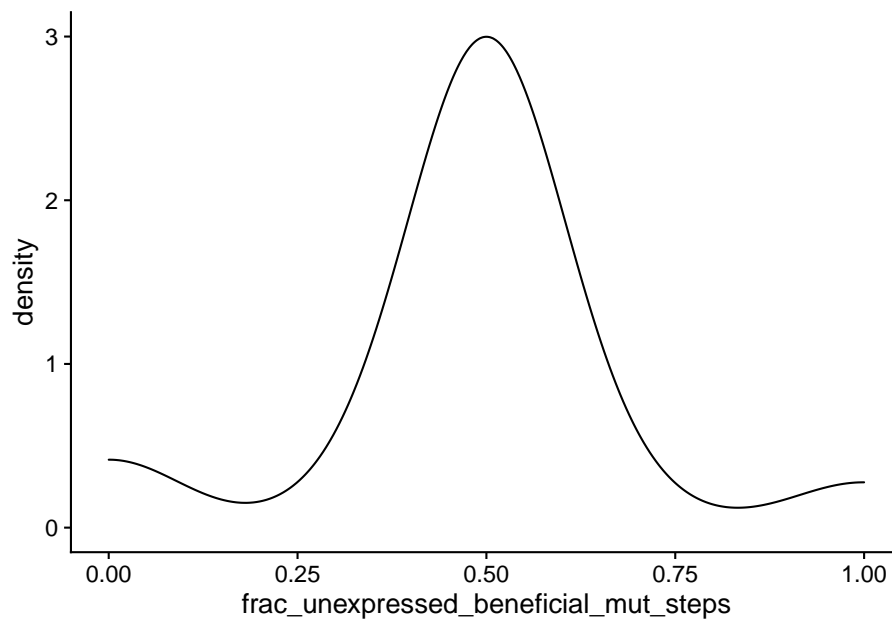

5.11. CHARACTERIZING VARIATION ALONG DOMINANT LINEAGES⁶⁵

```
## Level      Percentile
## 95%      ( 0.4402,  0.5954 )
## Calculations and Intervals on Original Scale
```

5.11.4.2 Beneficial mutations

```
summary_data$frac_unexpressed_beneficial_mut_steps <- summary_data$dominant_lineage_num_mut_steps

ggplot(
  filter(summary_data, condition=="PLASTIC" & dominant_lineage_num_mut_steps_that_change_unexpressed)
  aes(x=frac_unexpressed_beneficial_mut_steps)
) +
  geom_density() +
  theme(
    legend.position="none"
  )
```



```
bo <- boot(filter(summary_data, condition=="PLASTIC" & dominant_lineage_num_mut_steps_that_change_unexpressed))
print(bo)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
```

```
## boot(data = filter(summary_data, condition == "PLASTIC" & dominant_lineage_num_mut_
##      0)$frac_unexpressed_beneficial_mut_steps, statistic = samplemean,
##      R = 10000)
##
##
## Bootstrap Statistics :
##      original      bias    std. error
## t1* 0.4827586 -0.0002189655  0.03951968
print(boot.ci(bo, conf=0.95, type="perc"))

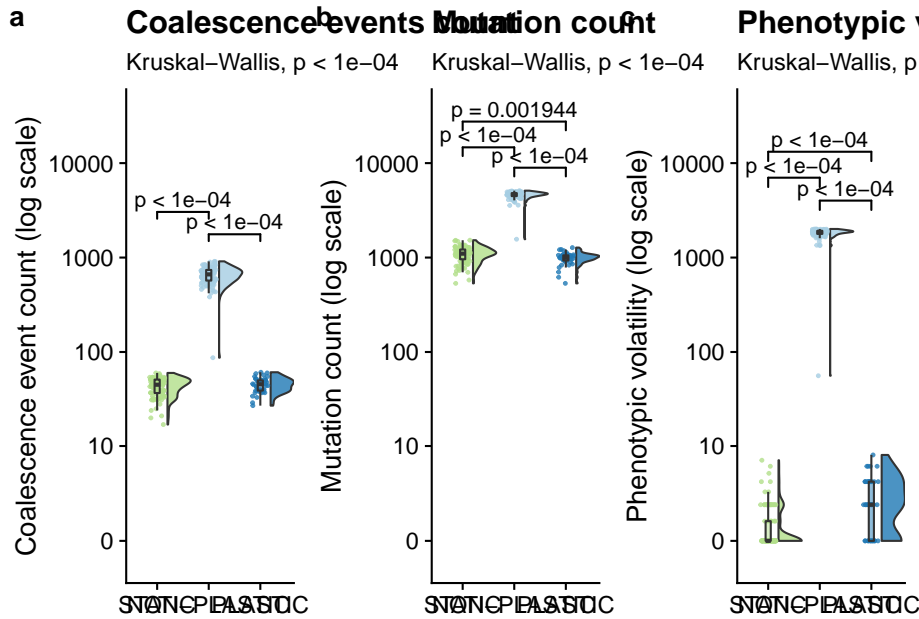
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bo, conf = 0.95, type = "perc")
##
## Intervals :
## Level      Percentile
## 95%      ( 0.4046,  0.5598 )
## Calculations and Intervals on Original Scale
```

5.12 Manuscript figures

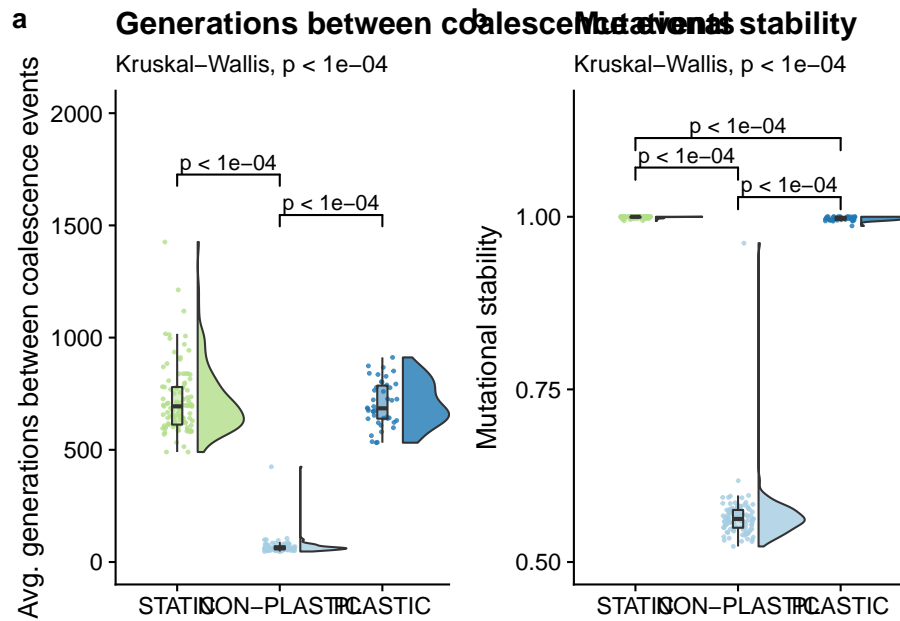
Figures styled for the paper.

```
magnitude_grid <- plot_grid(
  coalescence_events_fig +
    theme(
      legend.position="none",
      axis.title.x=element_blank()
    ) +
    ggtitle("Coalescence events count"),
  mutation_count_fig +
    theme(
      legend.position="none",
      axis.title.x=element_blank()
    ) +
    ggtitle("Mutation count"),
  phenotypic_volatility_fig +
    theme(
      legend.position="none",
      axis.title.x=element_blank()
    ) +
    ggtitle("Phenotypic volatility"),
  nrow=1,
```

```
ncol=3,
align="v",
labels="auto"
)
magnitude_grid
```



```
pace_grid <- plot_grid(
  coalescence_events_freq_fig +
    theme(
      legend.position="none",
      axis.title.x=element_blank()
    ) +
    ggtitle("Generations between coalescence events"),
  mutational_stability_fig +
    theme(
      legend.position="none",
      axis.title.x=element_blank()
    ) +
    ggtitle("Mutational stability"),
  nrow=1,
  ncol=2,
  align="v",
  labels="auto"
)
pace_grid
```



```
save_plot(
  paste0(working_directory, "plots/", "evolutionary-change-magnitude-panel.pdf"),
  magnitude_grid,
  base_height=6,
  base_asp=3/1
)

save_plot(
  paste0(working_directory, "plots/", "evolutionary-change-pace-panel.pdf"),
  pace_grid,
  base_height=6,
  base_asp=2/1
)

# save_plot(
#   paste0(working_directory, "plots/", "evolutionary-change-full-panel.pdf"),
#   grid,
#   base_height=12,
#   base_asp=3/2
# )
```

Chapter 6

Evolution and maintenance of novel traits

The effect of adaptive phenotypic plasticity on the evolution and maintenance of novel tasks.

6.1 Overview

```
total_updates <- 200000
replicates <- 100
alpha <- 0.05

focal_traits <- c("not", "nand", "and", "ornot", "or", "andnot")
traits_set_a <- c("not", "and", "or")
traits_set_b <- c("nand", "ornot", "andnot")
extra_traits <- c(
  "nor", "xor", "equals",
  "logic_3aa", "logic_3ab", "logic_3ac",
  "logic_3ad", "logic_3ae", "logic_3af",
  "logic_3ag", "logic_3ah", "logic_3ai",
  "logic_3aj", "logic_3ak", "logic_3al",
  "logic_3am", "logic_3an", "logic_3ao",
  "logic_3ap", "logic_3aq", "logic_3ar",
  "logic_3as", "logic_3at", "logic_3au",
  "logic_3av", "logic_3aw", "logic_3ax",
  "logic_3ay", "logic_3az", "logic_3ba",
  "logic_3bb", "logic_3bc", "logic_3bd",
  "logic_3be", "logic_3bf", "logic_3bg",
  "logic_3bh", "logic_3bi", "logic_3bj",
```

```

    "logic_3bk", "logic_3bl", "logic_3bm",
    "logic_3bn", "logic_3bo", "logic_3bp",
    "logic_3bq", "logic_3br", "logic_3bs",
    "logic_3bt", "logic_3bu", "logic_3bv",
    "logic_3bw", "logic_3bx", "logic_3by",
    "logic_3bz", "logic_3ca", "logic_3cb",
    "logic_3cc", "logic_3cd", "logic_3ce",
    "logic_3cf", "logic_3cg", "logic_3ch",
    "logic_3ci", "logic_3cj", "logic_3ck",
    "logic_3cl", "logic_3cm", "logic_3cn",
    "logic_3co", "logic_3cp"
  )

# Relative location of data.
working_directory <- "experiments/2021-01-31-complex-features/analysis/" # << For book
# working_directory <- "./"

```

6.2 Analysis dependencies

Load all required R libraries.

```

library(ggplot2)
library(rstatix)
library(ggsignif)
library(scales)
library(tidyverse)
library(cowplot)
library(RColorBrewer)
library(Hmisc)
library(boot)
source("https://gist.githubusercontent.com/benmarwick/2a1bb0133ff568cbe28d/raw/fb53bd9")

```

These analyses were conducted/knitted with the following computing environment:

```

print(version)

##
## platform      x86_64-pc-linux-gnu
## arch          x86_64
## os            linux-gnu
## system        x86_64, linux-gnu
## status
## major         4
## minor         1.0
## year          2021

```

```
## month          05
## day            18
## svn rev        80317
## language       R
## version.string R version 4.1.0 (2021-05-18)
## nickname       Camp Pontanezen
```

6.3 Setup

```
##### summary data #####
summary_data_loc <- paste0(working_directory, "data/aggregate.csv")
summary_data <- read.csv(summary_data_loc, na.strings="NONE")

summary_data$DISABLE_REACTION_SENSORS <- as.factor(summary_data$DISABLE_REACTION_SENSORS)
summary_data$chg_env <- summary_data$chg_env == "True"
summary_data$dominant_plastic_odd_even <- as.factor(summary_data$dominant_plastic_odd_even)
summary_data$sensors <- summary_data$DISABLE_REACTION_SENSORS == "0"
summary_data$is_plastic <- summary_data$dominant_plastic_odd_even == "True"
summary_data$extra_task_value <- as.factor(summary_data$extra_task_value)
summary_data <- filter(summary_data, extra_task_value == 0.1)

env_label_fun <- function(chg_env) {
  if (chg_env) {
    return("Fluctuating")
  } else {
    return("Constant")
  }
}

sensors_label_fun <- function(has_sensors) {
  if (has_sensors) {
    return("Sensors")
  } else {
    return("No sensors")
  }
}

condition_label_fun <- function(has_sensors, env_chg) {
  if (has_sensors && env_chg) {
    return("PLASTIC")
  } else if (env_chg) {
    return("NON-PLASTIC")
  } else {
    return("STATIC")
  }
}
```

```

    }
  }

summary_data$env_label <- mapply(
  env_label_fun,
  summary_data$chg_env
)
summary_data$sensors_label <- mapply(
  sensors_label_fun,
  summary_data$sensors
)
summary_data$condition <- mapply(
  condition_label_fun,
  summary_data$sensors,
  summary_data$chg_env
)

condition_order = c(
  "STATIC",
  "NON-PLASTIC",
  "PLASTIC"
)

pairwise_comparisons <- list(
  c("STATIC", "NON-PLASTIC"),
  c("STATIC", "PLASTIC"),
  c("PLASTIC", "NON-PLASTIC")
)

p_label <- function(p_value) {
  threshold = 0.0001
  if (p_value < threshold) {
    return(paste0("p < ", threshold))
  } else {
    return(paste0("p = ", p_value))
  }
}

# *really* inefficient way to identify outliers
is_outlier <- function(value, cond, data, column) {
  cond_data <- filter(data, condition==cond)
  q1 <- summary(cond_data[,column])[["1st Qu."]]
  q3 <- summary(cond_data[,column])[["3rd Qu."]]
  H <- 1.5 * IQR(cond_data[,column])
  return( (value < (q1-H)) || (value > (q3+H)) )
}

```



```
##### time series #####
lineage_time_series_data_loc <- paste0(working_directory, "data/lineage_series.csv")
lineage_time_series_data <- read.csv(lineage_time_series_data_loc)

lineage_time_series_data$DISABLE_REACTION_SENSORS <- as.factor(lineage_time_series_data$DISABLE_P
lineage_time_series_data$chg_env <- lineage_time_series_data$chg_env == "True"
lineage_time_series_data$sensors <- lineage_time_series_data$DISABLE_REACTION_SENSORS == "0"
lineage_time_series_data$extra_task_value <- as.factor(lineage_time_series_data$extra_task_value)

lineage_time_series_data$env_label <- mapply(
  env_label_fun,
  lineage_time_series_data$chg_env
)
lineage_time_series_data$sensors_label <- mapply(
  sensors_label_fun,
  lineage_time_series_data$sensors
)
lineage_time_series_data$condition <- mapply(
  condition_label_fun,
  lineage_time_series_data$sensors,
  lineage_time_series_data$chg_env
)

##### misc #####
# Configure our default graphing theme
theme_set(theme_cowplot())
# Palette
cb_palette <- "Paired"
# Create directory to dump plots
dir.create(paste0(working_directory, "plots"), showWarnings=FALSE)
# Sample mean function
samplemean <- function(x, d) {
  return(mean(x[d]))
}
```

6.4 The evolution of phenotypic plasticity

For sensor-enabled populations in fluctuating environments, we only transferred populations containing an optimally plastic genotype to phase two.

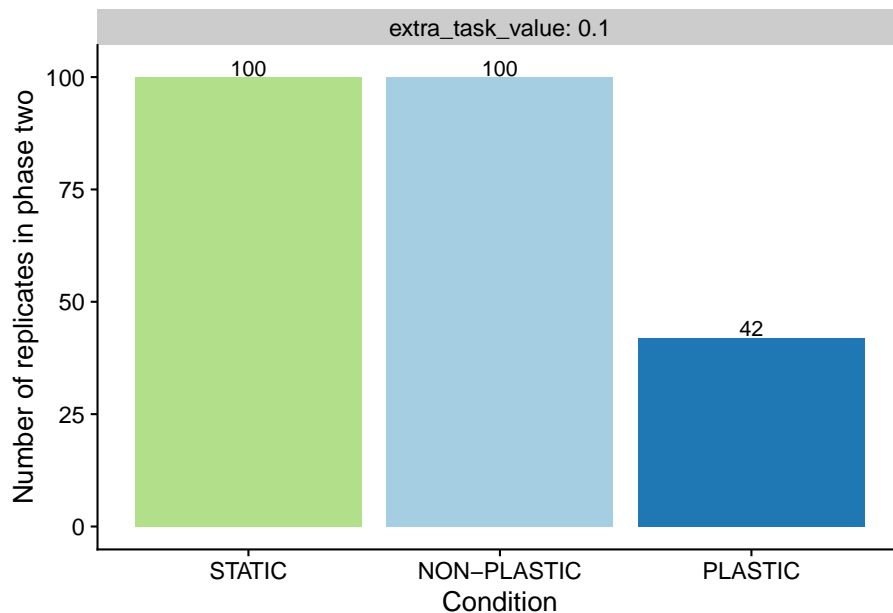
```
summary_data_grouped = dplyr::group_by(summary_data, sensors, env_label, condition, extra_task_value)
summary_data_group_counts = dplyr::summarize(summary_data_grouped, n=dplyr::n())

ggplot(summary_data_group_counts, aes(x=condition, y=n, fill=condition)) +
```

```

geom_col(position=position_dodge(0.9)) +
geom_text(aes(label=n, y=n+2)) +
scale_x_discrete(
  name="Condition",
  limits=condition_order
) +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
ylab("Number of replicates in phase two") +
facet_wrap(~extra_task_value, labeller=label_both) +
theme(
  legend.position="none"
)

```



We can confirm our expectation that the dominant genotypes in non-plastic conditions are not phenotypically plastic.

```

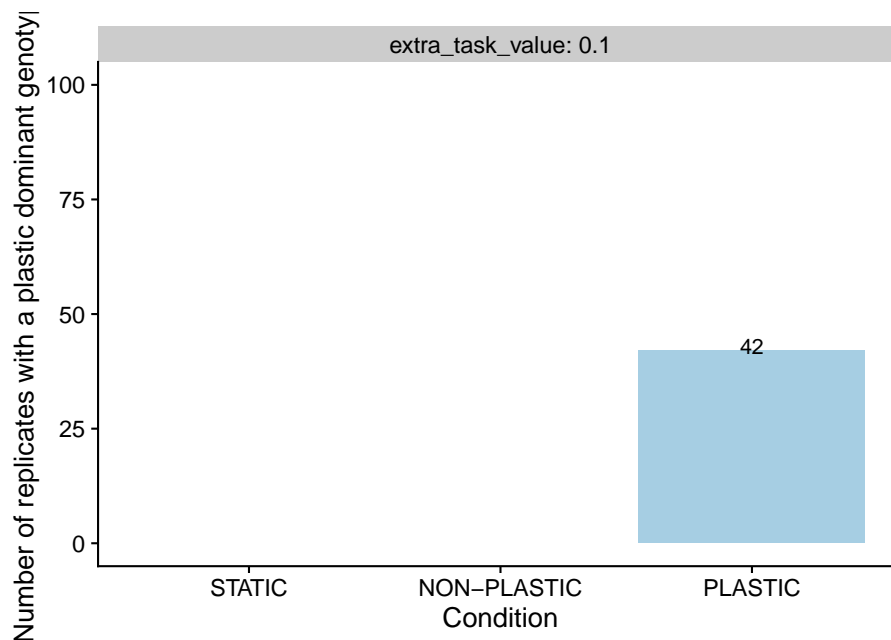
summary_data_grouped = dplyr::group_by(summary_data, condition, is_plastic, extra_task_value)
summary_data_group_counts = dplyr::summarize(summary_data_grouped, n=dplyr::n())
ggplot(filter(summary_data_group_counts, is_plastic), aes(x=condition, y=n, fill=condition)) +
  geom_col(position=position_dodge(0.9)) +

```

```

scale_x_discrete(
  name="Condition",
  limits=condition_order
) +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
ylim(0, 100) +
geom_text(aes(label=n, y=n+1)) +
ylab("Number of replicates with a plastic dominant genotype") +
facet_wrap(~extra_task_value, labeller=label_both) +
theme(
  legend.position="none"
)

```



6.5 Final novel task count (dominant genotype)

How many novel tasks do final dominant genotypes perform?

```

# Compute manual labels for geom_signif
stat.test <- summary_data %>%

```

```

wilcox_test(dominant_extra_tasks ~ condition) %>%
adjust_pvalue(method = "bonferroni") %>%
add_significance() %>%
add_xy_position(x="condition") # ,step.increase=1
# Tweak y.position manually to account for scaled axis (edge case that triggers bad be
stat.test$manual_position <- stat.test$y.position ## c(1.0,1.0,1.03)
stat.test$label <- mapapply(p_label,stat.test$p.adj)

summary_data$is_outlier <- mapapply(
  is_outlier,
  summary_data$dominant_extra_tasks,
  summary_data$condition,
  MoreArgs=list(data=summary_data, column="dominant_extra_tasks")
)

final_novel_task_count_fig <- ggplot(
  summary_data,
  aes(x=condition, y=dominant_extra_tasks, fill=condition)
) +
geom_flat_violin(
  # data=filter(summary_data,is_outlier==FALSE),
  scale="width",
  position = position_nudge(x = .2, y = 0),
  alpha = .8
) +
geom_point(
  mapping=aes(color=condition),
  position = position_jitter(width = .15),
  size = .5,
  alpha = 0.8
) +
geom_boxplot(
  width = .1,
  outlier.shape = NA,
  alpha = 0.5
) +
scale_x_discrete(
  name="Condition",
  limits=condition_order,
  labels=condition_order
) +
scale_y_continuous(
  name="Final novel task count"
) +
scale_fill_brewer(

```

```

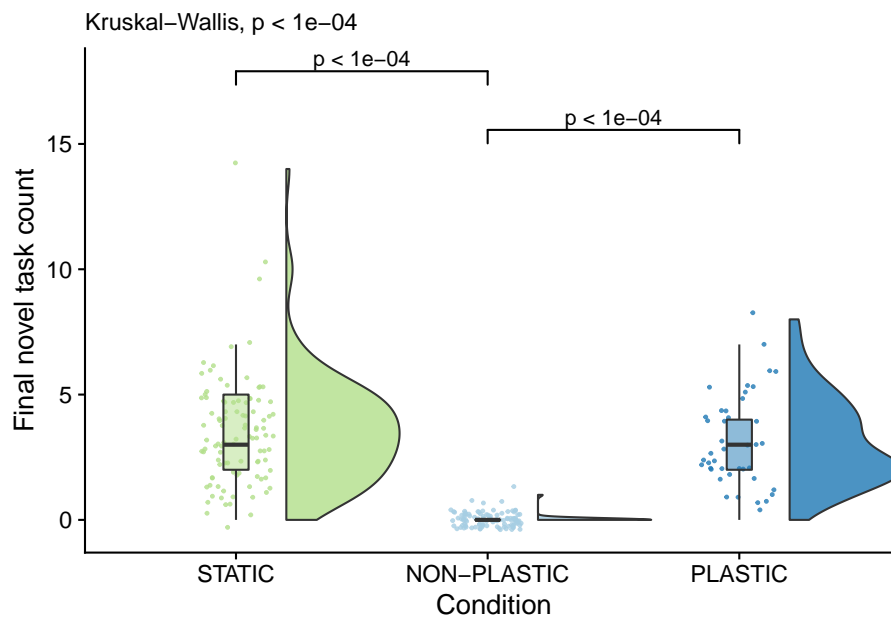
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  labs(
    subtitle=paste0(
      "Kruskal-Wallis, ",
      p_label(signif(kruskal.test(formula=dominant_extra_tasks~condition, data=summary_data)$p.val
    )
  ) +
  ggsignif::geom_signif(
    data=filter(stat.test, p.adj <= alpha),
    aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
    manual=TRUE,
    inherit.aes=FALSE
  ) +
  # coord_flip()
  theme(
    legend.position="none"
  )

```

```

## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
final_novel_task_count_fig

```



```

kruskal.test(
  formula=dominant_extra_tasks~condition,
  data=summary_data
)

##
## Kruskal-Wallis rank sum test
##
## data: dominant_extra_tasks by condition
## Kruskal-Wallis chi-squared = 177.17, df = 2, p-value < 2.2e-16

pairwise.wilcox.test(
  x=summary_data$dominant_extra_tasks,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
  conf.int=TRUE,
  conf.level=0.95
)

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$dominant_extra_tasks and summary_data$condition
##
##      NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      0.9
##
## P value adjustment method: bonferroni

paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$dominant_extra_tasks)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$dominant_extra_tasks)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$dominant_extra_tasks)
  )
)

## [1] "PLASTIC median: 3; STATIC median: 3; NON-PLASTIC median: 0"

```

```
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=dominant_extra_tasks~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

## [1] "STATIC<-->NON-PLASTIC: W=184"
## [1] "STATIC<-->PLASTIC: W=1871"
## [1] "PLASTIC<-->NON-PLASTIC: W=64"
```

6.6 Novel task count (final population)

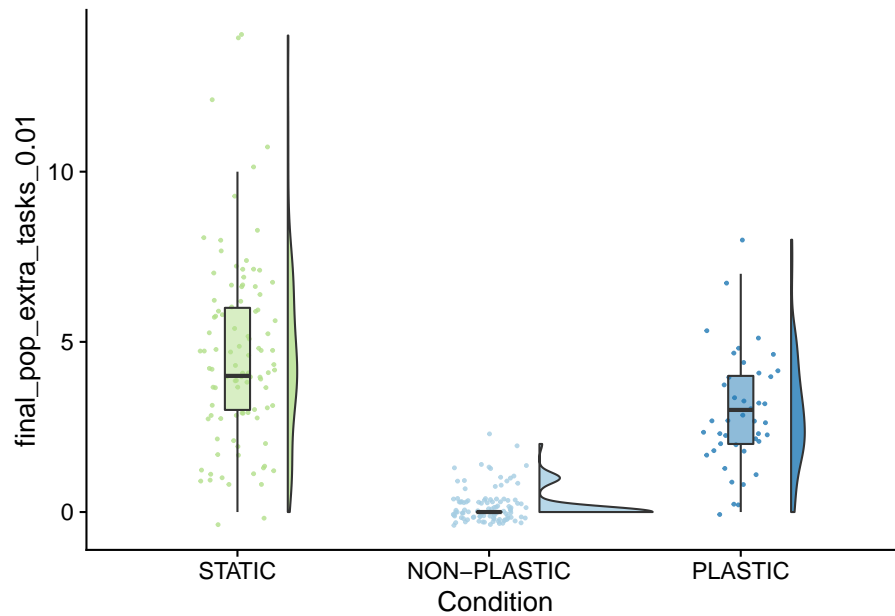
How many novel tasks are performed across the final population (1% of organisms must perform to count)?

```
ggplot(summary_data, aes(x=condition, y=final_pop_extra_tasks_0.01, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  scale_fill_brewer(
```

```

    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  # coord_flip() +
  theme(
    legend.position="none"
  )

```



```

kruskal.test(
  formula=final_pop_extra_tasks_0.01~condition,
  data=summary_data
)

```

```

##
##  Kruskal-Wallis rank sum test
##
## data:  final_pop_extra_tasks_0.01 by condition
## Kruskal-Wallis chi-squared = 169.47, df = 2, p-value < 2.2e-16

```

```

pairwise.wilcox.test(
  x=summary_data$final_pop_extra_tasks_0.01,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
  conf.int=TRUE,

```



```

    conf.level=0.95
  )

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$final_pop_extra_tasks_0.01 and summary_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC < 2e-16      -
## STATIC  < 2e-16    0.00016
##
## P value adjustment method: bonferroni
paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$final_pop_extra_tasks_0.01)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$final_pop_extra_tasks_0.01)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$final_pop_extra_tasks_0.01)
  )
)

## [1] "PLASTIC median: 3; STATIC median: 4; NON-PLASTIC median: 0"
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=final_pop_extra_tasks_0.01~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

```

```
## [1] "STATIC<-->NON-PLASTIC: W=227.5"
## [1] "STATIC<-->PLASTIC: W=1203"
## [1] "PLASTIC<-->NON-PLASTIC: W=225.5"
```

6.7 Novel task discovery (lineage)

```
# Compute manual labels for geom_signif
stat.test <- summary_data %>%
  wilcox_test(dominant_lineage_extra_traits_discovered ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="condition") # ,step.increase=1
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior)
stat.test$manual_position <- stat.test$y.position ## c(1.0,1.0,1.03)
stat.test$label <- mapply(p_label,stat.test$p.adj)

summary_data$is_outlier <- mapply(
  is_outlier,
  summary_data$dominant_lineage_extra_traits_discovered,
  summary_data$condition,
  MoreArgs=list(data=summary_data, column="dominant_lineage_extra_traits_discovered")
)

lineage_novel_task_discovery_fig <- ggplot(
  summary_data,
  aes(x=condition, y=dominant_lineage_extra_traits_discovered, fill=condition)
) +
  geom_flat_violin(
    # data=filter(summary_data,is_outlier==FALSE),
    scale="width",
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
```

```

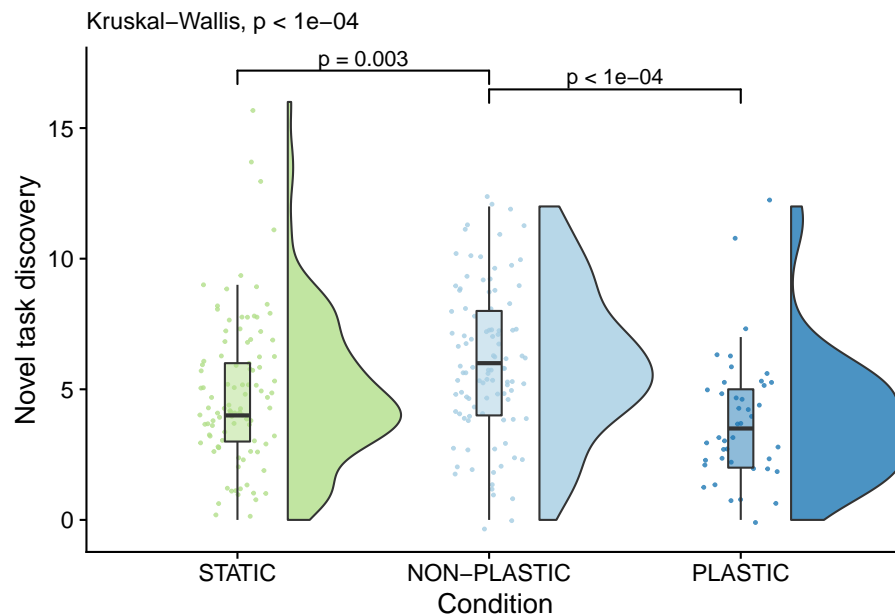
scale_x_discrete(
  name="Condition",
  limits=condition_order,
  labels=condition_order
) +
scale_y_continuous(
  name="Novel task discovery"
) +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
labs(
  subtitle=paste0(
    "Kruskal-Wallis, ",
    p_label(signif(kruskal.test(formula=dominant_lineage_extra_traits_discovered~condition, dat
  )
) +
ggsignif::geom_signif(
  data=filter(stat.test, p.adj <= alpha),
  aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
  manual=TRUE,
  inherit.aes=FALSE
) +
# coord_flip()
theme(
  legend.position="none"
)

```

```

## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
lineage_novel_task_discovery_fig

```



```
kruskal.test(
  formula=dominant_lineage_extra_traits_discovered~condition,
  data=summary_data
)

##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_extra_traits_discovered by condition
## Kruskal-Wallis chi-squared = 24.099, df = 2, p-value = 5.846e-06

pairwise.wilcox.test(
  x=summary_data$dominant_lineage_extra_traits_discovered,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
  conf.int=TRUE,
  conf.level=0.95
)

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$dominant_lineage_extra_traits_discovered and summary_data$condi
##
##          NON-PLASTIC PLASTIC
## PLASTIC 1.7e-05      -
```

```
## STATIC 0.0035      0.0561
##
## P value adjustment method: bonferroni

paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$dominant_lineage_extra_traits_discovered)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$dominant_lineage_extra_traits_discovered)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$dominant_lineage_extra_traits_discovered)
  )
)

## [1] "PLASTIC median: 3.5; STATIC median: 4; NON-PLASTIC median: 6"
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"

for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=dominant_lineage_extra_traits_discovered~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

## [1] "STATIC<-->NON-PLASTIC: W=6319.5"
## [1] "STATIC<-->PLASTIC: W=1578"
## [1] "PLASTIC<-->NON-PLASTIC: W=3110.5"
```

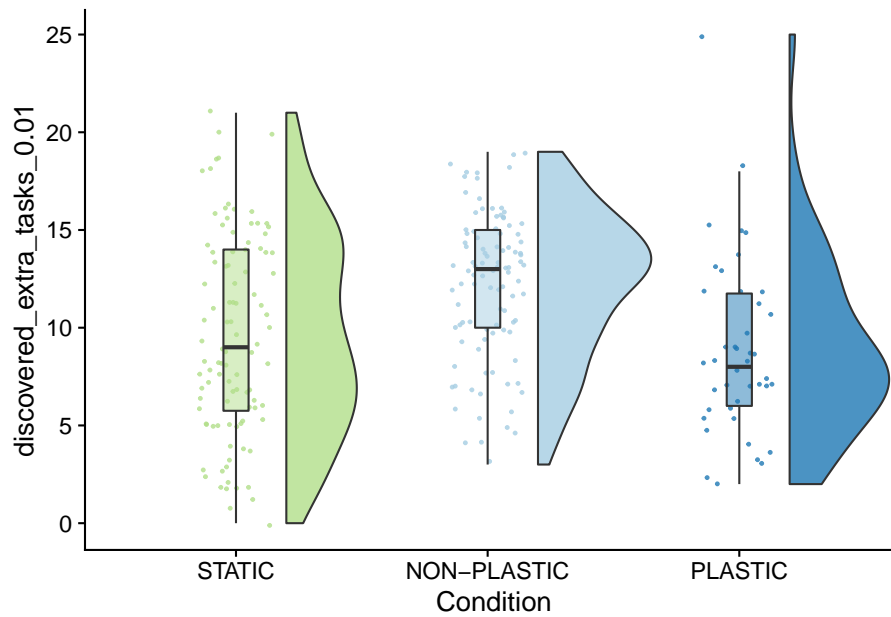
6.8 Novel task discovery (population)

```
ggplot(
  summary_data,
  aes(x=condition, y=discovered_extra_tasks_0.01, fill=condition)
```

```

) +
geom_flat_violin(
  position = position_nudge(x = .2, y = 0),
  alpha = .8
) +
geom_point(
  mapping=aes(color=condition),
  position = position_jitter(width = .15),
  size = .5,
  alpha = 0.8
) +
geom_boxplot(
  width = .1,
  outlier.shape = NA,
  alpha = 0.5
) +
scale_x_discrete(
  name="Condition",
  limits=condition_order
) +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
# coord_flip() +
theme(
  legend.position="none"
)

```



```
kruskal.test(
  formula=discovered_extra_tasks_0.01~condition,
  data=summary_data
)

##
##  Kruskal-Wallis rank sum test
##
## data:  discovered_extra_tasks_0.01 by condition
## Kruskal-Wallis chi-squared = 24.271, df = 2, p-value = 5.365e-06

pairwise.wilcox.test(
  x=summary_data$discovered_extra_tasks_0.01,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
  conf.int=TRUE,
  conf.level=0.95
)

##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  summary_data$discovered_extra_tasks_0.01 and summary_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC 2.4e-05      -
```

```
## STATIC 0.00035      1.00000
##
## P value adjustment method: bonferroni

paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$discovered_extra_tasks_0.01)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$discovered_extra_tasks_0.01)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$discovered_extra_tasks_0.01)
  )
)

## [1] "PLASTIC median: 8; STATIC median: 9; NON-PLASTIC median: 13"
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"

for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=discovered_extra_tasks_0.01~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

## [1] "STATIC<-->NON-PLASTIC: W=6573.5"
## [1] "STATIC<-->PLASTIC: W=1918.5"
## [1] "PLASTIC<-->NON-PLASTIC: W=3096"
```

6.9 Novel task discovery frequency (lineage)

```
summary_data$dominant_lineage_extra_traits_discovered_per_generation <- summary_data$d
summary_data$dominant_lineage_extra_traits_generations_per_discovery <- summary_data$d
```



```

# Compute manual labels for geom_signif
# stat.test <- filter(summary_data, dominant_lineage_extra_traits_discovered > 0) %>%
#   wilcox_test(dominant_lineage_extra_traits_generations_per_discovery ~ condition) %>%
#   adjust_pvalue(method = "bonferroni") %>%
#   add_significance() %>%
#   add_xy_position(x="condition") # ,step.increase=1
stat.test <- summary_data %>%
  wilcox_test(dominant_lineage_extra_traits_discovered_per_generation ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="condition", step.increase=0.0001) # ,step.increase=1
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior in g
stat.test$manual_position <- stat.test$y.position #* c(1.0,1.0,1.03)
stat.test$label <- mapply(p_label,stat.test$p.adj)

summary_data$is_outlier <- mapply(
  is_outlier,
  summary_data$dominant_lineage_extra_traits_discovered_per_generation,
  summary_data$condition,
  MoreArgs=list(data=summary_data, column="dominant_lineage_extra_traits_discovered_per_generation")
)

lineage_novel_task_discovery_freq_fig <- ggplot(
  # filter(summary_data, dominant_lineage_extra_traits_discovered > 0),
  summary_data,
  aes(x=condition, y=dominant_lineage_extra_traits_discovered_per_generation, fill=condition)
) +
  geom_flat_violin(
    # data=filter(summary_data,is_outlier==FALSE),
    scale="width",
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(

```

```

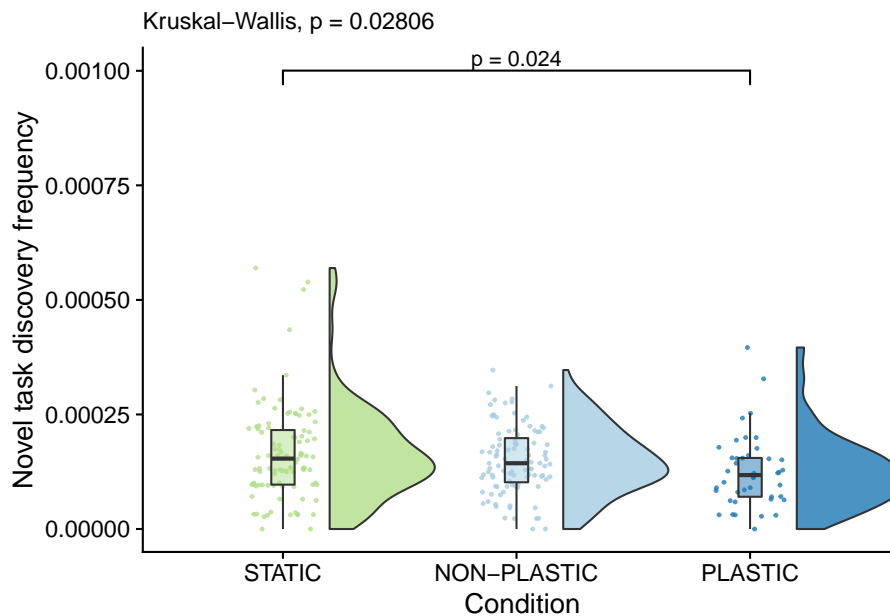
    name="Condition",
    limits=condition_order
  ) +
  ylab("Novel task discovery frequency") +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  labs(
    subtitle=paste0(
      "Kruskal-Wallis, ",
      p_label(signif(kruskal.test(formula=dominant_lineage_extra_traits_discovered_per.
    )
  ) +
  ggsignif::geom_signif(
    data=filter(stat.test, p.adj <= alpha),
    aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
    manual=TRUE,
    inherit.aes=FALSE
  ) +
  # coord_flip() +
  theme(
    legend.position="none"
  )

```

```

## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
lineage_novel_task_discovery_freq_fig

```



```
kruskal.test(
  formula=dominant_lineage_extra_traits_discovered_per_generation~condition,
  data=summary_data
)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_extra_traits_discovered_per_generation by condition
## Kruskal-Wallis chi-squared = 7.1465, df = 2, p-value = 0.02806
```

```
pairwise.wilcox.test(
  x=summary_data$dominant_lineage_extra_traits_discovered_per_generation,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
  conf.int=TRUE,
  conf.level=0.95
)
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$dominant_lineage_extra_traits_discovered_per_generation and summary_data$
##
##      NON-PLASTIC PLASTIC
## PLASTIC 0.092      -
```

```
## STATIC 1.000 0.025
##
## P value adjustment method: bonferroni

paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$dominant_lineage_extra_traits_discovered_per_generation
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$dominant_lineage_extra_traits_discovered_per_generation
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$dominant_lineage_extra_traits_discovered_per_generation
  )
)

## [1] "PLASTIC median: 0.000117695011124939; STATIC median: 0.00015363220504867; NON-PLASTIC median: 0.000117695011124939"
print("Wilcox rank sum test statistics:")

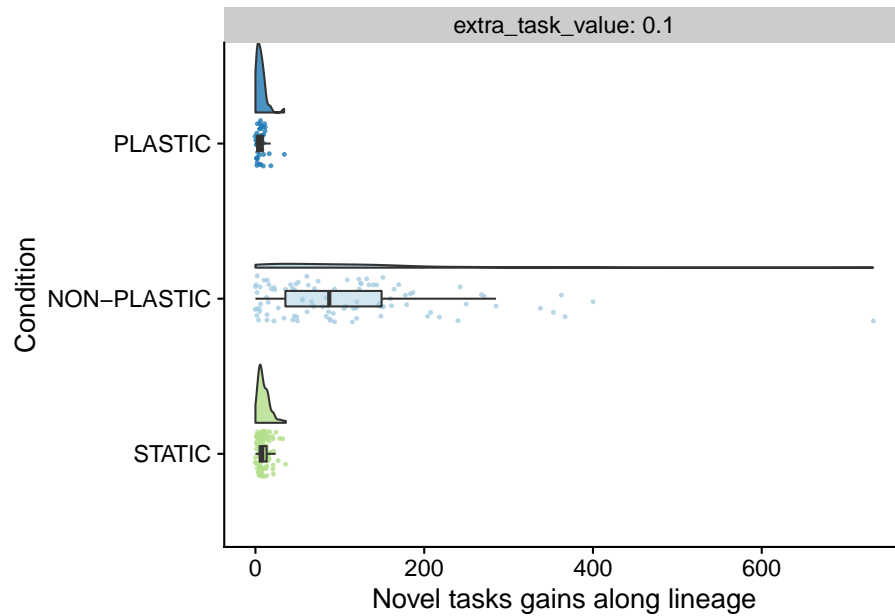
## [1] "Wilcox rank sum test statistics:"
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=dominant_lineage_extra_traits_discovered_per_generation~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

## [1] "STATIC<-->NON-PLASTIC: W=4751"
## [1] "STATIC<-->PLASTIC: W=1510.5"
## [1] "PLASTIC<-->NON-PLASTIC: W=2584"
```

6.10 Novel tasks gained (lineage)

```
ggplot(
  summary_data,
  aes(x=condition, y=dominant_lineage_extra_traits_gained, fill=condition)
```

```
) +  
geom_flat_violin(  
  position = position_nudge(x = .2, y = 0),  
  alpha = .8  
) +  
geom_point(  
  mapping=aes(color=condition),  
  position = position_jitter(width = .15),  
  size = .5,  
  alpha = 0.8  
) +  
geom_boxplot(  
  width = .1,  
  outlier.shape = NA,  
  alpha = 0.5  
) +  
scale_x_discrete(  
  name="Condition",  
  limits=condition_order  
) +  
ylab("Novel tasks gains along lineage") +  
scale_fill_brewer(  
  palette=cb_palette  
) +  
scale_color_brewer(  
  palette=cb_palette  
) +  
coord_flip() +  
facet_wrap(  
  ~extra_task_value,  
  labeller=label_both  
) +  
theme(  
  legend.position="none"  
) +  
ggsave(  
  paste0(working_directory, "plots/dominant-lineage-extra-tasks-gained.pdf"),  
  width=15,  
  height=10  
)
```



6.11 Novel task loss (lineage)

```
# Compute manual labels for geom_signif
stat.test <- summary_data %>%
  wilcox_test(dominant_lineage_extra_traits_lost ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="condition", step.increase=1)
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior)
stat.test$manual_position <- log10(stat.test$y.position) * c(1.0,1.0,1.03)
stat.test$label <- mapply(p_label,stat.test$p.adj)

summary_data$is_outlier <- mapply(
  is_outlier,
  summary_data$dominant_lineage_extra_traits_lost,
  summary_data$condition,
  MoreArgs=list(data=summary_data, column="dominant_lineage_extra_traits_lost")
)

lineage_novel_task_loss_fig <- ggplot(
  summary_data,
  aes(x=condition, y=dominant_lineage_extra_traits_lost, fill=condition)
) +
```

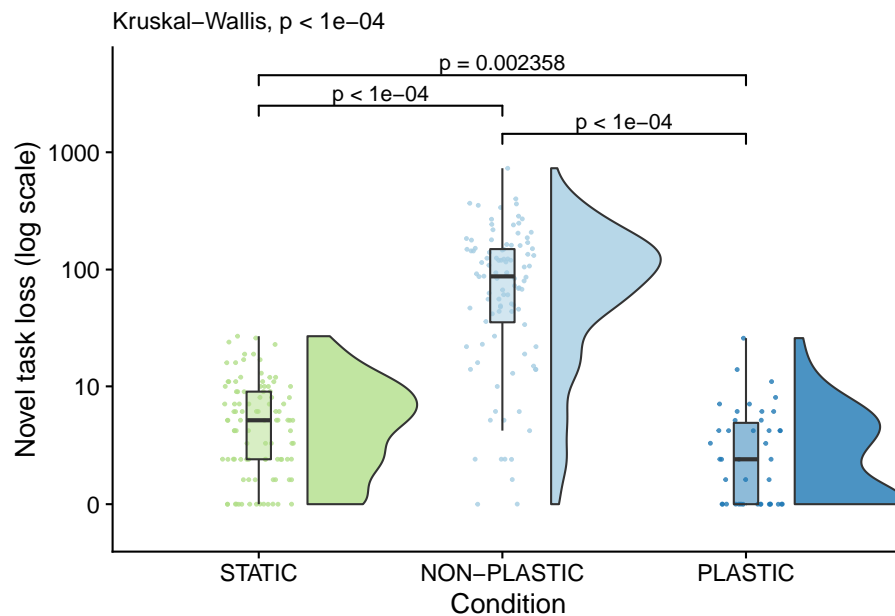
```

geom_flat_violin(
  # data=filter(summary_data, is_outlier==FALSE),
  scale="width",
  position = position_nudge(x = .2, y = 0),
  alpha = .8
) +
geom_point(
  mapping=aes(color=condition),
  position = position_jitter(width = .15),
  size = .5,
  alpha = 0.8
) +
geom_boxplot(
  width = .1,
  outlier.shape = NA,
  alpha = 0.5
) +
scale_x_discrete(
  name="Condition",
  limits=condition_order,
  labels=condition_order
) +
scale_y_continuous(
  name="Novel task loss (log scale)",
  trans=pseudo_log_trans(sigma=1, base=10),
  breaks=c(0,10,100,1000),
  limits=c(-1,5000)
) +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
labs(
  subtitle=paste0(
    "Kruskal-Wallis, ",
    p_label(signif(kruskal.test(formula=dominant_lineage_extra_traits_lost~condition, data=summ
  )
) +
ggsignif::geom_signif(
  data=filter(stat.test, p.adj<=alpha),
  aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
  manual=TRUE,
  inherit.aes=FALSE

```

```
) +
# coord_flip()
theme(
  legend.position="none"
)
```

```
## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
lineage_novel_task_loss_fig
```



```
kruskal.test(
  formula=dominant_lineage_extra_traits_lost~condition,
  data=summary_data
)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_extra_traits_lost by condition
## Kruskal-Wallis chi-squared = 129.06, df = 2, p-value < 2.2e-16
```

```
pairwise.wilcox.test(
  x=summary_data$dominant_lineage_extra_traits_lost,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
  conf.int=TRUE,
```



```

    conf.level=0.95
  )

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$dominant_lineage_extra_traits_lost and summary_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC 2.7e-16      -
## STATIC  < 2e-16    0.0024
##
## P value adjustment method: bonferroni
paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$dominant_lineage_extra_traits_lost)
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$dominant_lineage_extra_traits_lost)
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$dominant_lineage_extra_traits_lost)
  )
)

## [1] "PLASTIC median: 2; STATIC median: 5; NON-PLASTIC median: 87.5"
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"
for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=dominant_lineage_extra_traits_lost~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

```

```
## [1] "STATIC<-->NON-PLASTIC: W=9105"
## [1] "STATIC<-->PLASTIC: W=1353.5"
## [1] "PLASTIC<-->NON-PLASTIC: W=3959"
```

6.12 Frequency of novel task loss (lineage)

```
summary_data$dominant_lineage_extra_traits_lost_per_generation <- summary_data$dominant_lineage_extra_traits_lost_per_generation
summary_data$dominant_lineage_extra_traits_generations_per_loss <- summary_data$dominant_lineage_extra_traits_generations_per_loss

# Compute manual labels for geom_signif
# stat.test <- filter(summary_data, dominant_lineage_extra_traits_lost > 0) %>%
#   wilcox_test(dominant_lineage_extra_traits_generations_per_loss ~ condition) %>%
#   adjust_pvalue(method = "bonferroni") %>%
#   add_significance() %>%
#   add_xy_position(x="condition", step.increase=1)
stat.test <- summary_data %>%
  wilcox_test(dominant_lineage_extra_traits_lost_per_generation ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="condition", step.increase=.1)
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior)
stat.test$manual_position <- stat.test$y.position ## c(1.0,1.0,1.03)
stat.test$label <- mapply(p_label,stat.test$p.adj)

summary_data$is_outlier <- mapply(
  is_outlier,
  summary_data$dominant_lineage_extra_traits_lost_per_generation,
  summary_data$condition,
  MoreArgs=list(data=summary_data, column="dominant_lineage_extra_traits_lost_per_generation")
)

lineage_novel_task_loss_freq_fig <- ggplot(
  # filter(summary_data, dominant_lineage_extra_traits_lost > 0),
  summary_data,
  aes(x=condition, y=dominant_lineage_extra_traits_lost_per_generation, fill=condition)
) +
  geom_flat_violin(
    # data=filter(summary_data, is_outlier==FALSE),
    scale="width",
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
```

```

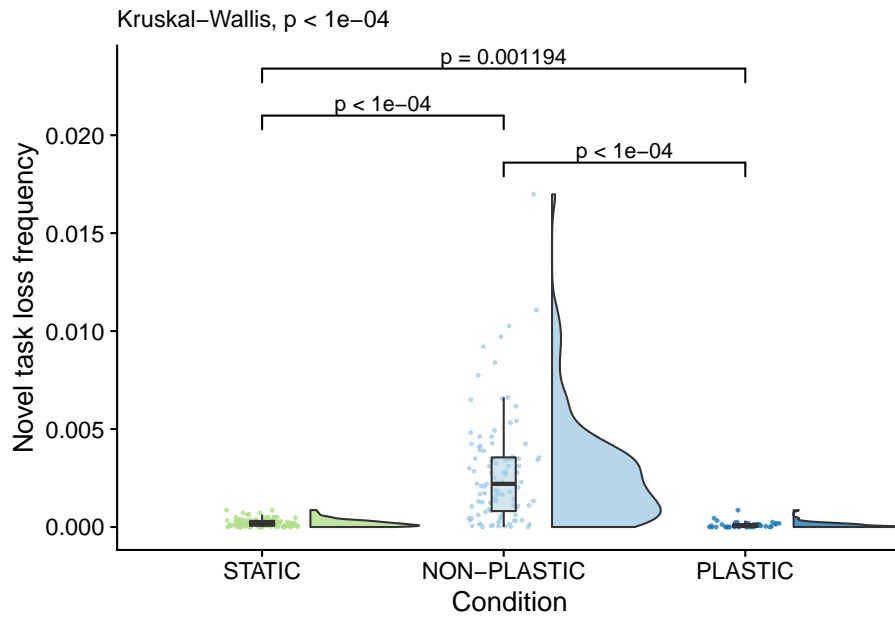
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  ylab("Novel task loss frequency") +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  labs(
    subtitle=paste0(
      "Kruskal-Wallis, ",
      p_label(signif(kruskal.test(formula=dominant_lineage_extra_traits_lost_per_generation~condi
    )
  ) +
  ggsignif::geom_signif(
    data=filter(stat.test, p.adj<=alpha),
    aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
    manual=TRUE,
    inherit.aes=FALSE
  ) +
  theme(
    legend.position="none"
  )
)

```

```

## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
lineage_novel_task_loss_freq_fig

```



```
kruskal.test(
  formula=dominant_lineage_extra_traits_lost_per_generation~condition,
  data=summary_data
)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_extra_traits_lost_per_generation by condition
## Kruskal-Wallis chi-squared = 121.41, df = 2, p-value < 2.2e-16
```

```
pairwise.wilcox.test(
  x=summary_data$dominant_lineage_extra_traits_lost_per_generation,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
  conf.int=TRUE,
  conf.level=0.95
)
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$dominant_lineage_extra_traits_lost_per_generation and summary_d
##
##      NON-PLASTIC PLASTIC
## PLASTIC 1.1e-15      -
```

6.13. HOW MANY INSTANCES OF NOVEL TRAIT LOSS CO-OCCURRED WITH CHANGES IN BASE PHENO

```
## STATIC < 2e-16      0.0012
##
## P value adjustment method: bonferroni

paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(summary_data, condition=="PLASTIC"))$dominant_lineage_extra_traits_lost_per_generation,
  ),
  paste0(
    "STATIC median: ",
    median(filter(summary_data, condition=="STATIC"))$dominant_lineage_extra_traits_lost_per_generation,
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(summary_data, condition=="NON-PLASTIC"))$dominant_lineage_extra_traits_lost_per_generation,
  )
)

## [1] "PLASTIC median: 6.25141973661864e-05; STATIC median: 0.000161396283669756; NON-PLASTIC median: 0.000161396283669756"
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"

for (pair in pairwise_comparisons) {
  pair_data <- filter(summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=dominant_lineage_extra_traits_lost_per_generation~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

## [1] "STATIC<-->NON-PLASTIC: W=8940"
## [1] "STATIC<-->PLASTIC: W=1311"
## [1] "PLASTIC<-->NON-PLASTIC: W=3922"
```

6.13 How many instances of novel trait loss co-occurred with changes in base phenotype?

Task loss linked with primary trait changes.

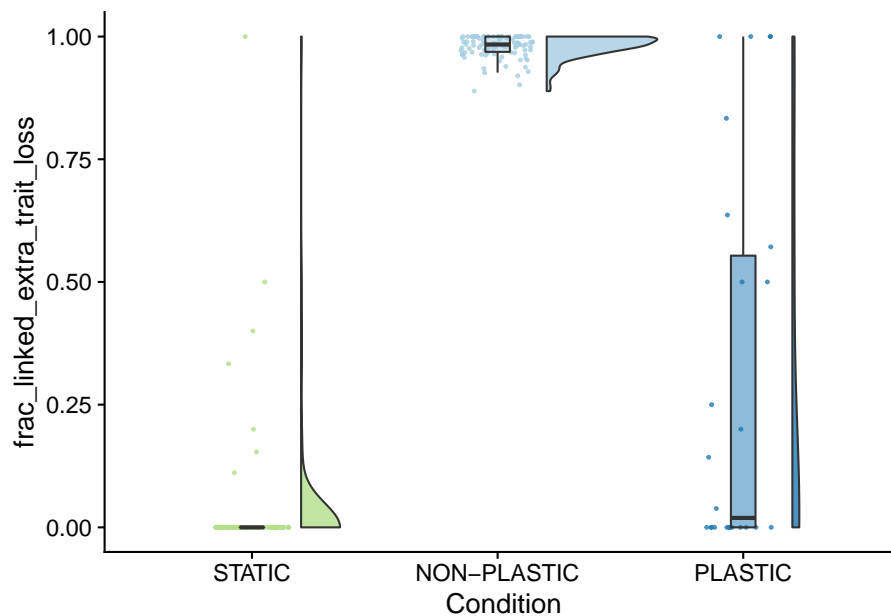
```

lost_traits_summary_data <- filter(summary_data, extra_task_value==0.1 & dominant_line)
lost_traits_summary_data$frac_linked_extra_trait_loss <- lost_traits_summary_data$dominant_line

ggplot(lost_traits_summary_data, aes(x=condition, y=frac_linked_extra_trait_loss, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  # coord_flip() +
  theme(
    legend.position="none"
  )

```

6.13. HOW MANY INSTANCES OF NOVEL TRAIT LOSS CO-OCCURRED WITH CHANGES IN BASE PHENO



```
kruskal.test(
  formula=frac_linked_extra_trait_loss~condition,
  data=lost_traits_summary_data
)

##
##  Kruskal-Wallis rank sum test
##
## data:  frac_linked_extra_trait_loss by condition
## Kruskal-Wallis chi-squared = 153.68, df = 2, p-value < 2.2e-16

pairwise.wilcox.test(
  x=lost_traits_summary_data$frac_linked_extra_trait_loss,
  g=lost_traits_summary_data$condition,
  p.adjust.method="bonferroni",
  conf.int=TRUE,
  conf.level=0.95
)

##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  lost_traits_summary_data$frac_linked_extra_trait_loss and lost_traits_summary_data$condition
##
##           NON-PLASTIC PLASTIC
## PLASTIC 1.9e-08      -
```

```
## STATIC < 2e-16      1.8e-06
##
## P value adjustment method: bonferroni

paste(
  sep="; ",
  paste0(
    "PLASTIC median: ",
    median(filter(loss_traits_summary_data, condition=="PLASTIC"))$frac_linked_extra_trait_loss
  ),
  paste0(
    "STATIC median: ",
    median(filter(loss_traits_summary_data, condition=="STATIC"))$frac_linked_extra_trait_loss
  ),
  paste0(
    "NON-PLASTIC median: ",
    median(filter(loss_traits_summary_data, condition=="NON-PLASTIC"))$frac_linked_extra_trait_loss
  )
)

## [1] "PLASTIC median: 0.0192307692307692; STATIC median: 0; NON-PLASTIC median: 0.983076923076923"
print("Wilcox rank sum test statistics:")

## [1] "Wilcox rank sum test statistics:"
for (pair in pairwise_comparisons) {
  pair_data <- filter(loss_traits_summary_data, condition %in% pair)
  pair_data$condition <- as.factor(pair_data$condition)
  wt <- wilcox.test(
    formula=frac_linked_extra_trait_loss~condition,
    data=pair_data,
    exact=FALSE,
    paired=FALSE
  )
  print(paste0(pair[1], "<-->", pair[2], ": W=", wt$statistic))
}

## [1] "STATIC<-->NON-PLASTIC: W=8344"
## [1] "STATIC<-->PLASTIC: W=1602"
## [1] "PLASTIC<-->NON-PLASTIC: W=2212"

sum(filter(loss_traits_summary_data, condition=="NON-PLASTIC"))$dominant_lineage_extra_trait_loss

## [1] 10998

sum(filter(loss_traits_summary_data, condition=="NON-PLASTIC"))$dominant_lineage_extra_trait_loss

## [1] 11229
```



```

aggregate_frac_linked_extra_trait_loss_nonplastic <- sum(filter(lost_traits_summary_data, condition=="PLASTIC"))$dominant_lineage_extra_traits_lost_link
aggregate_frac_linked_extra_trait_loss_nonplastic

## [1] 0.9794283
sum(filter(lost_traits_summary_data, condition=="PLASTIC"))$dominant_lineage_extra_traits_lost_link

## [1] 29
sum(filter(lost_traits_summary_data, condition=="PLASTIC"))$dominant_lineage_extra_traits_lost

## [1] 142
aggregate_frac_linked_extra_trait_loss_plastic <- sum(filter(lost_traits_summary_data, condition=="STATIC"))$dominant_lineage_extra_traits_lost_link
aggregate_frac_linked_extra_trait_loss_plastic

## [1] 0.2042254
sum(filter(lost_traits_summary_data, condition=="STATIC"))$dominant_lineage_extra_traits_lost_link

## [1] 13
sum(filter(lost_traits_summary_data, condition=="STATIC"))$dominant_lineage_extra_traits_lost

## [1] 631
aggregate_frac_linked_extra_trait_loss_nonplastic <- sum(filter(lost_traits_summary_data, condition=="PLASTIC"))$dominant_lineage_extra_traits_lost_link
aggregate_frac_linked_extra_trait_loss_nonplastic

## [1] 0.02060222

```

6.14 Manuscript figures

6.15 Combined panel

```

# grid <- plot_grid(
#   final_novel_task_count_fig +
#     theme(
#       axis.title.x=element_blank()
#     ) +
#   ggtitle("Final novel task count"),
#   lineage_novel_task_discovery_fig +
#     theme(
#       axis.title.x=element_blank()
#     ) +
#   ggtitle("Novel task discovery"),
#   lineage_novel_task_loss_fig +
#     theme(

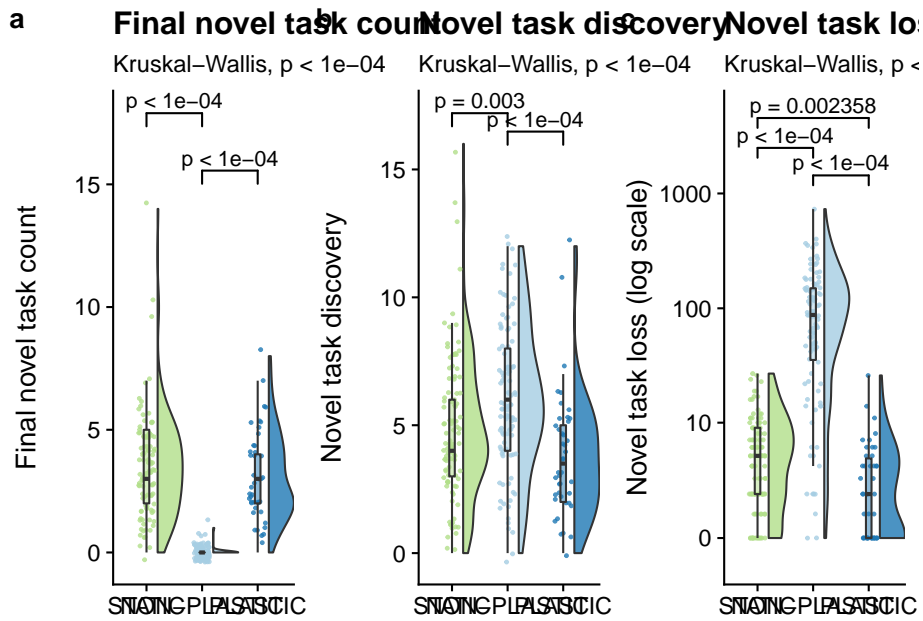
```

```

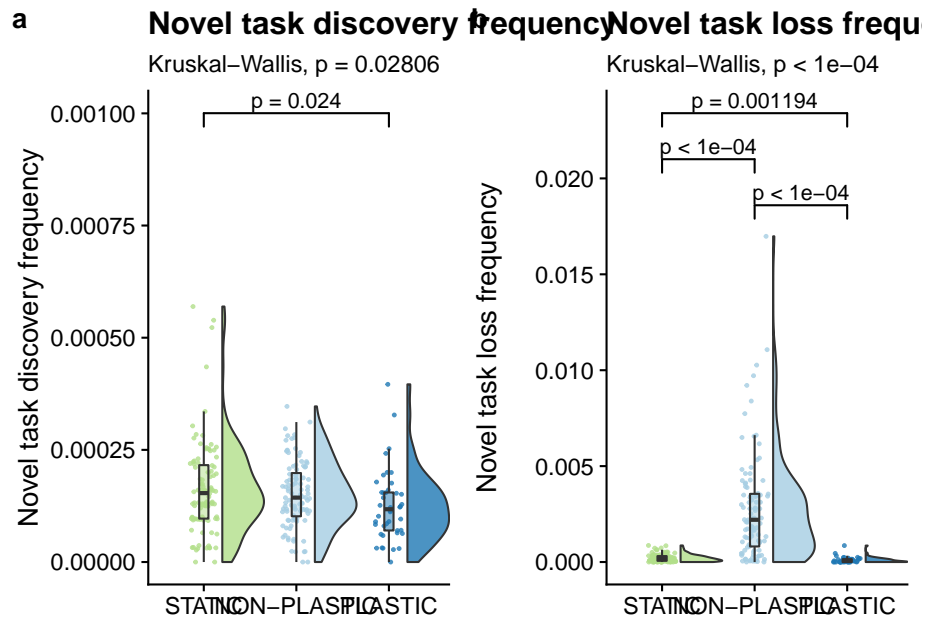
#       axis.title.x=element_blank()
#     ) +
#     ggtitle("Novel task loss"),
#     NULL,
#     lineage_novel_task_discovery_freq_fig +
#     theme(
#       axis.title.x=element_blank()
#     ) +
#     ggtitle("Novel task discovery frequency"),
#     lineage_novel_task_loss_freq_fig +
#     theme(
#       axis.title.x=element_blank()
#     ) +
#     ggtitle("Novel task loss frequency"),
#     nrow=2,
#     align="v",
#     # labels="auto"
#     labels=c("a", "b", "c", "", "d", "e")
# )

magnitude_grid <- plot_grid(
  final_novel_task_count_fig +
    theme(
      axis.title.x=element_blank()
    ) +
    ggtitle("Final novel task count"),
  lineage_novel_task_discovery_fig +
    theme(
      axis.title.x=element_blank()
    ) +
    ggtitle("Novel task discovery"),
  lineage_novel_task_loss_fig +
    theme(
      axis.title.x=element_blank()
    ) +
    ggtitle("Novel task loss"),
  nrow=1,
  align="v",
  labels="auto"
)
magnitude_grid

```



```
pace_grid <- plot_grid(
  lineage_novel_task_discovery_freq_fig +
    theme(
      axis.title.x=element_blank()
    ) +
    ggtitle("Novel task discovery frequency"),
  lineage_novel_task_loss_freq_fig +
    theme(
      axis.title.x=element_blank()
    ) +
    ggtitle("Novel task loss frequency"),
  nrow=1,
  align="v",
  labels="auto"
)
pace_grid
```



```

save_plot(
  paste0(working_directory, "plots/", "complex-traits-magnitude-panel.pdf"),
  magnitude_grid,
  base_height=6,
  base_asp=3/1
)

save_plot(
  paste0(working_directory, "plots/", "complex-traits-pace-panel.pdf"),
  pace_grid,
  base_height=6,
  base_asp=2/1
)

# save_plot(
#   paste0(working_directory, "plots/", "complex-traits-panel.pdf"),
#   grid,
#   base_height=12,
#   base_asp=3/2
# )
grid

## function (nx = NULL, ny = nx, col = "lightgray", lty = "dotted",
##         lwd = par("lwd"), equilog = TRUE)
## {

```

```

##      if (is.null(nx) || (!is.na(nx) && nx >= 1)) {
##          log <- par("xlog")
##          if (is.null(nx)) {
##              ax <- par("xaxp")
##              if (log && equilogs && ax[3L] > 0)
##                  ax[3L] <- 1
##              at <- axTicks(1, axp = ax, log = log)
##          }
##          else {
##              U <- par("usr")
##              at <- seq.int(U[1L], U[2L], length.out = nx + 1)
##              at <- (if (log)
##                  10^at
##                  else at)[-c(1, nx + 1)]
##          }
##          abline(v = at, col = col, lty = lty, lwd = lwd)
##      }
##      if (is.null(ny) || (!is.na(ny) && ny >= 1)) {
##          log <- par("ylog")
##          if (is.null(ny)) {
##              ax <- par("yaxp")
##              if (log && equilogs && ax[3L] > 0)
##                  ax[3L] <- 1
##              at <- axTicks(2, axp = ax, log = log)
##          }
##          else {
##              U <- par("usr")
##              at <- seq.int(U[3L], U[4L], length.out = ny + 1)
##              at <- (if (log)
##                  10^at
##                  else at)[-c(1, ny + 1)]
##          }
##          abline(h = at, col = col, lty = lty, lwd = lwd)
##      }
##  }
## <bytecode: 0x561e01e29ec8>
## <environment: namespace:graphics>

```


Chapter 7

Accumulation of deleterious instructions

The effect of adaptive phenotypic plasticity on the accumulation of deleterious genes.

7.1 Overview

```
total_updates <- 200000
replicates <- 100
alpha <- 0.05
focal_poison_penalty <- 0.1

focal_traits <- c("not", "nand", "and", "ornot", "or", "andnot")
traits_set_a <- c("not", "and", "or")
traits_set_b <- c("nand", "ornot", "andnot")

# Relative location of data.
working_directory <- "experiments/2021-02-05-hitchhiking/analysis/" # << For bookdown
# working_directory <- "./"
```

7.2 Analysis dependencies

Load all required R libraries.

```
library(RColorBrewer)
library(ggplot2)
library(rstatix)
```

```
library(ggsignif)
library(scales)
library(tidyverse)
library(cowplot)
library(Hmisc)
library(boot)
library(fmsb)
library(knitr)
source("https://gist.githubusercontent.com/benmarwick/2a1bb0133ff568cbe28d/raw/fb53bd9")
```

These analyses were conducted/knitted with the following computing environment:

```
print(version)

##
## platform      x86_64-pc-linux-gnu
## arch          x86_64
## os            linux-gnu
## system        x86_64, linux-gnu
## status
## major         4
## minor         1.0
## year          2021
## month         05
## day           18
## svn rev       80317
## language      R
## version.string R version 4.1.0 (2021-05-18)
## nickname      Camp Pontanezen
```

7.3 Setup

```
##### summary data #####
summary_data_loc <- paste0(working_directory, "data/aggregate.csv")
summary_data <- read.csv(summary_data_loc, na.strings="NONE")

summary_data$DISABLE_REACTION_SENSORS <- as.factor(summary_data$DISABLE_REACTION_SENSORS)
summary_data$chg_env <- summary_data$chg_env == "True"
summary_data$dominant_plastic_odd_even <- as.factor(summary_data$dominant_plastic_odd_even)
summary_data$sensors <- summary_data$DISABLE_REACTION_SENSORS == "0"
summary_data$is_plastic <- summary_data$dominant_plastic_odd_even == "True"
summary_data$POISON_PENALTY <- as.factor(summary_data$POISON_PENALTY)

summary_data$dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation <-
```



```

summary_data$frac_hitchhiking_linked_trait_change <- summary_data$dominant_lineage_num_times_hitchhike
summary_data$frac_unexpressed_hitchhiker_inc <- summary_data$dominant_lineage_num_times_hitchhike
summary_data$frac_expressed_hitchhiker_inc <- summary_data$dominant_lineage_num_times_hitchhike_in

env_label_fun <- function(chg_env) {
  if (chg_env) {
    return("Fluctuating")
  } else {
    return("Constant")
  }
}

sensors_label_fun <- function(has_sensors) {
  if (has_sensors) {
    return("Sensors")
  } else {
    return("No sensors")
  }
}

condition_label_fun <- function(has_sensors, env_chg) {
  if (has_sensors && env_chg) {
    return("PLASTIC")
  } else if (env_chg) {
    return("NON-PLASTIC")
  } else {
    return("STATIC")
  }
}

summary_data$env_label <- mapply(
  env_label_fun,
  summary_data$chg_env
)

summary_data$sensors_label <- mapply(
  sensors_label_fun,
  summary_data$sensors
)

summary_data$condition <- mapply(
  condition_label_fun,
  summary_data$sensors,
  summary_data$chg_env
)

condition_order = c(

```

```

    "STATIC",
    "NON-PLASTIC",
    "PLASTIC"
  )

pairwise_comparisons <- list(
  c("STATIC", "NON-PLASTIC"),
  c("STATIC", "PLASTIC"),
  c("PLASTIC", "NON-PLASTIC")
)

p_label <- function(p_value) {
  threshold = 0.0001
  if (p_value < threshold) {
    return(paste0("p < ", threshold))
  } else {
    return(paste0("p = ", p_value))
  }
}

poison_penalties <- levels(summary_data$POISON_PENALTY)

##### time series #####
lineage_time_series_data_loc <- paste0(working_directory, "data/lineage_series.csv")
lineage_time_series_data <- read.csv(lineage_time_series_data_loc)

lineage_time_series_data$DISABLE_REACTION_SENSORS <- as.factor(lineage_time_series_data$DISABLE_REACTION_SENSORS == "True")
lineage_time_series_data$chg_env <- lineage_time_series_data$chg_env == "True"
lineage_time_series_data$sensors <- lineage_time_series_data$DISABLE_REACTION_SENSORS == "True"
lineage_time_series_data$POISON_PENALTY <- as.factor(lineage_time_series_data$POISON_PENALTY)

lineage_time_series_data$env_label <- mapapply(
  env_label_fun,
  lineage_time_series_data$chg_env
)

lineage_time_series_data$sensors_label <- mapapply(
  sensors_label_fun,
  lineage_time_series_data$sensors
)

lineage_time_series_data$condition <- mapapply(
  condition_label_fun,
  lineage_time_series_data$sensors,
  lineage_time_series_data$chg_env
)

```

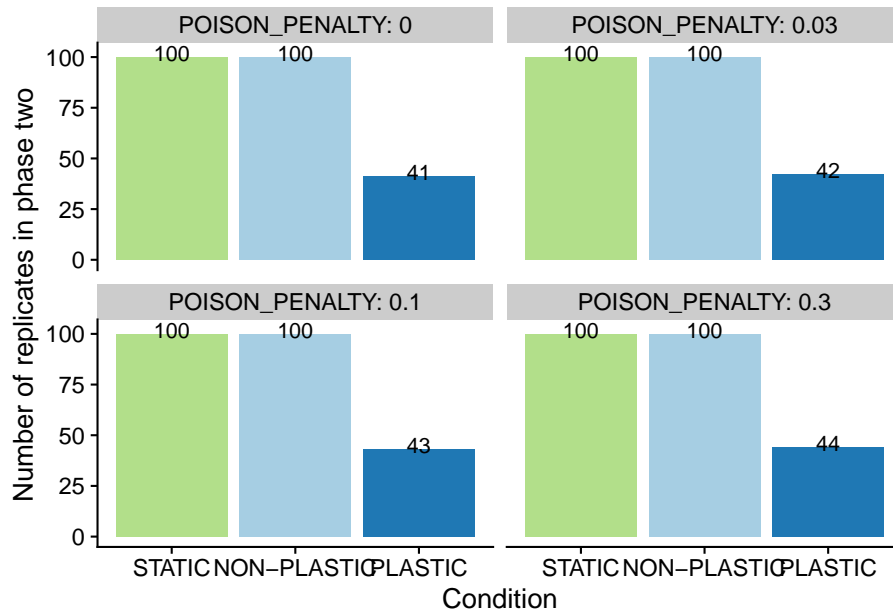
```
##### misc #####
# Configure our default graphing theme
focal_summary_data <- filter(summary_data, POISON_PENALTY==focal_poison_penalty)
theme_set(theme_cowplot())
cb_palette <- "Paired"
dir.create(paste0(working_directory, "plots"), showWarnings=FALSE)
samplemean <- function(x, d) {
  return(mean(x[d]))
}
```

7.4 Evolution of phenotypic plasticity

For sensor-enabled populations in fluctuating environments, we only transferred populations containing an optimally plastic genotype to phase-two.

```
summary_data_grouped = dplyr::group_by(summary_data, sensors, env_label, condition, POISON_PENALTY)
summary_data_group_counts = dplyr::summarize(summary_data_grouped, n=dplyr::n())
```

```
ggplot(summary_data_group_counts, aes(x=condition, y=n, fill=condition)) +
  geom_col(position=position_dodge(0.9)) +
  geom_text(aes(label=n, y=n+2)) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  ylab("Number of replicates in phase two") +
  facet_wrap(~POISON_PENALTY, labeller=label_both) +
  theme(
    legend.position="none"
  )
```

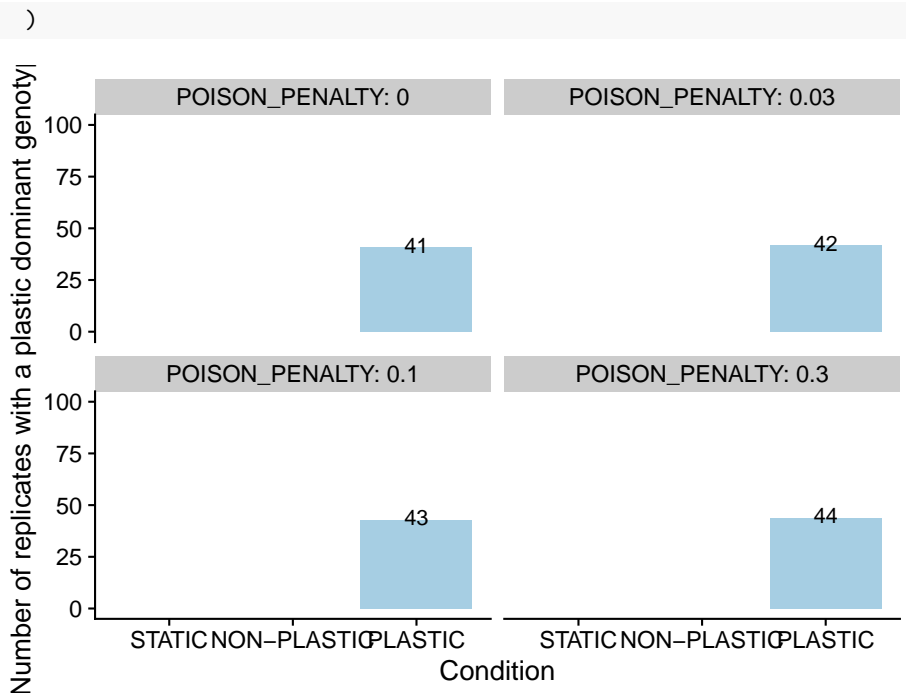


We can confirm our expectation that the dominant genotypes in non-plastic conditions are not phenotypically plastic.

```
summary_data_grouped = dplyr::group_by(summary_data, condition, is_plastic, POISON_PENALTY)
summary_data_group_counts = dplyr::summarize(summary_data_grouped, n=dplyr::n())
```

`summarise()` has grouped output by 'condition', 'is_plastic'. You can override using `ungroup()`.

```
ggplot(filter(summary_data_group_counts, is_plastic), aes(x=condition, y=n, fill=condition)) +
  geom_col(position=position_dodge(0.9)) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  geom_text(aes(label=n, y=n+1)) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  ylab("Number of replicates with a plastic dominant genotype") +
  ylim(0, 100) +
  facet_wrap(~POISON_PENALTY, labeller=label_both) +
  theme(
    legend.position="none"
```



7.5 Poison instruction execution

7.5.1 Number of replicates where final dominant genotype executes the poison instruction

```
for (penalty in poison_penalties) {
  occurrences <- c(
    length(filter(summary_data, POISON_PENALTY==penalty & condition=="NON-PLASTIC" & dominant_times_p
    length(filter(summary_data, POISON_PENALTY==penalty & condition=="PLASTIC" & dominant_times_p
    length(filter(summary_data, POISON_PENALTY==penalty & condition=="STATIC" & dominant_times_p
  )
  trials <- c(
    length(filter(summary_data, POISON_PENALTY==penalty & condition=="NON-PLASTIC")$RANDOM_SEED),
    length(filter(summary_data, POISON_PENALTY==penalty & condition=="PLASTIC")$RANDOM_SEED),
    length(filter(summary_data, POISON_PENALTY==penalty & condition=="STATIC" )$RANDOM_SEED)
  )
  names(trials) <- c(
    "NON-PLASTIC",
    "PLASTIC",
    "STATIC"
  )
}
```

```

names(occurrences) <- c(
  "NON-PLASTIC",
  "PLASTIC",
  "STATIC"
)
poison_exec_table <- data.frame(
  executes.poison=occurrences,
  replicates=trials
)
cat(paste0("#### Penalty: ", penalty, "\n"))
cat(print(kable(poison_exec_table)))
cat("\n")
ft <- pairwise.fisher.test(x=occurrences, n=trials, p.adjust.method="bonferroni")
print(ft)
cat("\n\n")
}

```

```

## #### Penalty: 0
##
## \begin{tabular}{l|r|r}
## \hline
##   & executes.poison & replicates\\
## \hline
## NON-PLASTIC & 86 & 100\\
## \hline
## PLASTIC & 27 & 41\\
## \hline
## STATIC & 85 & 100\\
## \hline
## \end{tabular}
##
##
## Pairwise comparisons using Pairwise comparison of proportions (Fisher)
##
## data:  occurrences out of trials
##
##           NON-PLASTIC PLASTIC
## PLASTIC 0.03           -
## STATIC  1.00           0.06
##
## P value adjustment method: bonferroni
##
##
## #### Penalty: 0.03
##

```

```

## \begin{tabular}{l|r|r}
## \hline
##   & executes.poison & replicates\\
## \hline
## NON-PLASTIC & 46 & 100\\
## \hline
## PLASTIC & 1 & 42\\
## \hline
## STATIC & 1 & 100\\
## \hline
## \end{tabular}
##
##
## Pairwise comparisons using Pairwise comparison of proportions (Fisher)
##
## data:  occurrences out of trials
##
##           NON-PLASTIC PLASTIC
## PLASTIC 1.2e-07      -
## STATIC  2.9e-15      1
##
## P value adjustment method: bonferroni
##
##
## ##### Penalty: 0.1
##
## \begin{tabular}{l|r|r}
## \hline
##   & executes.poison & replicates\\
## \hline
## NON-PLASTIC & 14 & 100\\
## \hline
## PLASTIC & 0 & 43\\
## \hline
## STATIC & 0 & 100\\
## \hline
## \end{tabular}
##
##
## Pairwise comparisons using Pairwise comparison of proportions (Fisher)
##
## data:  occurrences out of trials
##
##           NON-PLASTIC PLASTIC
## PLASTIC 0.03212      -
## STATIC  0.00022      1.00000

```

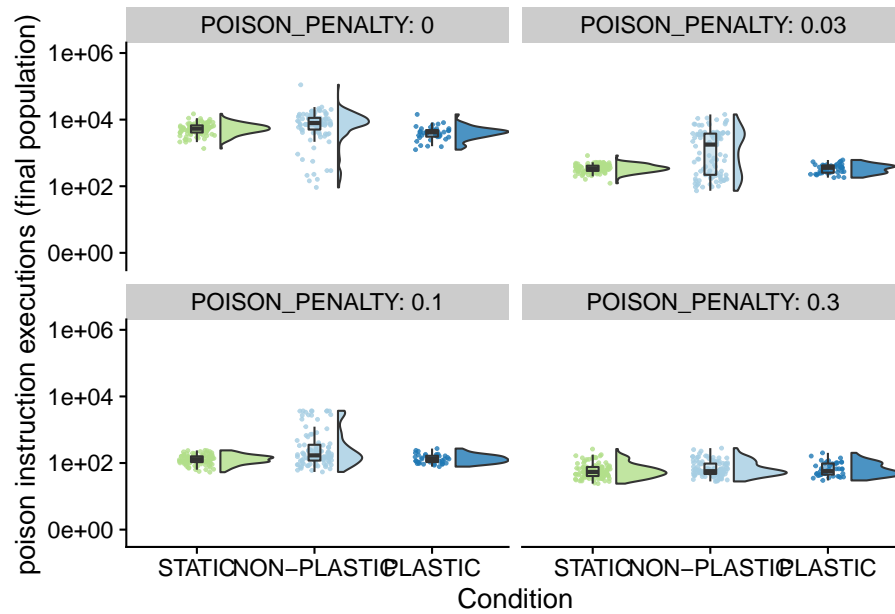
```
##
## P value adjustment method: bonferroni
##
##
## ##### Penalty: 0.3
##
## \begin{tabular}{l|r|r}
## \hline
##   & executes.poison & replicates\\
## \hline
## NON-PLASTIC & 0 & 100\\
## \hline
## PLASTIC & 0 & 44\\
## \hline
## STATIC & 0 & 100\\
## \hline
## \end{tabular}
##
##
## Pairwise comparisons using Pairwise comparison of proportions (Fisher)
##
## data:  occurrences out of trials
##
##          NON-PLASTIC PLASTIC
## PLASTIC 1             -
## STATIC  1             1
##
## P value adjustment method: bonferroni
```

7.5.2 Poison instruction execution (final population)

```
ggplot(summary_data, aes(x=condition, y=final_population_poison, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
```



```
alpha = 0.5
) +
scale_x_discrete(
  name="Condition",
  limits=condition_order
) +
scale_y_continuous(
  name="poison instruction executions (final population)",
  trans=pseudo_log_trans(sigma=1,base=10),
  breaks=c(0,100,10000,1000000),
  limits=c(-1,1000000)
) +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
facet_wrap(
  ~POISON_PENALTY,
  labeller=label_both
) +
# coord_flip() +
theme(
  legend.position="none"
) +
ggsave(
  paste0(working_directory, "plots/final-population-poison-log.pdf"),
  width=15,
  height=10
)
```



```

for (penalty in poison_penalties) {
  stat_data <- filter(summary_data, POISON_PENALTY==penalty)
  print(
    paste0(
      "PENALTY: ", penalty
    )
  )
  kt <- kruskal.test(
    formula=final_population_poison~condition,
    data=stat_data
  )
  print(
    kt
  )
  if (is.na(kt$p.value)) { next }
  if (kt$p.value > 0.05) { next }
  print(
    pairwise.wilcox.test(
      x=stat_data$final_population_poison,
      g=stat_data$condition,
      p.adjust.method="bonferroni"
    )
  )
}

```

```
## [1] "PENALTY: 0"
##
## Kruskal-Wallis rank sum test
##
## data: final_population_poison by condition
## Kruskal-Wallis chi-squared = 43.589, df = 2, p-value = 3.426e-10
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: stat_data$final_population_poison and stat_data$condition
##
##      NON-PLASTIC PLASTIC
## PLASTIC 8.7e-07      -
## STATIC  9.8e-07      0.00074
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.03"
##
## Kruskal-Wallis rank sum test
##
## data: final_population_poison by condition
## Kruskal-Wallis chi-squared = 20.74, df = 2, p-value = 3.136e-05
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: stat_data$final_population_poison and stat_data$condition
##
##      NON-PLASTIC PLASTIC
## PLASTIC 0.003      -
## STATIC  1e-04      1.000
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.1"
##
## Kruskal-Wallis rank sum test
##
## data: final_population_poison by condition
## Kruskal-Wallis chi-squared = 20.608, df = 2, p-value = 3.35e-05
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: stat_data$final_population_poison and stat_data$condition
##
```

```
##          NON-PLASTIC PLASTIC
## PLASTIC 0.0093      -
## STATIC 4.9e-05     1.0000
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.3"
##
## Kruskal-Wallis rank sum test
##
## data: final_population_poison by condition
## Kruskal-Wallis chi-squared = 3.3994, df = 2, p-value = 0.1827
```

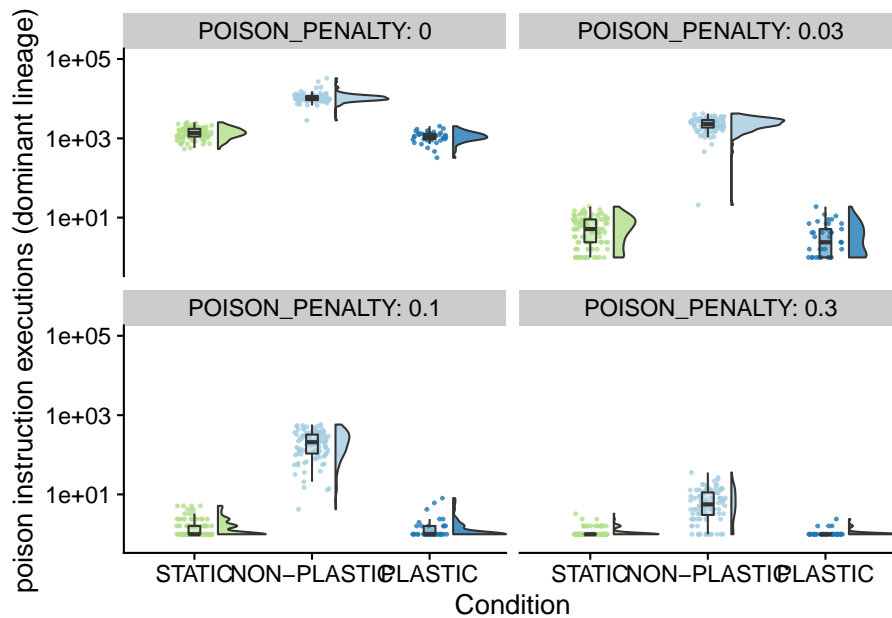
7.5.3 Cumulative poison instruction execution along final dominant lineages

```
ggplot(summary_data, aes(x=condition, y=dominant_lineage_times_poison_executed, fill=c
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  scale_y_continuous(
    name="poison instruction executions (dominant lineage)",
    trans=pseudo_log_trans(sigma = 1, base = 10),
    breaks=c(10,1000,100000),
    limits=c(-1,100000)
  ) +
  facet_wrap(
    ~POISON_PENALTY,
    labeller=label_both
  ) +
```

```

scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
theme(
  legend.position="none"
) +
ggsave(
  paste0(working_directory, "plots/final-dominant-lineage-poison-log.pdf"),
  width=15,
  height=10
)

```



```

for (penalty in poison_penalties) {
  stat_data <- filter(summary_data, POISON_PENALTY==penalty)
  print(
    paste0(
      "PENALTY: ", penalty
    )
  )
  kt <- kruskal.test(
    formula=dominant_lineage_times_poison_executed~condition,
    data=stat_data
  )
}

```

```

    )
    print(
      kt
    )
    if (is.na(kt$p.value)) { next }
    if (kt$p.value > 0.05) { next }
    print(
      pairwise.wilcox.test(
        x=stat_data$dominant_lineage_times_poison_executed,
        g=stat_data$condition,
        p.adjust.method="bonferroni"
      )
    )
  }
}

```

```

## [1] "PENALTY: 0"
##
##   Kruskal-Wallis rank sum test
##
## data:  dominant_lineage_times_poison_executed by condition
## Kruskal-Wallis chi-squared = 178.84, df = 2, p-value < 2.2e-16
##
##
##   Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  stat_data$dominant_lineage_times_poison_executed and stat_data$condition
##
##           NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      0.0018
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.03"
##
##   Kruskal-Wallis rank sum test
##
## data:  dominant_lineage_times_poison_executed by condition
## Kruskal-Wallis chi-squared = 178.62, df = 2, p-value < 2.2e-16
##
##
##   Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  stat_data$dominant_lineage_times_poison_executed and stat_data$condition
##
##           NON-PLASTIC PLASTIC

```

```

## PLASTIC <2e-16      -
## STATIC  <2e-16      0.011
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.1"
##
## Kruskal-Wallis rank sum test
##
## data:  dominant_lineage_times_poison_executed by condition
## Kruskal-Wallis chi-squared = 184.83, df = 2, p-value < 2.2e-16
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  stat_data$dominant_lineage_times_poison_executed and stat_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      0.21
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.3"
##
## Kruskal-Wallis rank sum test
##
## data:  dominant_lineage_times_poison_executed by condition
## Kruskal-Wallis chi-squared = 149.48, df = 2, p-value < 2.2e-16
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  stat_data$dominant_lineage_times_poison_executed and stat_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC 4.4e-16      -
## STATIC  < 2e-16      0.84
##
## P value adjustment method: bonferroni

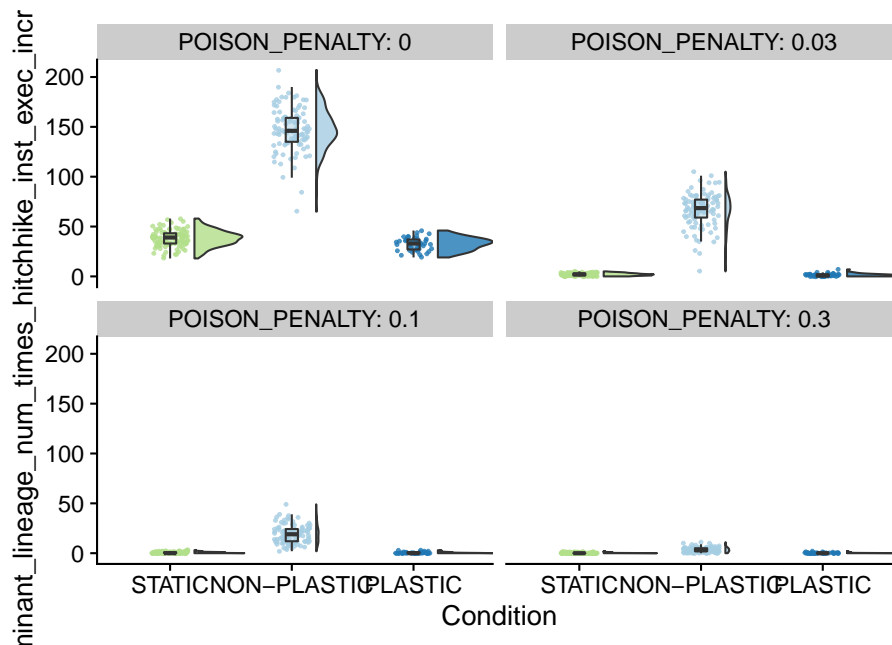
```

7.6 Characterizing mutations that increase poison instruction execution

```

ggplot(summary_data, aes(x=condition, y=dominant_lineage_num_times_hitchhike_inst_exec.
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  facet_wrap(
    ~POISON_PENALTY,
    labeller=label_both
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  theme(
    legend.position="none"
  )

```

```
for (penalty in poison_penalties) {
  stat_data <- filter(summary_data, POISON_PENALTY==penalty)
  print(
    paste0(
      "PENALTY: ", penalty
    )
  )
  kt <- kruskal.test(
    formula=dominant_lineage_num_times_hitchhike_inst_exec_incr~condition,
    data=stat_data
  )
  print(
    kt
  )
  if (is.na(kt$p.value)) { next }
  if (kt$p.value > 0.05) { next }
  print(
    pairwise.wilcox.test(
      x=stat_data$dominant_lineage_num_times_hitchhike_inst_exec_incr,
      g=stat_data$condition,
      p.adjust.method="bonferroni"
    )
  )
}
```

```

## [1] "PENALTY: 0"
##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_num_times_hitchhike_inst_exec_increases by condition
## Kruskal-Wallis chi-squared = 179.79, df = 2, p-value < 2.2e-16
##
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: stat_data$dominant_lineage_num_times_hitchhike_inst_exec_increases and stat_c
##
##      NON-PLASTIC PLASTIC
## PLASTIC < 2e-16      -
## STATIC  < 2e-16      0.00046
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.03"
##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_num_times_hitchhike_inst_exec_increases by condition
## Kruskal-Wallis chi-squared = 179.35, df = 2, p-value < 2.2e-16
##
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: stat_data$dominant_lineage_num_times_hitchhike_inst_exec_increases and stat_c
##
##      NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      0.03
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.1"
##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_num_times_hitchhike_inst_exec_increases by condition
## Kruskal-Wallis chi-squared = 185.34, df = 2, p-value < 2.2e-16
##
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: stat_data$dominant_lineage_num_times_hitchhike_inst_exec_increases and stat_c
##

```

7.6. CHARACTERIZING MUTATIONS THAT INCREASE POISON INSTRUCTION EXECUTION131

```
##          NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      0.27
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.3"
##
## Kruskal-Wallis rank sum test
##
## data:  dominant_lineage_num_times_hitchhike_inst_exec_increases by condition
## Kruskal-Wallis chi-squared = 146.35, df = 2, p-value < 2.2e-16
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  stat_data$dominant_lineage_num_times_hitchhike_inst_exec_increases and stat_data$condit
##
##          NON-PLASTIC PLASTIC
## PLASTIC 7.8e-16      -
## STATIC  < 2e-16      0.86
##
## P value adjustment method: bonferroni
# sum(filter(summary_data, condition=="NON-PLASTIC" & POISON_PENALTY==0.1)$dominant_lineage_num_t
# sum(filter(summary_data, condition=="PLASTIC" & POISON_PENALTY==0.1)$dominant_lineage_num_times
# sum(filter(summary_data, condition=="STATIC" & POISON_PENALTY==0.1)$dominant_lineage_num_times
```

Focal figure for the manuscript:

```
# Compute manual labels for geom_signif
stat.test <- focal_summary_data %>%
  wilcox_test(dominant_lineage_num_times_hitchhike_inst_exec_increases ~ condition) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="condition")
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior in g
stat.test$manual_position <- stat.test$y.position
stat.test$label <- mapply(p_label,stat.test$p.adj)

poison_increases_fig <- ggplot(
  focal_summary_data,
  aes(x=condition, y=dominant_lineage_num_times_hitchhike_inst_exec_increases, fill=condition)
) +
  geom_flat_violin(
    scale="width",
    position = position_nudge(x = .2, y = 0),
    alpha = .8
```

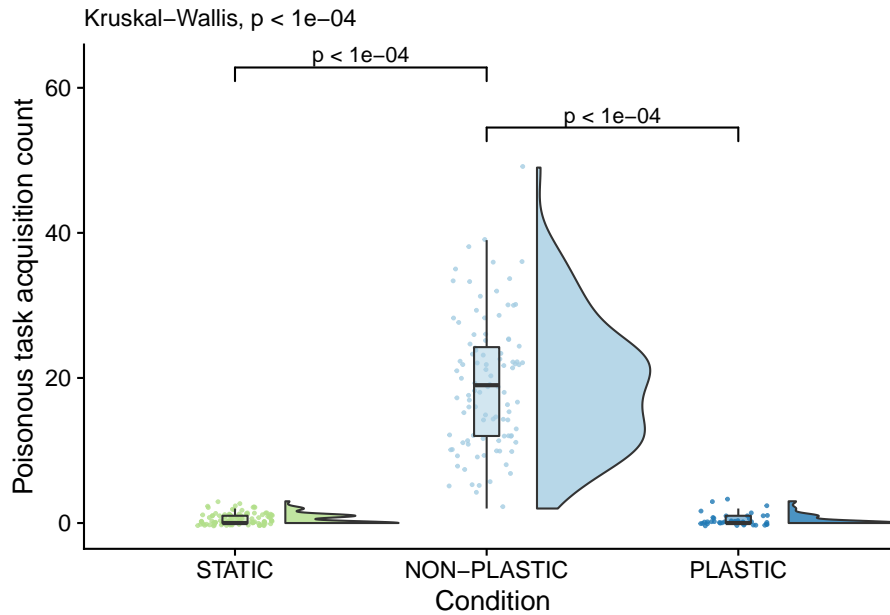
```

) +
geom_point(
  mapping=aes(color=condition),
  position = position_jitter(width = .15),
  size = .5,
  alpha = 0.8
) +
geom_boxplot(
  width = .1,
  outlier.shape = NA,
  alpha = 0.5
) +
scale_x_discrete(
  name="Condition",
  limits=condition_order,
  labels=condition_order
) +
scale_y_continuous(
  name="Poisonous task acquisition count",
) +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
# coord_flip()
labs(
  subtitle=paste0(
    "Kruskal-Wallis, ",
    p_label(signif(kruskal.test(formula=dominant_lineage_num_times_hitchhike_inst_ex
  )
) +
ggsignif::geom_signif(
  data=filter(stat.test, p.adj <= alpha),
  aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
  manual=TRUE,
  inherit.aes=FALSE
) +
theme(
  legend.position="none"
)

```

```
## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
```

poison_increases_fig



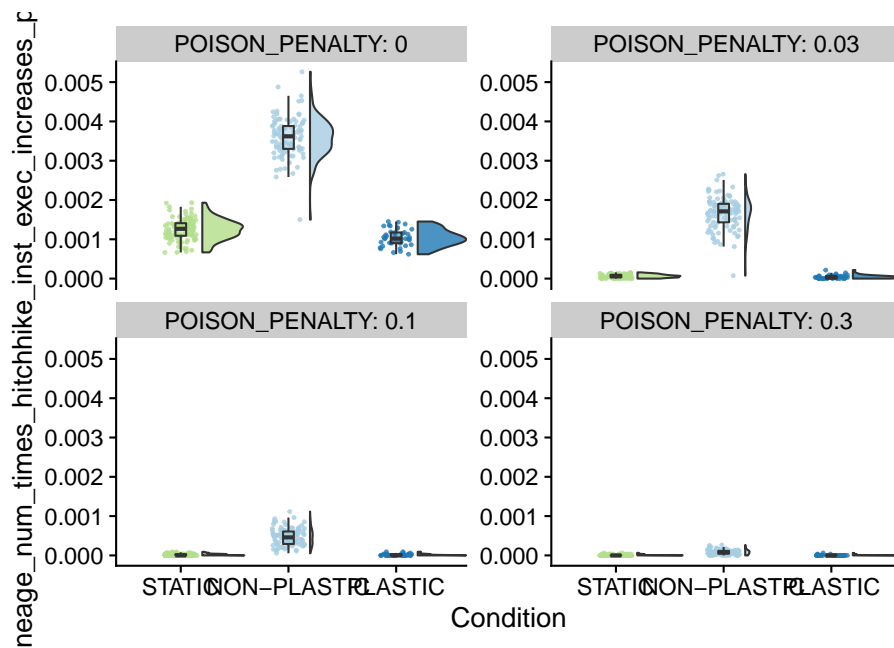
7.6.2 Frequency of increases in poison instruction execution (lineage)

```
ggplot(summary_data, aes(x=condition, y=dominant_lineage_num_times_hitchhike_inst_exec_increases_
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  )
```

```

) +
scale_fill_brewer(
  palette=cb_palette
) +
scale_color_brewer(
  palette=cb_palette
) +
facet_wrap(
  ~POISON_PENALTY,
  labeller=label_both,
  scales="free_y"
) +
# coord_flip() +
theme(
  legend.position="none"
) +
ggsave(
  paste0(working_directory, "plots/final-dominant-lineage-poison-increase-per-genera
  width=15,
  height=10
)

```



```

for (penalty in poison_penalties) {
  stat_data <- filter(summary_data, POISON_PENALTY==penalty)
  print(

```

```

    paste0(
      "PENALTY: ", penalty
    )
  )
  kt <- kruskal.test(
    formula=dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation~condition,
    data=stat_data
  )
  print(
    kt
  )
  if (is.na(kt$p.value)) { next }
  if (kt$p.value > 0.05) { next }
  print(
    pairwise.wilcox.test(
      x=stat_data$dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation,
      g=stat_data$condition,
      p.adjust.method="bonferroni"
    )
  )
}

```

```

## [1] "PENALTY: 0"
##
##   Kruskal-Wallis rank sum test
##
## data:  dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation by condition
## Kruskal-Wallis chi-squared = 180.05, df = 2, p-value < 2.2e-16
##
##
##   Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  stat_data$dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation and s
##
##           NON-PLASTIC PLASTIC
## PLASTIC < 2e-16      -
## STATIC  < 2e-16      7.8e-05
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.03"
##
##   Kruskal-Wallis rank sum test
##
## data:  dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation by condition
## Kruskal-Wallis chi-squared = 176.25, df = 2, p-value < 2.2e-16

```

```

##
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: stat_data$dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation
##
##      NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      0.019
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.1"
##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation by condition
## Kruskal-Wallis chi-squared = 184.17, df = 2, p-value < 2.2e-16
##
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: stat_data$dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation
##
##      NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      0.2
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.3"
##
## Kruskal-Wallis rank sum test
##
## data: dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation by condition
## Kruskal-Wallis chi-squared = 140.99, df = 2, p-value < 2.2e-16
##
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: stat_data$dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation
##
##      NON-PLASTIC PLASTIC
## PLASTIC 2.2e-15      -
## STATIC  < 2e-16      0.79
##
## P value adjustment method: bonferroni

```


Figure for the manuscript:

```
# Compute manual labels for geom_signif
stat.test <- focal_summary_data %>%
  wilcox_test(dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation ~ condition,
    adjust_pvalue(method = "bonferroni") %>%
    add_significance() %>%
    add_xy_position(x="condition", step.increase=0.2)
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior in g
stat.test$manual_position <- stat.test$y.position
stat.test$label <- mapply(p_label, stat.test$p.adj)

poison_increases_per_gen_fig <- ggplot(
  focal_summary_data,
  aes(x=condition, y=dominant_lineage_num_times_hitchhike_inst_exec_increases_per_generation, f
) +
  geom_flat_violin(
    scale="width",
    position = position_nudge(x = .2, y = 0),
    alpha = .8
) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order,
    labels=condition_order
) +
  scale_y_continuous(
    name="Poisonous task acquisition frequency",
) +
  scale_fill_brewer(
    palette=cb_palette
) +
  scale_color_brewer(
    palette=cb_palette
) +
```

```

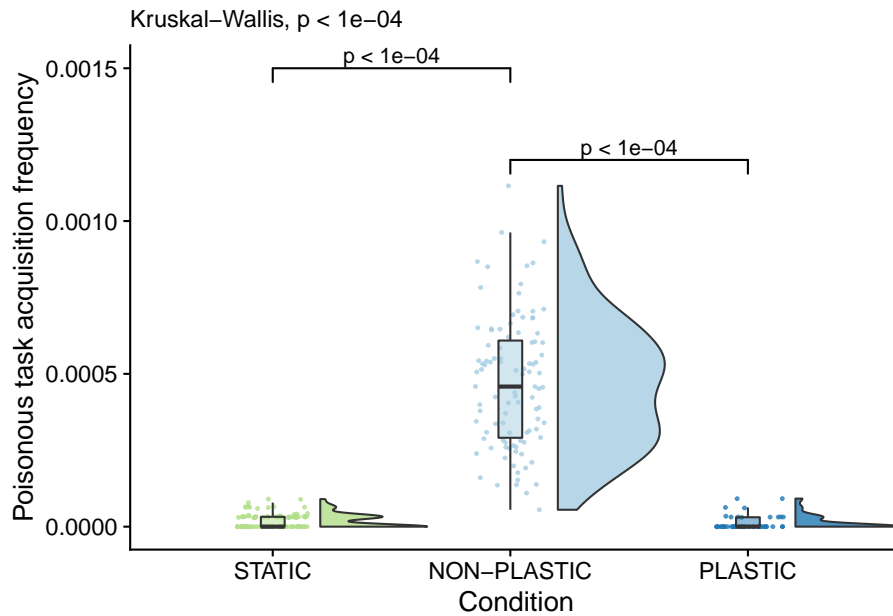
# coord_flip()
labs(
  subtitle=paste0(
    "Kruskal-Wallis, ",
    p_label(signif(kruskal.test(formula=dominant_lineage_num_times_hitchhike_inst_ex
  )
) +
ggsignif::geom_signif(
  data=filter(stat.test, p.adj <= alpha),
  aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
  manual=TRUE,
  inherit.aes=FALSE
) +
theme(
  legend.position="none"
)

```

```

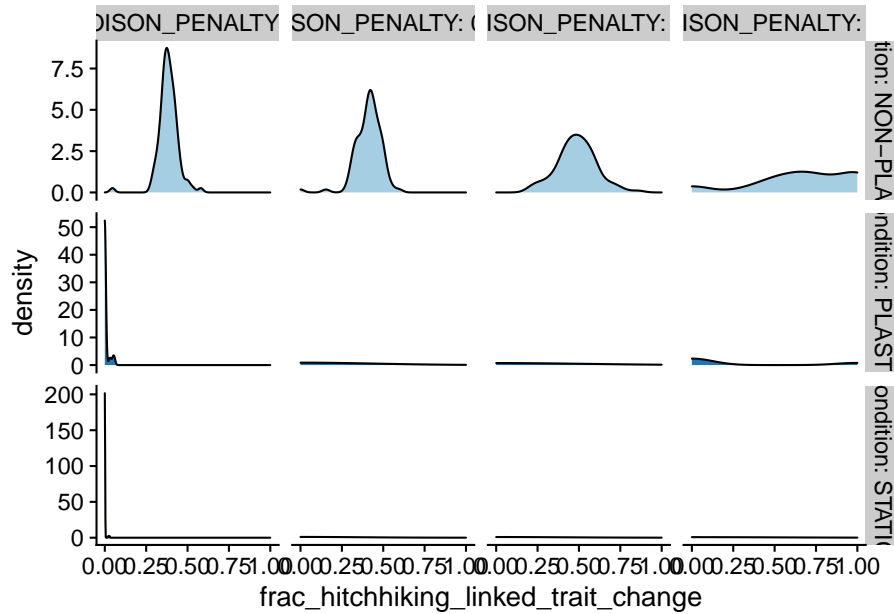
## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
poison_increases_per_gen_fig

```



7.6.3 What fraction of mutations that increase poison instruction execution co-occur with base trait changes?

```
ggplot(filter(summary_data, dominant_lineage_num_times_hitchhike_inst_exec_increases>0), aes(x=fr
  geom_density() +
  facet_grid(
    condition~POISON_PENALTY,
    labeller=label_both,
    scales="free_y"
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  theme(
    legend.position="none"
  ) +
  ggsave(
    paste0(working_directory, "plots/dominant-lineage-frac_hitchhiking_linked_trait_change.png"),
    width=15,
    height=10
  )
```

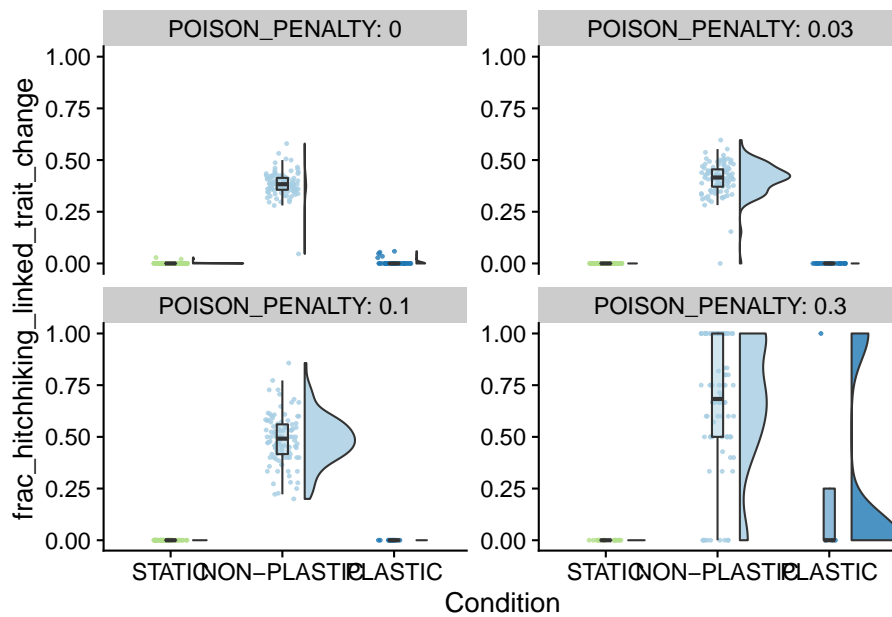


```
ggplot(filter(summary_data, dominant_lineage_num_times_hitchhike_inst_exec_increases>0)
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  )
```

```

) +
facet_wrap(
  ~POISON_PENALTY,
  labeller=label_both,
  scales="free_y"
) +
# coord_flip() +
theme(
  legend.position="none"
)

```



```

for (penalty in poison_penalties) {
  stat_data <- filter(summary_data, POISON_PENALTY==penalty & dominant_lineage_num_times_hitchhik
  print(
    paste0(
      "PENALTY: ", penalty
    )
  )
  kt <- kruskal.test(
    formula=frac_hitchhiking_linked_trait_change~condition,
    data=stat_data
  )
  print(
    kt
  )
}

```

```

if (is.na(kt$p.value)) { next }
if (kt$p.value > 0.05) { next }
print(
  pairwise.wilcox.test(
    x=stat_data$frac_hitchhiking_linked_trait_change,
    g=stat_data$condition,
    p.adjust.method="bonferroni",
    exact=FALSE
  )
)
}

```

```

## [1] "PENALTY: 0"
##
##   Kruskal-Wallis rank sum test
##
## data:  frac_hitchhiking_linked_trait_change by condition
## Kruskal-Wallis chi-squared = 211.29, df = 2, p-value < 2.2e-16
##
##
##   Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  stat_data$frac_hitchhiking_linked_trait_change and stat_data$condition
##
##           NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      0.031
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.03"
##
##   Kruskal-Wallis rank sum test
##
## data:  frac_hitchhiking_linked_trait_change by condition
## Kruskal-Wallis chi-squared = 186.88, df = 2, p-value < 2.2e-16
##
##
##   Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  stat_data$frac_hitchhiking_linked_trait_change and stat_data$condition
##
##           NON-PLASTIC PLASTIC
## PLASTIC 2.9e-16      -
## STATIC  < 2e-16      -
##

```

7.6. CHARACTERIZING MUTATIONS THAT INCREASE POISON INSTRUCTION EXECUTION143

```
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.1"
##
## Kruskal-Wallis rank sum test
##
## data: frac_hitchhiking_linked_trait_change by condition
## Kruskal-Wallis chi-squared = 113.72, df = 2, p-value < 2.2e-16
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: stat_data$frac_hitchhiking_linked_trait_change and stat_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC 3.3e-08      -
## STATIC  < 2e-16      -
##
## P value adjustment method: bonferroni
## [1] "PENALTY: 0.3"
##
## Kruskal-Wallis rank sum test
##
## data: frac_hitchhiking_linked_trait_change by condition
## Kruskal-Wallis chi-squared = 34.791, df = 2, p-value = 2.788e-08
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: stat_data$frac_hitchhiking_linked_trait_change and stat_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC 0.26          -
## STATIC  2.4e-08      0.18
##
## P value adjustment method: bonferroni
denom <- sum(filter(summary_data, condition=="NON-PLASTIC" & POISON_PENALTY==0.1)$dominant_lineage_num)
num <- sum(filter(summary_data, condition=="NON-PLASTIC" & POISON_PENALTY==0.1)$dominant_lineage_num)
paste0("NON-PLASTIC (0.1 penalty): ", num/denom, " (", num, "/", denom, ")")

## [1] "NON-PLASTIC (0.1 penalty): 0.498956158663883(956/1916)"
denom <- sum(filter(summary_data, condition=="PLASTIC" & POISON_PENALTY==0.1)$dominant_lineage_num)
num <- sum(filter(summary_data, condition=="PLASTIC" & POISON_PENALTY==0.1)$dominant_lineage_num)
paste0("PLASTIC (0.1 penalty): ", num/denom, " (", num, "/", denom, ")")

## [1] "PLASTIC (0.1 penalty): 0 (0/18)"
```

```
denom <- sum(filter(summary_data, condition=="STATIC" & POISON_PENALTY==0.1)$dominant_lineage_num_times_hitchhike_inst_exec_increases)
num <- sum(filter(summary_data, condition=="STATIC" & POISON_PENALTY==0.1)$dominant_lineage_num_times_hitchhike_inst_exec_increases)
paste0("STATIC (0.1 penalty): ", num/denom, " (", num, "/", denom, ")")
```

```
## [1] "STATIC (0.1 penalty): 0 (0/58)"
```

Focal figure for the manuscript:

```
# Compute manual labels for geom_signif
stat.test <- filter(focal_summary_data, dominant_lineage_num_times_hitchhike_inst_exec_increases)
wilcox_test(frac_hitchhiking_linked_trait_change ~ condition, comparisons=list(c("PL", "NL")))
adjust_pvalue(method = "bonferroni") %>%
add_significance() %>%
add_xy_position(x="condition")
# Tweak y.position manually to account for scaled axis (edge case that triggers bad behavior)
stat.test$manual_position <- stat.test$y.position
stat.test$label <- mapply(p_label, stat.test$p.adj)

linked_trait_change_fig <- ggplot(
  filter(focal_summary_data, dominant_lineage_num_times_hitchhike_inst_exec_increases)
  aes(x=condition, y=frac_hitchhiking_linked_trait_change, fill=condition)
) +
geom_flat_violin(
  scale="width",
  position = position_nudge(x = .2, y = 0),
  alpha = .8
) +
geom_point(
  mapping=aes(color=condition),
  position = position_jitter(width = .15),
  size = .5,
  alpha = 0.8
) +
geom_boxplot(
  width = .1,
  outlier.shape = NA,
  alpha = 0.5
) +
scale_x_discrete(
  name="Condition",
  limits=condition_order,
  labels=condition_order
) +
scale_y_continuous(
  name="Fraction of linked poisonous task acquisition",
  limits=c(-0.01, 1.2),
```



```

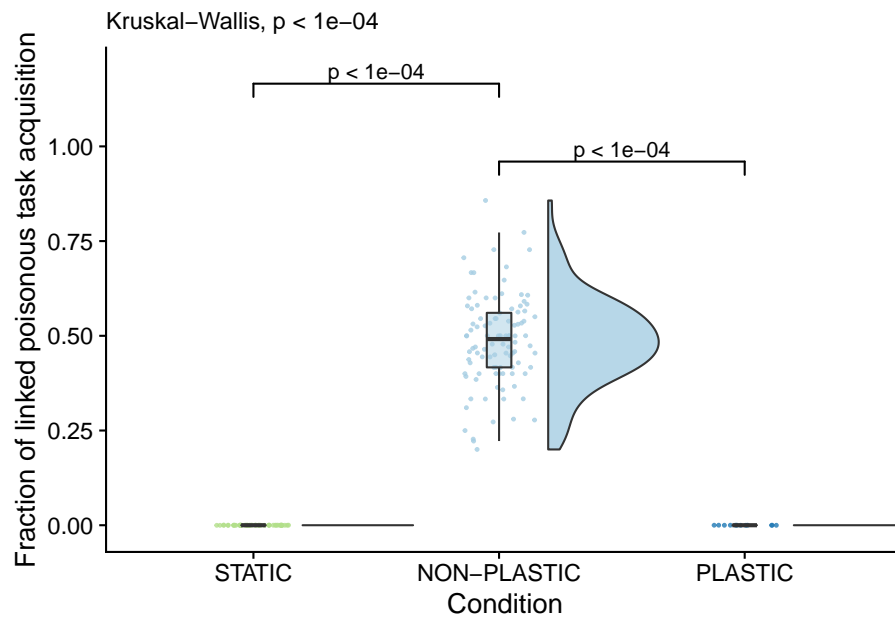
    breaks=c(0, 0.25, 0.50, 0.75, 1.0)
  ) +
  scale_fill_brewer(
    palette=cb_palette
  ) +
  scale_color_brewer(
    palette=cb_palette
  ) +
  labs(
    subtitle=paste0(
      "Kruskal-Wallis, ",
      p_label(signif(kruskal.test(formula=frac_hitchhiking_linked_trait_change~condition, data=fo
    )
  ) +
  ggsignif::geom_signif(
    data=filter(stat.test, p.adj <= alpha),
    aes(xmin=group1,xmax=group2,annotations=label,y_position=manual_position),
    manual=TRUE,
    inherit.aes=FALSE
  ) +
  theme(
    legend.position="none"
  )

```

```

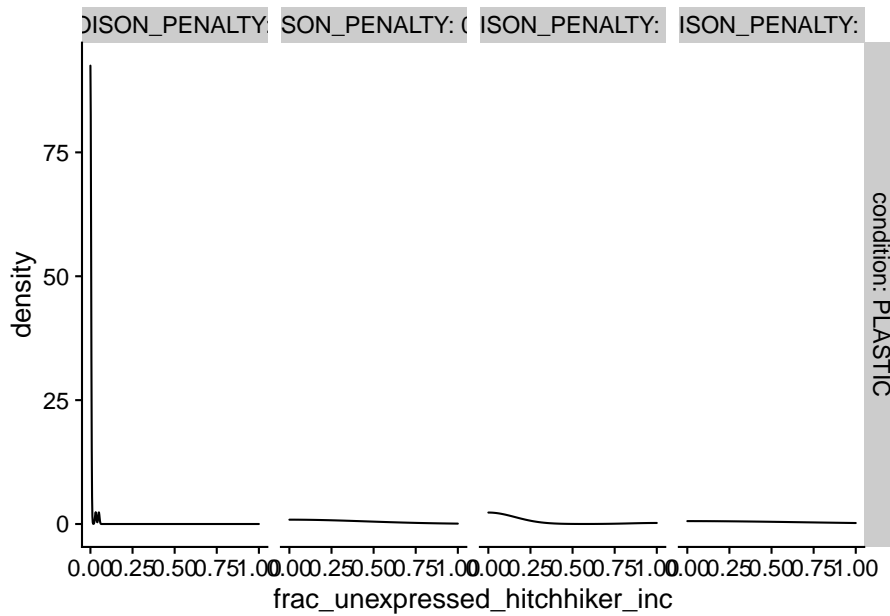
## Warning: Ignoring unknown aesthetics: xmin, xmax, annotations, y_position
linked_trait_change_fig

```



7.7 What fraction of poison execution increases

```
ggplot(filter(summary_data, dominant_lineage_num_times_hitchhike_inst_exec_increases > 0)) +
  geom_density() +
  facet_grid(
    condition ~ POISON_PENALTY,
    labeller = label_both,
    scales = "free_y"
  ) +
  theme(
    legend.position = "none"
  )
```



```
denom <- sum(filter(summary_data, condition=="PLASTIC" & POISON_PENALTY==0.1)$dominant_lineage_num)
num <- sum(filter(summary_data, condition=="PLASTIC" & POISON_PENALTY==0.1)$dominant_lineage_num)
paste0("PLASTIC: ", num/denom, " (", num, "/", denom, ")")
```

```
## [1] "PLASTIC: 0.0555555555555556 (1/18)"
```

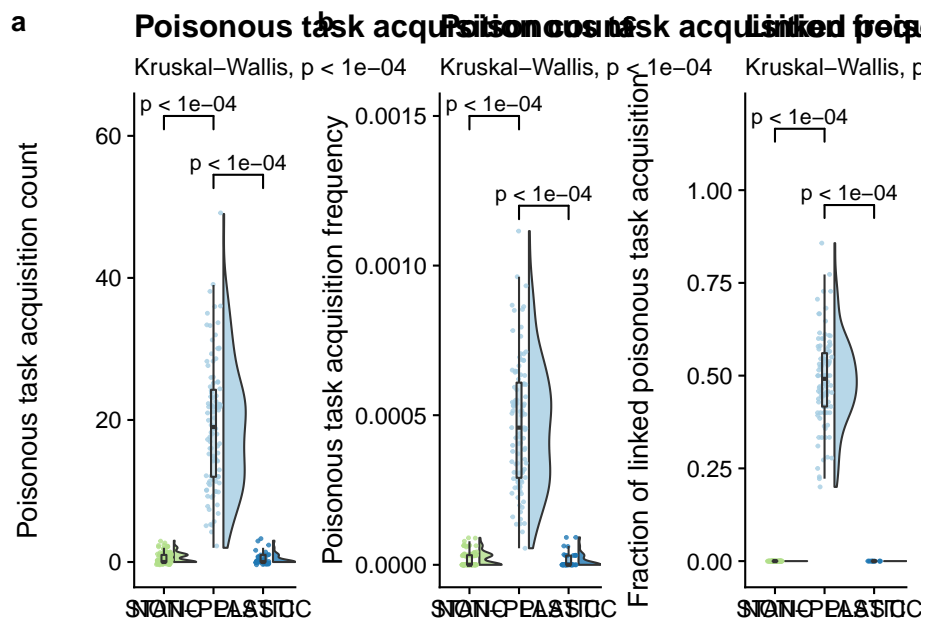
7.8 Manuscript figures

```
grid <- plot_grid(
  poison_increases_fig +
    theme(
      axis.title.x=element_blank()
    ) +
    ggtitle("Poisonous task acquisition count"),
  poison_increases_per_gen_fig +
    theme(
      axis.title.x=element_blank()
    ) +
    ggtitle("Poisonous task acquisition frequency"),
  linked_trait_change_fig +
    theme(
      axis.title.x=element_blank()
    ) +
    ggtitle("Linked poisonous task acquisition"),
```

```

nrow=1,
align="v",
labels="auto"
)
save_plot(
  paste0(working_directory, "plots/", "poison-accumulation-panel.pdf"),
  grid,
  base_height=6,
  base_asp=3/1
)
grid

```



Chapter 8

Regulation in Avida

8.1 Overview

```
total_updates <- 200000
replicates <- 100

all_traits <- c("not","nand","and","ornot","or","andnot")
traits_set_a <- c("not", "and", "or")
traits_set_b <- c("nand", "ornot", "andnot")

# Relative location of data.
working_directory <- "experiments/2021-02-08-evo-dynamics/analysis/" # << For bookdown
# working_directory <- "./" # << For local analysis
```

8.2 Analysis dependencies

Load all required R libraries.

```
library(ggplot2)
library(tidyverse)
library(cowplot)
library(RColorBrewer)
library(Hmisc)
library(boot)
source("https://gist.githubusercontent.com/benmarwick/2a1bb0133ff568cbe28d/raw/fb53bd97121f7f9ce9")
```

These analyses were conducted/knitted with the following computing environment:

```
print(version)

##
## platform      x86_64-pc-linux-gnu
## arch          x86_64
## os            linux-gnu
## system        x86_64, linux-gnu
## status
## major         4
## minor         1.0
## year          2021
## month         05
## day           18
## svn rev       80317
## language      R
## version.string R version 4.1.0 (2021-05-18)
## nickname      Camp Pontanezen
```

8.3 Setup

```
trace_summary_data_loc <- paste0(working_directory, "data/trace_summary.csv")
trace_summary_data <- read.csv(trace_summary_data_loc, na.strings="NONE")

trace_summary_data$DISABLE_REACTION_SENSORS <- as.factor(trace_summary_data$DISABLE_RE
trace_summary_data$chg_env <- trace_summary_data$chg_env == "True"
trace_summary_data$sensors <- trace_summary_data$DISABLE_REACTION_SENSORS == "0"

env_label_fun <- function(chg_env) {
  if (chg_env) {
    return("Fluctuating")
  } else {
    return("Constant")
  }
}

sensors_label_fun <- function(has_sensors) {
  if (has_sensors) {
    return("Sensors")
  } else {
    return("No sensors")
  }
}
```

8.4. HOW MANY INSTRUCTIONS DO PLASTIC GENOMES TOGGLE DEPENDING ON ENVIRONMENTAL C

```
# note that this labeler makes assumptions about how we set up our experiment
condition_label_fun <- function(has_sensors, env_chg) {
  if (has_sensors && env_chg) {
    return("PLASTIC")
  } else if (env_chg) {
    return("NON-PLASTIC")
  } else {
    return("STATIC")
  }
}

trace_summary_data$env_label <- mapply(
  env_label_fun,
  trace_summary_data$chg_env
)
trace_summary_data$sensors_label <- mapply(
  sensors_label_fun,
  trace_summary_data$sensors
)
trace_summary_data$condition <- mapply(
  condition_label_fun,
  trace_summary_data$sensors,
  trace_summary_data$chg_env
)

##### misc #####
# Configure our default graphing theme
theme_set(theme_cowplot())
dir.create(paste0(working_directory, "plots"), showWarnings=FALSE)
```

8.4 How many instructions do plastic genomes toggle depending on environmental context?

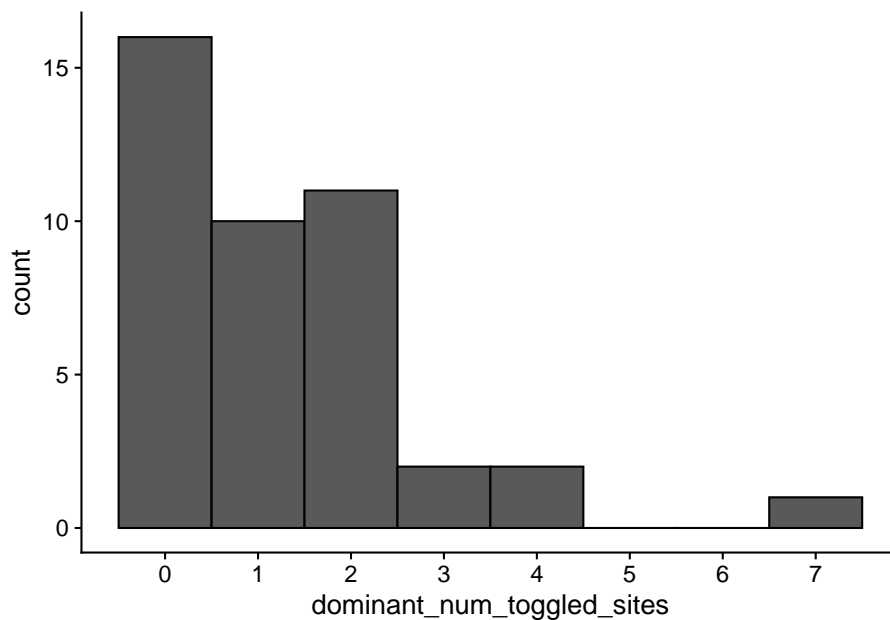
```
ggplot(trace_summary_data, aes(x=dominant_num_toggled_sites)) +
  geom_histogram(
    binwidth=1,
    color="black"
  ) +
  scale_fill_brewer(
    palette="Paired"
  ) +
  scale_color_brewer(
    palette="Paired"
```

```

) +
scale_x_continuous(
  breaks=seq(0, max(trace_summary_data$dominant_num_toggled_sites)+1)
) +
theme(
  legend.position="none"
) +
ggsave(paste0(working_directory, "plots/", "toggled-sites.png"))

## Saving 6.5 x 4.5 in image

```



8.5 What is the distribution of toggled sequence sizes?

```

chunk_sizes <- data.frame(
  size=integer()
)
for (sizes in trace_summary_data$dominant_toggled_chunk_sizes) {
  if (sizes == "") { next }
  sizes <- unlist(lapply(str_split(sizes, ';'), as.integer))
  chunk_sizes <- rbind(chunk_sizes, data.frame(size=c(sizes)))
}

```

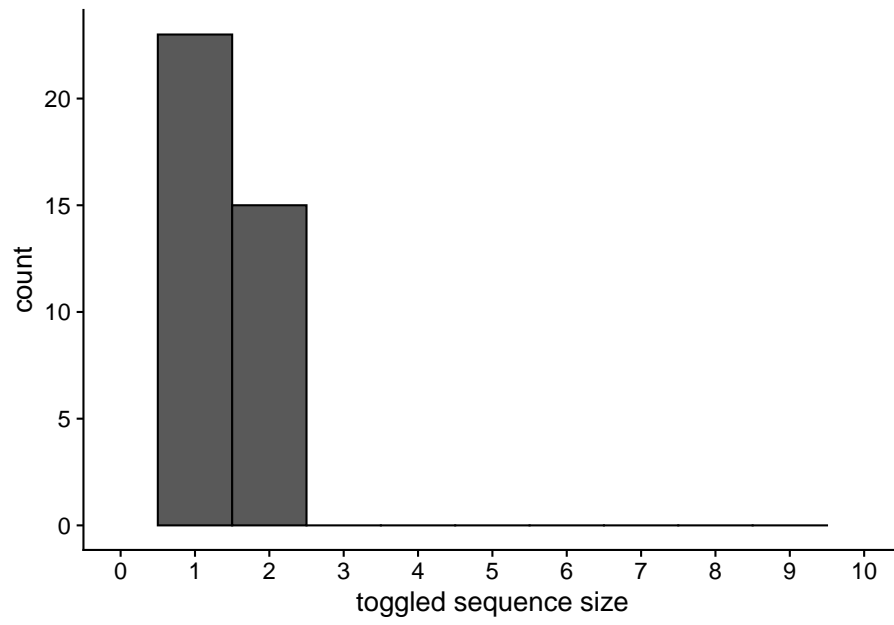

8.5. WHAT IS THE DISTRIBUTION OF TOGGLED SEQUENCE SIZES?153

```
ggplot(chunk_sizes, aes(x=size)) +  
  geom_histogram(  
    binwidth=1,  
    color="black"  
  ) +  
  scale_fill_brewer(  
    palette="Paired"  
  ) +  
  scale_color_brewer(  
    palette="Paired"  
  ) +  
  scale_x_continuous(  
    name="toggled sequence size",  
    breaks=seq(0, 10),  
    limits=c(0, 10)  
  ) +  
  theme(  
    legend.position="none"  
  ) +  
  ggsave(paste0(working_directory, "plots/", "toggled-chunk-sizes.png"))
```

```
## Saving 6.5 x 4.5 in image
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



Chapter 9

Evolutionary change (variable length genomes)

9.1 Overview

```
total_updates <- 200000
replicates <- 100

all_traits <- c("not", "nand", "and", "ornot", "or", "andnot")
traits_set_a <- c("not", "and", "or")
traits_set_b <- c("nand", "ornot", "andnot")

# Relative location of data.
working_directory <- "experiments/2021-01-30-evo-dynamics/analysis/" # << For bookdown
# working_directory <- "./" # << For local analysis
```

9.2 Analysis dependencies

Load all required R libraries.

```
library(ggplot2)
library(tidyverse)
library(cowplot)
library(RColorBrewer)
library(Hmisc)
library(boot)
source("https://gist.githubusercontent.com/benmarwick/2a1bb0133ff568cbe28d/raw/fb53bd97121f7f9ce9")
```

These analyses were conducted/knitted with the following computing environ-

ment:

```
print(version)

##
## platform      x86_64-pc-linux-gnu
## arch          x86_64
## os            linux-gnu
## system        x86_64, linux-gnu
## status
## major         4
## minor         1.0
## year          2021
## month         05
## day           18
## svn rev       80317
## language      R
## version.string R version 4.1.0 (2021-05-18)
## nickname      Camp Pontanezen
```

9.3 Setup

```
summary_data_loc <- paste0(working_directory, "data/aggregate.csv")
summary_data <- read.csv(summary_data_loc, na.strings="NONE")

summary_data$DISABLE_REACTION_SENSORS <- as.factor(summary_data$DISABLE_REACTION_SENSORS)
summary_data$chg_env <- summary_data$chg_env == "True"
summary_data$dominant_plastic_odd_even <- as.factor(summary_data$dominant_plastic_odd_even)
summary_data$sensors <- summary_data$DISABLE_REACTION_SENSORS == "0"
summary_data$is_plastic <- summary_data$dominant_plastic_odd_even == "True"

env_label_fun <- function(chg_env) {
  if (chg_env) {
    return("Fluctuating")
  } else {
    return("Constant")
  }
}

sensors_label_fun <- function(has_sensors) {
  if (has_sensors) {
    return("Sensors")
  } else {
    return("No sensors")
  }
}
```

```

}

# note that this labeler makes assumptions about how we set up our experiment
condition_label_fun <- function(has_sensors, env_chg) {
  if (has_sensors && env_chg) {
    return("PLASTIC")
  } else if (env_chg) {
    return("NON-PLASTIC")
  } else {
    return("STATIC")
  }
}

summary_data$env_label <- mapply(
  env_label_fun,
  summary_data$chg_env
)
summary_data$sensors_label <- mapply(
  sensors_label_fun,
  summary_data$sensors
)
summary_data$condition <- mapply(
  condition_label_fun,
  summary_data$sensors,
  summary_data$chg_env
)

condition_order = c(
  "STATIC",
  "NON-PLASTIC",
  "PLASTIC"
)

##### misc #####
# Configure our default graphing theme
theme_set(theme_cowplot())
dir.create(paste0(working_directory, "plots"), showWarnings=FALSE)

```

9.4 Evolution of phenotypic plasticity

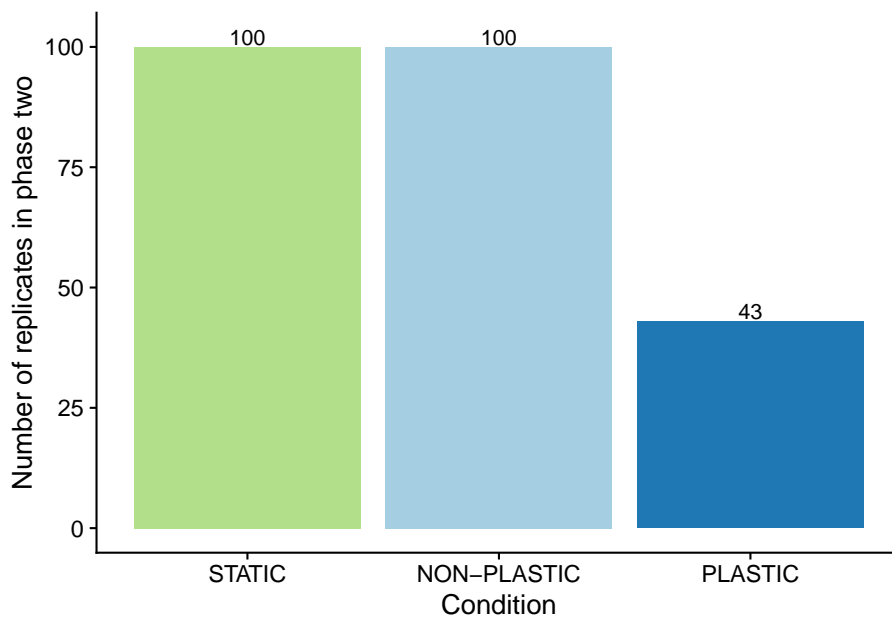
For sensor-enabled populations in fluctuating environments, we only transferred populations containing an optimally plastic genotype to phase-two.

```

summary_data_grouped = dplyr::group_by(summary_data, sensors, env_label, condition)
summary_data_group_counts = dplyr::summarize(summary_data_grouped, n=dplyr::n())

```

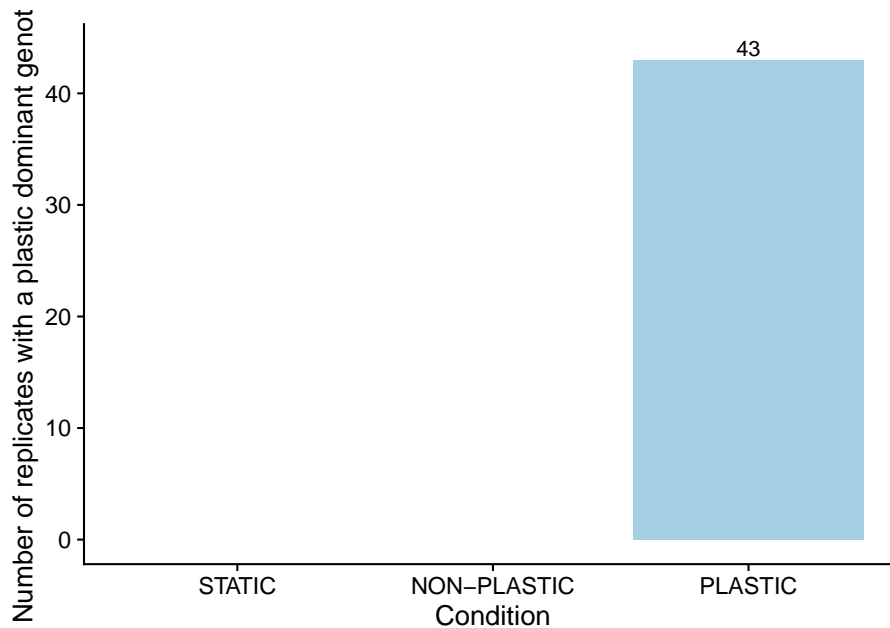
```
## `summarise()` has grouped output by 'sensors', 'env_label'. You can override using `ungroup()`
ggplot(summary_data_group_counts, aes(x=condition, y=n, fill=condition)) +
  geom_col(position=position_dodge(0.9)) +
  geom_text(aes(label=n, y=n+2)) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  scale_fill_brewer(
    palette="Paired"
  ) +
  scale_color_brewer(
    palette="Paired"
  ) +
  ylab("Number of replicates in phase two") +
  theme(
    legend.position="none"
  )
```



We can confirm our expectation that the dominant genotypes in non-plastic conditions are not phenotypically plastic.

```
summary_data_grouped = dplyr::group_by(summary_data, condition, is_plastic)
summary_data_group_counts = dplyr::summarize(summary_data_grouped, n=dplyr::n())
```

```
ggplot(filter(summary_data_group_counts, is_plastic), aes(x=condition, y=n, fill=condition)) +  
  geom_col(position=position_dodge(0.9)) +  
  scale_x_discrete(  
    name="Condition",  
    limits=condition_order  
  ) +  
  scale_fill_brewer(  
    palette="Paired"  
  ) +  
  scale_color_brewer(  
    palette="Paired"  
  ) +  
  geom_text(aes(label=n, y=n+1)) +  
  ylab("Number of replicates with a plastic dominant genotype") +  
  theme(  
    legend.position="none"  
  )  
)
```



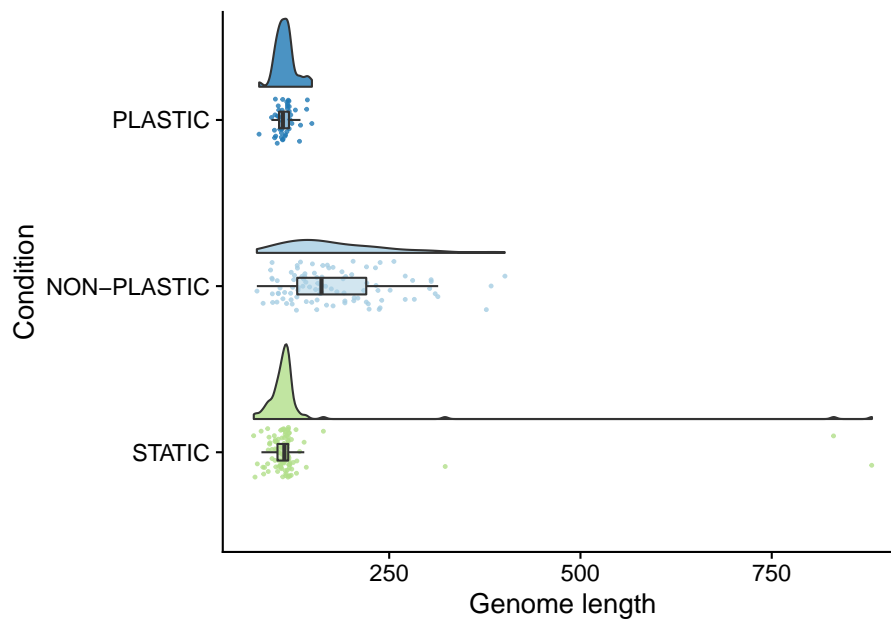
9.5 Genome length

Single-instruction insertions and deletions were possible for this experiment, so genome size also evolved.

```

ggplot(summary_data, aes(x=condition, y=dominant_genome_length, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  scale_fill_brewer(
    palette="Paired"
  ) +
  scale_color_brewer(
    palette="Paired"
  ) +
  coord_flip() +
  ylab("Genome length") +
  theme(
    legend.position="none"
  )

```

```
kruskal.test(
  formula=dominant_genome_length~condition,
  data=summary_data
)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: dominant_genome_length by condition
## Kruskal-Wallis chi-squared = 82.798, df = 2, p-value < 2.2e-16
```

```
pairwise.wilcox.test(
  x=summary_data$dominant_genome_length,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$dominant_genome_length and summary_data$condition
##
##      NON-PLASTIC PLASTIC
## PLASTIC 1.8e-10      -
## STATIC  < 2e-16      1
##
```

```
## P value adjustment method: bonferroni
median(filter(summary_data, condition=="PLASTIC")$phylo_mrca_changes)

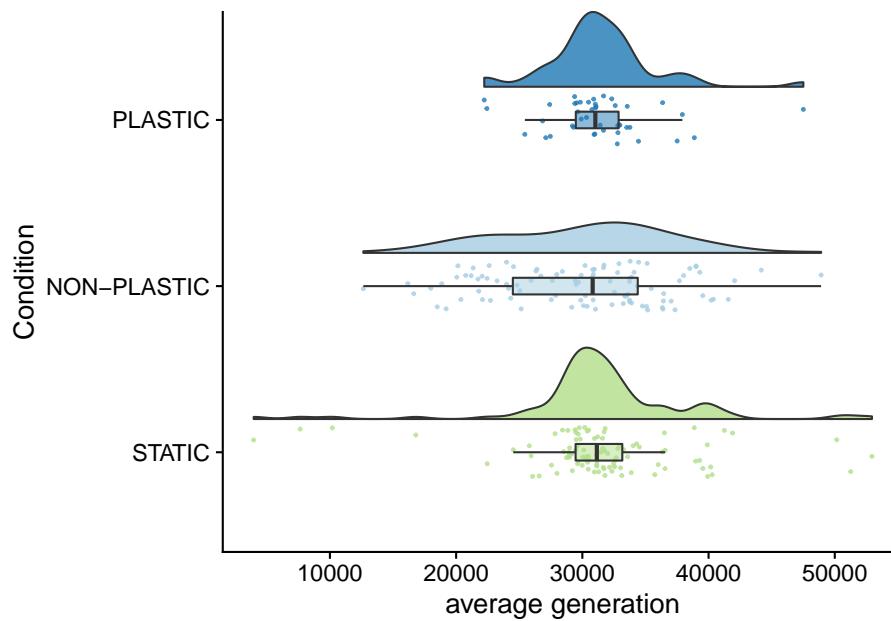
## [1] 45
median(filter(summary_data, condition=="STATIC")$phylo_mrca_changes)

## [1] 47
median(filter(summary_data, condition=="NON-PLASTIC")$phylo_mrca_changes)

## [1] 393
```

9.6 Average generation

```
ggplot(summary_data, aes(x=condition, y=time_average_generation, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  scale_fill_brewer(
    palette="Paired"
  ) +
  scale_color_brewer(
    palette="Paired"
  ) +
  coord_flip() +
  ylab("average generation") +
  theme(
    legend.position="none"
  )
```



```

median(filter(summary_data, condition=="PLASTIC")$time_average_generation)

## [1] 31028.6
median(filter(summary_data, condition=="STATIC")$time_average_generation)

## [1] 31147.5
median(filter(summary_data, condition=="NON-PLASTIC")$time_average_generation)

## [1] 30817.95
kruskal.test(
  formula=time_average_generation~condition,
  data=summary_data
)

##
##  Kruskal-Wallis rank sum test
##
## data:  time_average_generation by condition
## Kruskal-Wallis chi-squared = 1.3804, df = 2, p-value = 0.5015

```

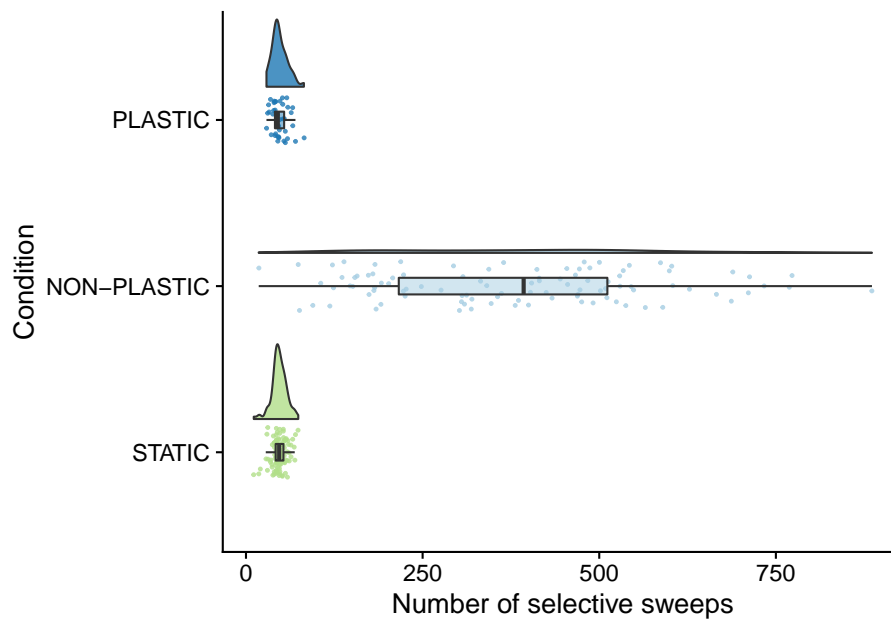
9.7 Coalescence events

The number of times the most recent common ancestor changes gives us the number of selective sweeps that occur during the experiment.

```

ggplot(summary_data, aes(x=condition, y=phylo_mrca_changes, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_fill_brewer(
    palette="Paired"
  ) +
  scale_color_brewer(
    palette="Paired"
  ) +
  coord_flip() +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
  ylab("Number of selective sweeps") +
  theme(
    legend.position="none"
  )

```



```
paste0(
  "PLASTIC: ",
  median(filter(summary_data, condition=="PLASTIC")$phylo_mrca_changes)
)

## [1] "PLASTIC: 45"

paste0(
  "STATIC: ",
  median(filter(summary_data, condition=="STATIC")$phylo_mrca_changes)
)

## [1] "STATIC: 47"

paste0(
  "NON-PLASTIC: ",
  median(filter(summary_data, condition=="NON-PLASTIC")$phylo_mrca_changes)
)

## [1] "NON-PLASTIC: 393"

kruskal.test(
  formula=phylo_mrca_changes~condition,
  data=summary_data
)

##
## Kruskal-Wallis rank sum test
```

```
##
## data: phylo_mrca_changes by condition
## Kruskal-Wallis chi-squared = 168.89, df = 2, p-value < 2.2e-16

pairwise.wilcox.test(
  x=summary_data$phylo_mrca_changes,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)

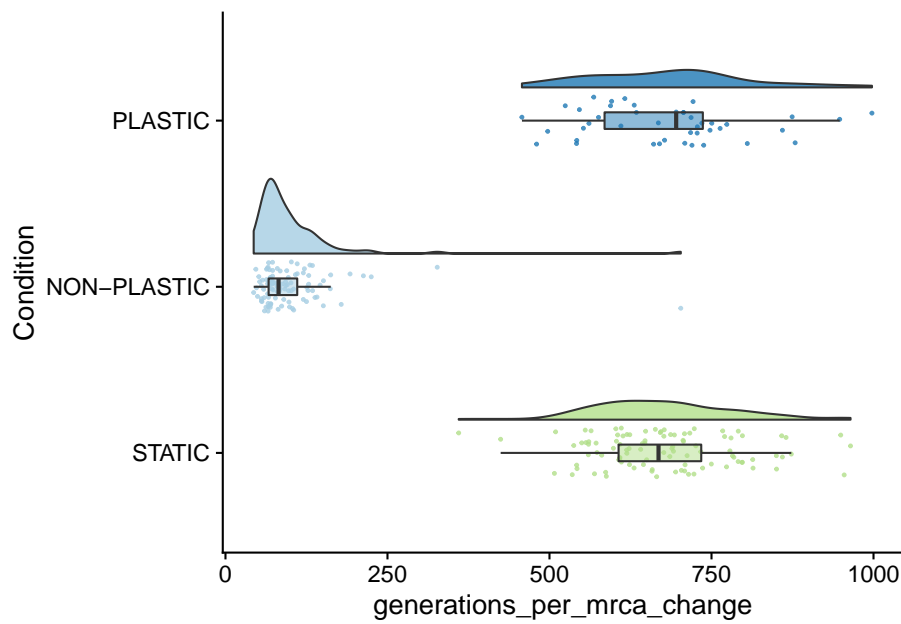
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$phylo_mrca_changes and summary_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      1
##
## P value adjustment method: bonferroni
```

9.7.1 Average number of generations between selective sweeps

```
summary_data$generations_per_mrca_change <- summary_data$time_average_generation / sum

ggplot(summary_data, aes(x=condition, y=generations_per_mrca_change, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
```

```
scale_fill_brewer(
  palette="Paired"
) +
scale_color_brewer(
  palette="Paired"
) +
coord_flip() +
theme(
  legend.position="none"
)
```



```
paste0(
  "PLASTIC: ",
  median(filter(summary_data, condition=="PLASTIC"))$generations_per_mrca_change)
)
```

```
## [1] "PLASTIC: 695.504761904762"
```

```
paste0(
  "STATIC: ",
  median(filter(summary_data, condition=="STATIC"))$generations_per_mrca_change)
)
```

```
## [1] "STATIC: 668.25523255814"
```

```

paste0(
  "NON-PLASTIC: ",
  median(filter(summary_data, condition=="NON-PLASTIC")$generations_per_mrca_change)
)

## [1] "NON-PLASTIC: 81.9208459944751"

kruskal.test(
  formula=generations_per_mrca_change~condition,
  data=summary_data
)

##
## Kruskal-Wallis rank sum test
##
## data: generations_per_mrca_change by condition
## Kruskal-Wallis chi-squared = 171.73, df = 2, p-value < 2.2e-16

pairwise.wilcox.test(
  x=summary_data$generations_per_mrca_change,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$generations_per_mrca_change and summary_data$condition
##
##      NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      1
##
## P value adjustment method: bonferroni

```

9.8 Phenotypic volatility along the dominant lineage

```

ggplot(summary_data, aes(x=condition, y=dominant_lineage_trait_volatility, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),

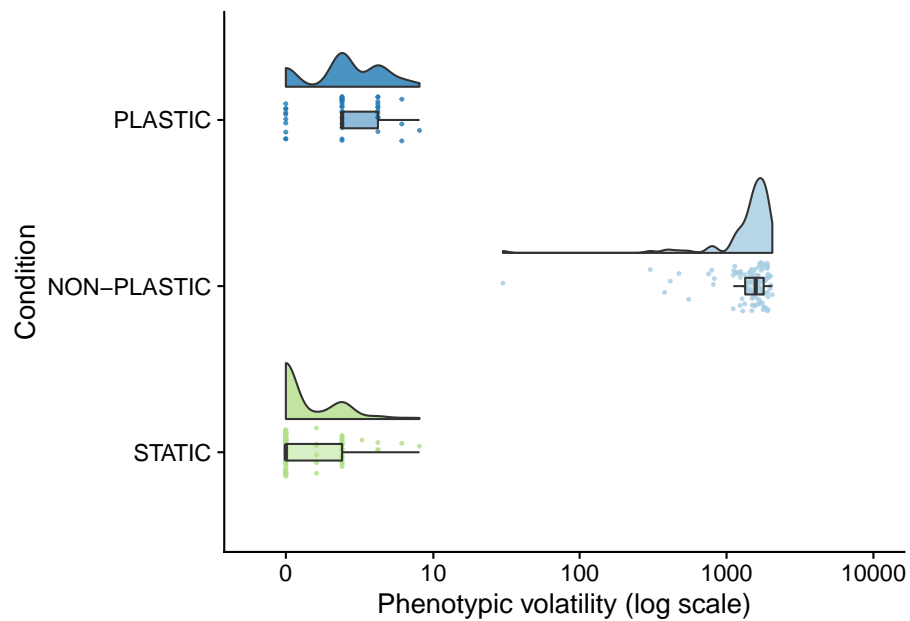
```



```

    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
) +
geom_boxplot(
  width = .1,
  outlier.shape = NA,
  alpha = 0.5
) +
scale_x_discrete(
  name="Condition",
  limits=condition_order
) +
scale_y_continuous(
  name="Phenotypic volatility (log scale)",
  trans="pseudo_log",
  breaks=c(0, 10, 100, 1000, 10000),
  limits=c(-1,10000)
) +
scale_fill_brewer(
  palette="Paired"
) +
scale_color_brewer(
  palette="Paired"
) +
coord_flip() +
theme(
  legend.position="none"
)

```



```
paste0(
  "PLASTIC: ",
  median(filter(summary_data, condition=="PLASTIC")$dominant_lineage_trait_volatility)
)
```

```
## [1] "PLASTIC: 2"
```

```
paste0(
  "STATIC: ",
  median(filter(summary_data, condition=="STATIC")$dominant_lineage_trait_volatility)
)
```

```
## [1] "STATIC: 0"
```

```
paste0(
  "NON-PLASTIC: ",
  median(filter(summary_data, condition=="NON-PLASTIC")$dominant_lineage_trait_volatility)
)
```

```
## [1] "NON-PLASTIC: 1580"
```

```
kruskal.test(
  formula=dominant_lineage_trait_volatility~condition,
  data=summary_data
)
```

```
##
```

```
## Kruskal-Wallis rank sum test
```

9.9. MUTATION ACCUMULATION ALONG THE DOMINANT LINEAGE171

```
##
## data: dominant_lineage_trait_volatility by condition
## Kruskal-Wallis chi-squared = 191.98, df = 2, p-value < 2.2e-16
pairwise.wilcox.test(
  x=summary_data$dominant_lineage_trait_volatility,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: summary_data$dominant_lineage_trait_volatility and summary_data$condition
##
##          NON-PLASTIC PLASTIC
## PLASTIC < 2e-16      -
## STATIC  < 2e-16      5.2e-08
##
## P value adjustment method: bonferroni
```

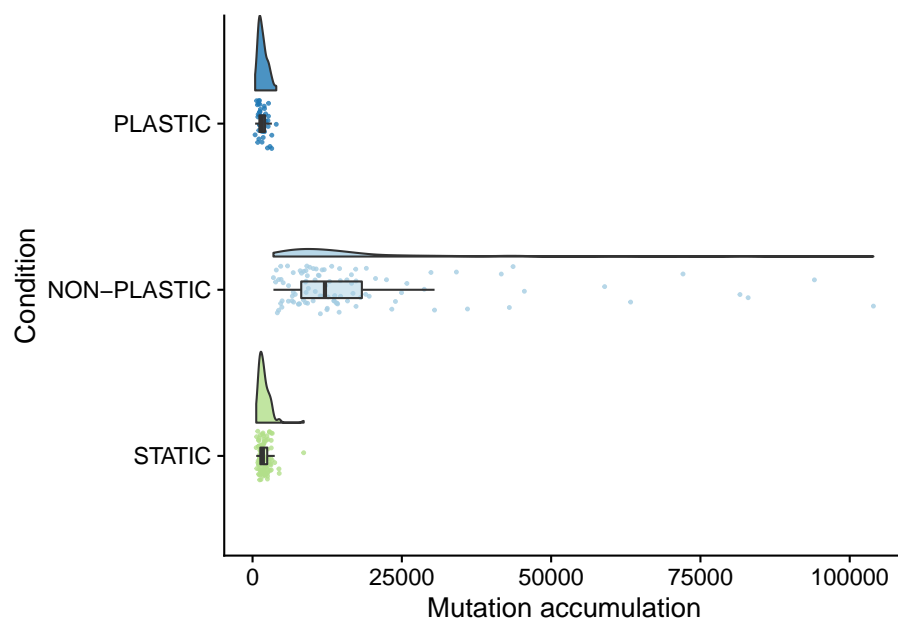
9.9 Mutation accumulation along the dominant lineage

```
ggplot(summary_data, aes(x=condition, y=dominant_lineage_total_mut_cnt, fill=condition)) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8
  ) +
  geom_point(
    mapping=aes(color=condition),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  ylab("Mutation accumulation") +
  scale_x_discrete(
    name="Condition",
    limits=condition_order
  ) +
```

```

scale_fill_brewer(
  palette="Paired"
) +
scale_color_brewer(
  palette="Paired"
) +
coord_flip() +
theme(
  legend.position="none"
)

```



```

paste0(
  "PLASTIC: ",
  median(filter(summary_data, condition=="PLASTIC"))$dominant_lineage_total_mut_cnt)
)

```

```
## [1] "PLASTIC: 1552"
```

```

paste0(
  "STATIC: ",
  median(filter(summary_data, condition=="STATIC"))$dominant_lineage_total_mut_cnt)
)

```

```
## [1] "STATIC: 1724.5"
```

9.9. MUTATION ACCUMULATION ALONG THE DOMINANT LINEAGE173

```
paste0(
  "NON-PLASTIC: ",
  median(filter(summary_data, condition=="NON-PLASTIC")$dominant_lineage_total_mut_cnt)
)

## [1] "NON-PLASTIC: 12123"

kruskal.test(
  formula=dominant_lineage_total_mut_cnt~condition,
  data=summary_data
)

##
## Kruskal-Wallis rank sum test
##
## data:  dominant_lineage_total_mut_cnt by condition
## Kruskal-Wallis chi-squared = 174.38, df = 2, p-value < 2.2e-16

pairwise.wilcox.test(
  x=summary_data$dominant_lineage_total_mut_cnt,
  g=summary_data$condition,
  p.adjust.method="bonferroni",
)

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  summary_data$dominant_lineage_total_mut_cnt and summary_data$condition
##
##      NON-PLASTIC PLASTIC
## PLASTIC <2e-16      -
## STATIC  <2e-16      0.57
##
## P value adjustment method: bonferroni
```


Bibliography

Lalejini, A. M. and Ferguson, A. J. (2021). Data for evolutionary consequences of phenotypic plasticity.