Supplemental info about phylodiversity in evolutionary computation

Jose Guadalupe Hernandez, Alexander Lalejini, Emily Dolson

2021-09-05

Contents

Chapter 1

Introduction

This is the supplemental material for our work, 'What can phylogenetic metrics tell us about useful diversity in evolutionary algorithms?'. This is not intended as a stand-alone document, but as a companion to our paper.

1.1 About our supplemental material

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

The source code/configuration files for this supplemental material and all experiments in the paper can be found in this GitHub repository.

Our supplemental material includes the following:

TODO

1.2 Contributing authors

- Jose Guadalupe Hernandez
- Alexander Lalejini
- Emily Dolson

1.3 Research overview

1.3.1 Abstract

**It is generally accepted that "diversity" is associated with success in evolutionary algorithms. However, diversity is a broad concept

that can be measured and defined in a multitude of ways. To date, most evolutionary computation research has measured diversity using the richness and/or evenness of a particular genotypic or phenotypic property. While these metrics are informative, we hypothesize that other diversity metrics are more strongly predictive of success. Phylogenetic diversity metrics are a class of metrics popularly used in biology, which take into account the evolutionary history of a population. Here, we investigate the extent to which 1) these metrics provide different information than those traditionally used in evolutionary computation, and 2) these metrics better predict the long-term success of a run of evolutionary computation. We find that, in most cases, phylogenetic metrics behave meaningfully differently from other diversity metrics. Moreover, our results suggest that phylogenetic diversity is indeed a better predictor of success.

Chapter 2

Exploration Diagnostic

2.1 Setup

Include dependencies

```
library(ggplot2)        # For plotting
library(tidyverse)      # For data wrangling
library(knitr)          # For making nice rmarkdown documents
library(cowplot)        # For theme
library(viridis)        # For color scale
library(RColorBrewer)   # For more color scales
library(rstatix)
library(ggsignif)       # For adding pairwise significance to plots
library(Hmisc)          # For bootstrapping condifence internvals
library(kableExtra)     # For displaying nice tables
source("https://gist.githubusercontent.com/benmarwick/2a1bb0133ff568cbe28d/raw/fb53bd97121f7f9ce9
library(readr)         # For reading in data
library(stringr)       # For manipulating string data
library(ggpubr)        # For displaying correlation statistics on plots
library(infotheo)      # For causality analysis
library(scales)        # For displaying scales nicely in facetted plots
library(osfr)          # For downloading the data for this project
```

These analyses were conducted in the following computing environment:

```
print(version)
```

```
##                _
## platform       x86_64-pc-linux-gnu
## arch           x86_64
## os             linux-gnu
```

```
## system         x86_64, linux-gnu
## status
## major          4
## minor          0.4
## year           2021
## month          02
## day            15
## svn rev        80002
## language       R
## version.string R version 4.0.4 (2021-02-15)
## nickname       Lost Library Book
```

Setup constants to be used across plots

```
# Labeler for stats annotations
p_label <- function(p_value) {
  threshold = 0.0001
  if (p_value < threshold) {
    return(paste0("p < ", threshold))
  } else {
    return(paste0("p = ", p_value))
  }
}

# Significance threshold
alpha <- 0.05

# Common graph variables
performance_ylim <- 1
coverage_ylim <- 1.0

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

```
# Read in data
osf_retrieve_file("esm4r") %>% osf_download(conflicts = "skip")  # Download data from
```

```
## # A tibble: 1 x 4
##   name                  id              local_path               meta
##   <chr>                 <chr>           <chr>                    <list>
## 1 final_exploration_diag~ 612fe97aab8bca0~ ./final_exploration_diag~ <named lis~
```

```
data_loc <- "final_exploration_diagnostic_data.csv"

data <- read_csv(data_loc, na=c("NONE", "NA", ""))

## Clean up data columns
```

```
# Make selection name column human readable
data <- data %>% mutate(selection_name = as.factor(case_when(
  selection_name == "EpsilonLexicase" ~ "Lexicase",
  TOUR_SIZE == 1 ~ "Random",
  selection_name == "Tournament" ~ "Tournament",
  selection_name == "FitnessSharing" ~ "Fitness Sharing",
  selection_name == "EcoEa" ~ "EcoEA"
)))

# Calculate performance statistics.
# Elite trait avg is the avg per-site performance of the best individual
data$elite_trait_avg <-
  data$ele_agg_per / data$OBJECTIVE_CNT
data$unique_start_positions_coverage <-
  data$uni_str_pos / data$OBJECTIVE_CNT

# Convert elite_trait_avg to percent of maximum possible
data$elite_trait_avg <- data$elite_trait_avg/data$TARGET

# Grab data from just the final time point
final_data <- filter(data, evaluations==max(data$evaluations))
```

2.2 Performance

For context, it's important to know how each selection scheme performed on the exploration diagnostic.

2.2.1 Over time

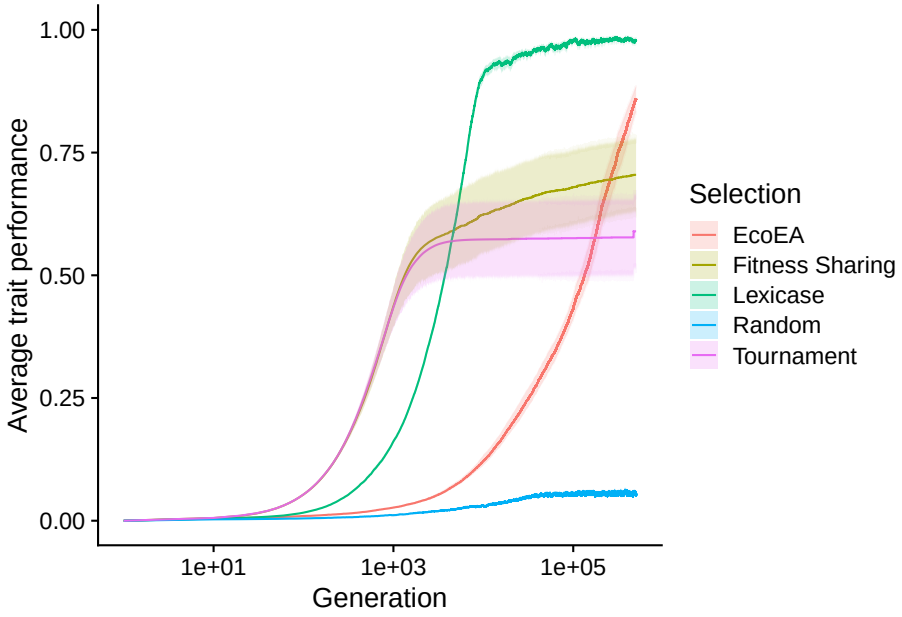
First we look at the dynamics of performance over time.

2.2.1.1 Trait performance

Plot average trait performance (i.e. fitness) over time for each selection scheme. We log the x-axis because Eco-EA gains fitness over a very long time scale, whereas the interesting dynamics for the other selection schemes occur relatively quickly.

```
ggplot(
    data,
    aes(
      x=gen,                # Generations
      y=elite_trait_avg,    # Performance
      color=selection_name, # Selection scheme
      fill=selection_name
```

```
    )
  ) +
  stat_summary(geom="line", fun=mean) + # Plot line showing mean for each selection sc
  stat_summary(  # Add shading around each line indicating 95% confiedence interval
    geom="ribbon",
    fun.data="mean_cl_boot",
    fun.args=list(conf.int=0.95),
    alpha=0.2,
    linetype=0
  ) +
  scale_y_continuous(
    name="Average trait performance", # Set y axis title
    limits=c(0, performance_ylim)  # Set y axis range to include all possible performa
  ) +
  scale_x_log10(  # Log x axis
    name="Generation" # Set x axis title
  ) +
  scale_color_discrete("Selection") + # Set legend title
  scale_fill_discrete("Selection")    # Set legend title
```



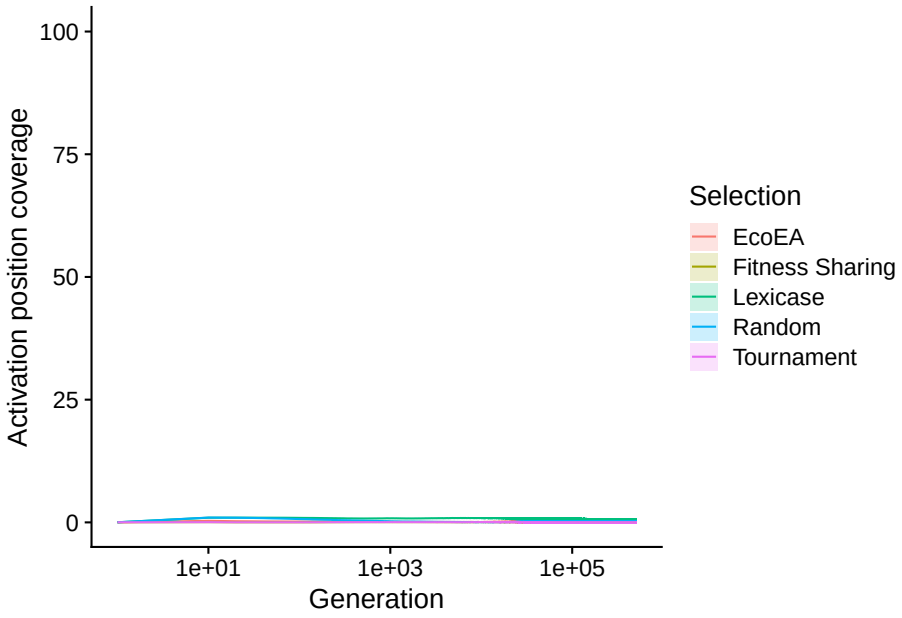
As observed by Hernandez et al. in their original paper on the exploration diagnostic, fitness in tournament selection initially increases quickly and then plateaus. Fitness in lexicase selection increases slightly slower but plateaus at a much higher value (nearly 100%). Fitness sharing behaves similarly to tournament selection, but maintains a slight upward trajectory (note that,

because the x axis is on a log scale, this trajectory is very gradual). Eco-EA takes substantially longer to increase in fitness but ultimately surpasses fitness sharing and tournament selection. It is unclear whether it would pass lexicase selection if these runs were allowed to continue for slightly longer; they do not appear to have plateaued yet. We chose to cut them off at 500,000 generations due to resource constraints and the fact that the questions we're asking here are not really about final fitness.

2.2.1.2 Activation position coverage

Out of curiosity, we also ran the analysis of unique activation positions present in the population, used by Hernandez et. al. This analysis tells us about the diversity of start positions for the coding region represented in the population. As the set of start positions in the population tends to represent a meaningful constraint on the number of paths through the fitness landscape that are currently accessible, this is in some sense a metric of useful diversity in the population

```
ggplot(data, aes(x=gen, y=unique_start_positions_coverage, color=selection_name, fill=selection_n
  stat_summary(geom="line", fun=mean) +
  stat_summary(
    geom="ribbon",
    fun.data="mean_cl_boot",
    fun.args=list(conf.int=0.95),
    alpha=0.2,
    linetype=0
  ) +
  scale_y_continuous(
    name="Activation position coverage",
    limits=c(0, 100)
  ) +
  scale_x_log10(
    name="Generation"
  ) +
  scale_color_discrete("Selection")+
  scale_fill_discrete("Selection")
```



We see that lexicase selection maintains by far that largest number of unique start positions, even surpassing the number maintained by random drift. This suggests that lexicase selection is actively selecting for maintaining a diversity of start positions. Tournament selection and fitness sharing perform virtually identically, with Eco-EA falling in between.

2.2.2 Final

While trends over time are more informative, it can be hard to visualize the full distribution (particularly the extent of variation). Thus, we also conduct a more detailed analysis of performance at the final time point.

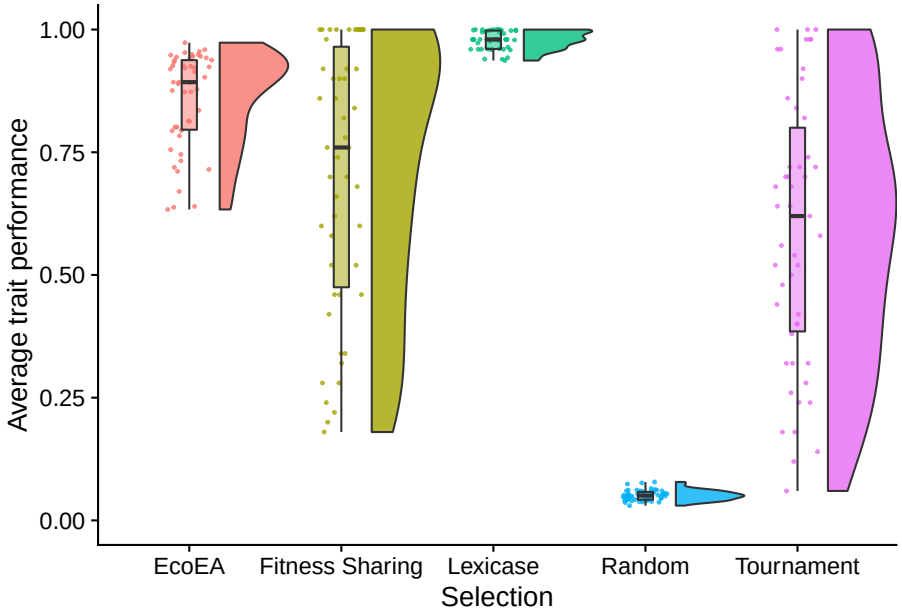
2.2.2.1 Trait performance

First we conduct statistics to identify which selection schemes are significantly different from each other.

```
# Compute manual labels for geom_signif
stat.test <- final_data %>%
  wilcox_test(elite_trait_avg ~ selection_name) %>%
  adjust_pvalue(method = "bonferroni") %>%  # Apply Bonferroni correction for multiple
  add_significance() %>%
  add_xy_position(x="selection_name",step.increase=1)
stat.test$label <- mapply(p_label,stat.test$p.adj)
```

Then we make raincloud plots of each selection scheme.

```
elite_final_performance_fig <- ggplot(
    final_data,
    aes(
      x=selection_name,
      y=elite_trait_avg,
      fill=selection_name
    )
  ) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8,
    scale="width"
  ) +
  geom_point(
    mapping=aes(color=selection_name),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_y_continuous(
    name="Average trait performance",
    limits=c(0, performance_ylim)
  ) +
  scale_x_discrete(
    name="Selection"
  ) +
  scale_fill_discrete(
    name="Selection"
  ) +
  scale_color_discrete(
    name="Selection"
  ) +
  theme(legend.position="none")
elite_final_performance_fig
```



These observations look fairly consistent with the timeseries plots.

Next, we output the results of our significance testing.

```
stat.test %>%
  kbl() %>%
  kable_styling(
    bootstrap_options = c(
      "striped",
      "hover",
      "condensed",
      "responsive"
    )
  ) %>%
  scroll_box(width="600px")
```

2.2.2.2 Final activation position Coverage

Now we do the same analysis for final activation position coverage.

First we calculate the statistics

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

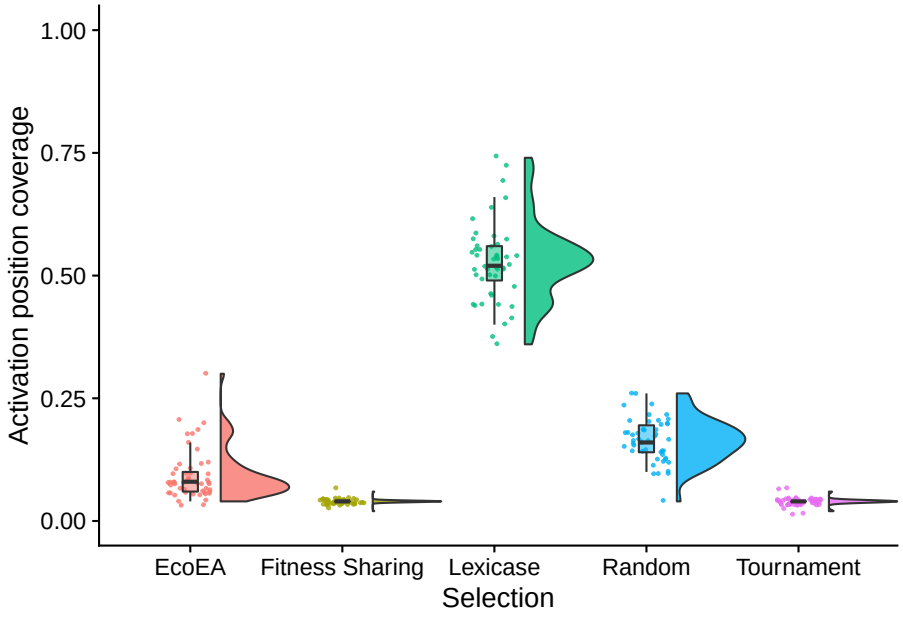
.y.	group1	group2	n1	n2	statistic	p	p.adj	p.adj.signif	y
elite_trait_avg	EcoEA	Fitness Sharing	50	50	1561	3.20e-02	3.20e-01	ns	
elite_trait_avg	EcoEA	Lexicase	50	47	60	0.00e+00	0.00e+00	****	
elite_trait_avg	EcoEA	Random	50	50	2500	0.00e+00	0.00e+00	****	
elite_trait_avg	EcoEA	Tournament	50	50	1939	2.10e-06	2.07e-05	****	
elite_trait_avg	Fitness Sharing	Lexicase	50	47	593	2.69e-05	2.69e-04	***	
elite_trait_avg	Fitness Sharing	Random	50	50	2500	0.00e+00	0.00e+00	****	
elite_trait_avg	Fitness Sharing	Tournament	50	50	1549	4.00e-02	4.00e-01	ns	
elite_trait_avg	Lexicase	Random	47	50	2350	0.00e+00	0.00e+00	****	
elite_trait_avg	Lexicase	Tournament	47	50	2098	0.00e+00	0.00e+00	****	1
elite_trait_avg	Random	Tournament	50	50	10	0.00e+00	0.00e+00	****	1

```
  add_xy_position(x="selection_name",step.increase=1)
stat.test$manual_position <- stat.test$y.position * 1.05
stat.test$label <- mapply(p_label,stat.test$p.adj)
```

Then we make raincloud plots

```
unique_start_positions_coverage_final_fig <- ggplot(
    final_data,
    aes(
      x=selection_name,
      y=unique_start_positions_coverage,
      fill=selection_name
    )
  ) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8,
    scale="width"
  ) +
  geom_point(
    mapping=aes(color=selection_name),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_y_continuous(
    name="Activation position coverage",
    limits=c(0, coverage_ylim)
```

```
  ) +
  scale_x_discrete(
    name="Selection"
  ) +
  scale_fill_discrete(
    name="Selection"
  ) +
  scale_color_discrete(
    name="Selection"
  ) +
  theme(
    legend.position="none"
  )
unique_start_positions_coverage_final_fig
```



These also look unsurprising.

Lastly, we output the results of significance testing.

```
stat.test %>%
  kbl() %>%
  kable_styling(
    bootstrap_options = c(
      "striped",
      "hover",
      "condensed",
```

.y.	group1	group2	n1	n2	statistic	p	p.adj	p.ad
unique_start_positions_coverage	EcoEA	Fitness Sharing	50	50	2392.5	0.000	0	****
unique_start_positions_coverage	EcoEA	Lexicase	50	47	0.0	0.000	0	****
unique_start_positions_coverage	EcoEA	Random	50	50	339.0	0.000	0	****
unique_start_positions_coverage	EcoEA	Tournament	50	50	2387.0	0.000	0	****
unique_start_positions_coverage	Fitness Sharing	Lexicase	50	47	0.0	0.000	0	****
unique_start_positions_coverage	Fitness Sharing	Random	50	50	25.0	0.000	0	****
unique_start_positions_coverage	Fitness Sharing	Tournament	50	50	1274.5	0.708	1	ns
unique_start_positions_coverage	Lexicase	Random	47	50	2350.0	0.000	0	****
unique_start_positions_coverage	Lexicase	Tournament	47	50	2350.0	0.000	0	****
unique_start_positions_coverage	Random	Tournament	50	50	2475.5	0.000	0	****

```
      "responsive"
    )
  ) %>%
  scroll_box(width="600px")
```

2.3 Phylogenetic diversity

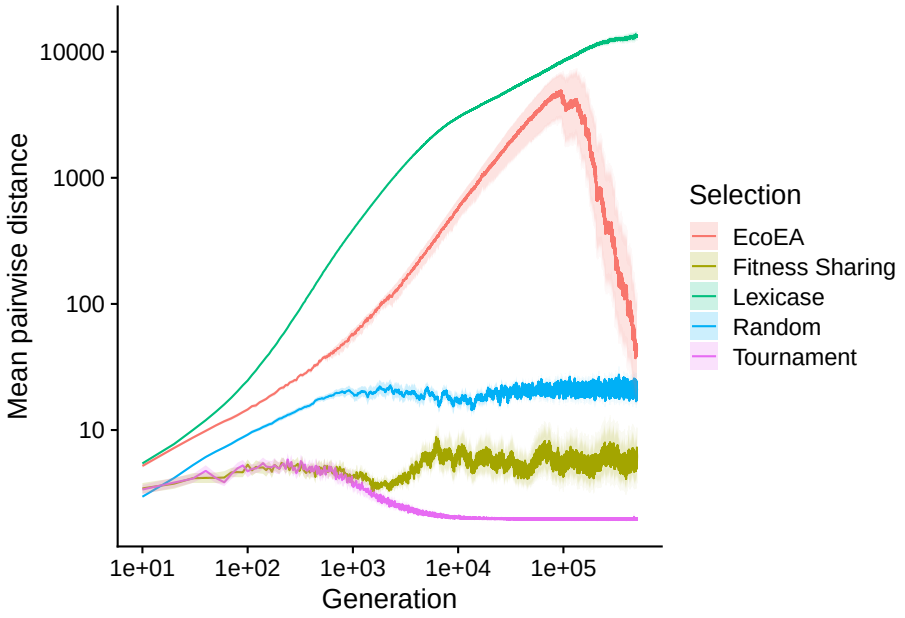
Next, we analyze the behavior of phylogenetic diversity on the exploration diagnostic.

2.3.1 Over time

First we plot mean pairwise distance over time. We log the y axis because there is such variation in mean pairwise distance across selection schemes.

```
ggplot(
    data,
    aes(
      x=gen,
      y=mean_phenotype_pairwise_distance,
      color=selection_name,
      fill=selection_name
    )
  ) +
  stat_summary(geom="line", fun=mean) +
  stat_summary(
    geom="ribbon",
    fun.data="mean_cl_boot",
    fun.args=list(conf.int=0.95),
    alpha=0.2,
    linetype=0
  ) +
```

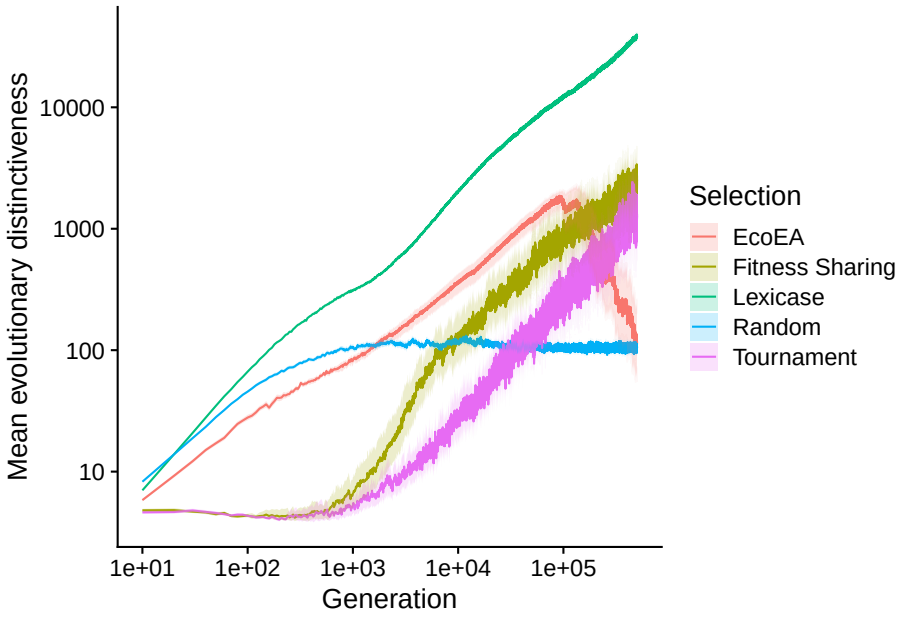
```
  scale_y_log10(
    name="Mean pairwise distance"
  ) +
  scale_x_log10(
    name="Generation"
  ) +
  scale_color_discrete("Selection") +
  scale_fill_discrete("Selection")
```



Lexicase selection maintains a monotonic increase in phylogenetic diversity over the course of the entire experiment. It likely never experiences a full coalesence event (where the most-recent common ancestor changes). Eco-EA nearly keeps pace with lexicase selection until towards the end, when its phylogenetic diversity crashes. This is likely the result of selective sweeps that begin to occur as the population discovers high fitness solutions. Fitness sharing shows a slight dip at the same time that its fitness plateaus (likely also the result of a selective sweep), but phylogenetic diversity recovers afterwards, making for a relatively constant level. over time. Tournament selection, on the other hand, maintains the same (low) level of phylogenetic diversity as fitness sharing, up until the point that fitness plateaus, at which point tournament selection's phylodiversity drops to nearly 0. Interestingly, lexicase selection and Eco-EA both maintain more phylodiversity than random selection, whereas fitness sharing and tournament selection maintain less.

For comparison, we make the same plot with mean evolutionary distinctiveness.

```
ggplot(
    data,
    aes(
      x=gen,
      y=mean_phenotype_evolutionary_distinctiveness,
      color=selection_name,
      fill=selection_name
    )
  ) +
  stat_summary(geom="line", fun=mean) +
  stat_summary(
    geom="ribbon",
    fun.data="mean_cl_boot",
    fun.args=list(conf.int=0.95),
    alpha=0.2,
    linetype=0
  ) +
  scale_y_log10(
    name="Mean evolutionary distinctiveness"
  ) +
  scale_x_log10(
    name="Generation"
  ) +
  scale_color_discrete("Selection") +
  scale_fill_discrete("Selection")
```



2.3.2 Final

Next, we perform a more in-depth analysis of phylodiversity distributions at the final time point.

2.3.2.1 Mean pairwise distance

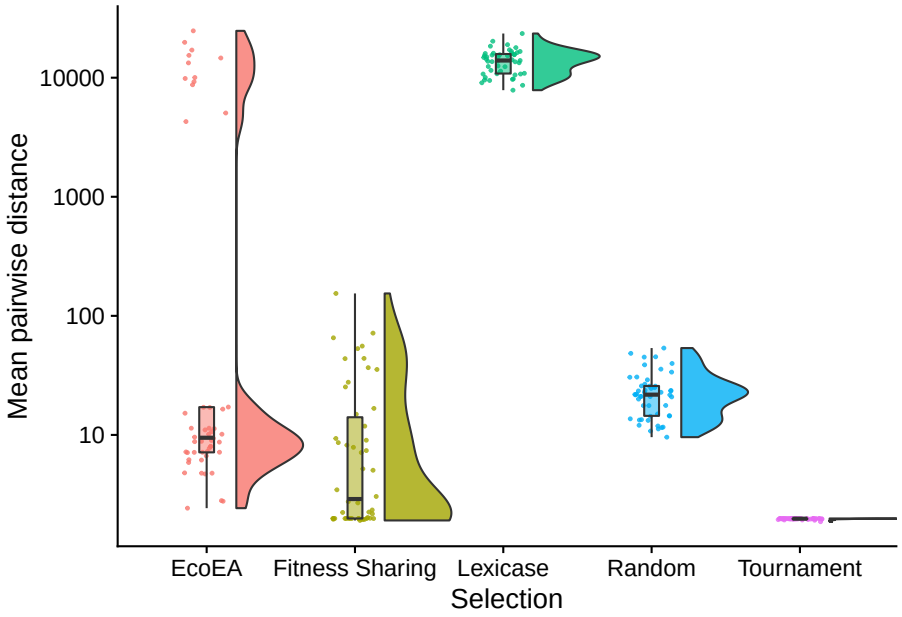
```
# Pairwise wilcoxon teset to determine which conditions are significantly different fr
stat.test <- final_data %>%
  wilcox_test(mean_phenotype_pairwise_distance ~ selection_name) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="selection_name",step.increase=1)
stat.test$label <- mapply(p_label,stat.test$p.adj)

# Output stats
stat.test %>%
  kbl() %>%
  kable_styling(
    bootstrap_options = c(
      "striped",
      "hover",
      "condensed",
      "responsive"
```

.y.	group1	group2	n1	n2	statistic	p	p
mean_phenotype_pairwise_distance	EcoEA	Fitness Sharing	50	50	1824.0	7.70e-05	7.70
mean_phenotype_pairwise_distance	EcoEA	Lexicase	50	47	227.0	0.00e+00	0.00e
mean_phenotype_pairwise_distance	EcoEA	Random	50	50	690.0	1.15e-04	1.15
mean_phenotype_pairwise_distance	EcoEA	Tournament	50	50	2500.0	0.00e+00	0.00e
mean_phenotype_pairwise_distance	Fitness Sharing	Lexicase	50	47	0.0	0.00e+00	0.00e
mean_phenotype_pairwise_distance	Fitness Sharing	Random	50	50	536.0	9.00e-07	8.70
mean_phenotype_pairwise_distance	Fitness Sharing	Tournament	50	50	2232.5	0.00e+00	0.00e
mean_phenotype_pairwise_distance	Lexicase	Random	47	50	2350.0	0.00e+00	0.00e
mean_phenotype_pairwise_distance	Lexicase	Tournament	47	50	2350.0	0.00e+00	0.00e
mean_phenotype_pairwise_distance	Random	Tournament	50	50	2500.0	0.00e+00	0.00e

```
    )
  ) %>%
  scroll_box(width="600px")
```

```
# Raincloud plot of final mean pairwise distance
final_phylogeny_fig <- ggplot(
    final_data,
    aes(
      x=selection_name,
      y=mean_phenotype_pairwise_distance,
      fill=selection_name
    )
  ) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8,
    scale="width"
  ) +
  geom_point(
    mapping=aes(color=selection_name),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_y_log10(
    name="Mean pairwise distance"
  ) +
```

```
  scale_x_discrete(
    name="Selection"
  ) +
  scale_fill_discrete(
    name="Selection"
  ) +
  scale_color_discrete(
    name="Selection"
  ) +
  theme(legend.position = "none")
final_phylogeny_fig
```



This shows something interesting! Final phylogenetic diversity in Eco-EA is heavily bimodal. In later analysis, we will see that the runs with high phylogenetic diversity are the ones with lower fitness, suggesting that they have no yet experienced a selective sweep resulting from the discovery of a high-fitness solution.

2.3.2.2 Mean evolutionary distinctiveness

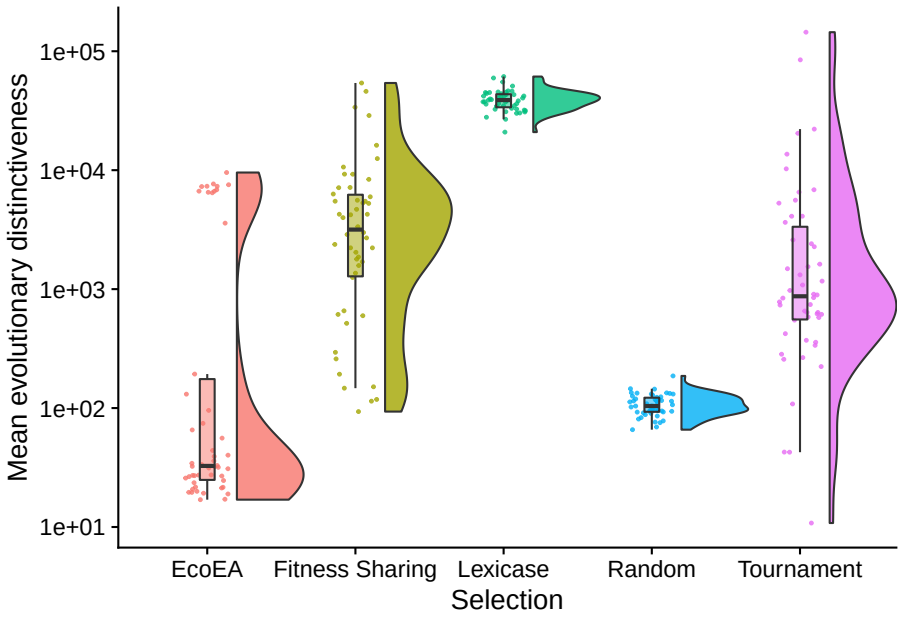
```
# Pairwise wilcoxon teset to determine which conditions are significantly different fr
stat.test <- final_data %>%
  wilcox_test(mean_phenotype_evolutionary_distinctiveness ~ selection_name) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
```

.y.	group1	group2	n1	n2	statistic	
mean_phenotype_evolutionary_distinctiveness	EcoEA	Fitness Sharing	50	50	469	1.00e
mean_phenotype_evolutionary_distinctiveness	EcoEA	Lexicase	50	47	0	0.00e+
mean_phenotype_evolutionary_distinctiveness	EcoEA	Random	50	50	711	2.05e
mean_phenotype_evolutionary_distinctiveness	EcoEA	Tournament	50	50	569	2.70e
mean_phenotype_evolutionary_distinctiveness	Fitness Sharing	Lexicase	50	47	100	0.00e+
mean_phenotype_evolutionary_distinctiveness	Fitness Sharing	Random	50	50	2428	0.00e+
mean_phenotype_evolutionary_distinctiveness	Fitness Sharing	Tournament	50	50	1614	1.20e
mean_phenotype_evolutionary_distinctiveness	Lexicase	Random	47	50	2350	0.00e+
mean_phenotype_evolutionary_distinctiveness	Lexicase	Tournament	47	50	2255	0.00e+
mean_phenotype_evolutionary_distinctiveness	Random	Tournament	50	50	173	0.00e+

```
  add_xy_position(x="selection_name",step.increase=1)
stat.test$label <- mapply(p_label,stat.test$p.adj)

# Output stats

stat.test %>%
  kbl() %>%
  kable_styling(
    bootstrap_options = c(
      "striped",
      "hover",
      "condensed",
      "responsive"
    )
  ) %>%
  scroll_box(width="600px")
```

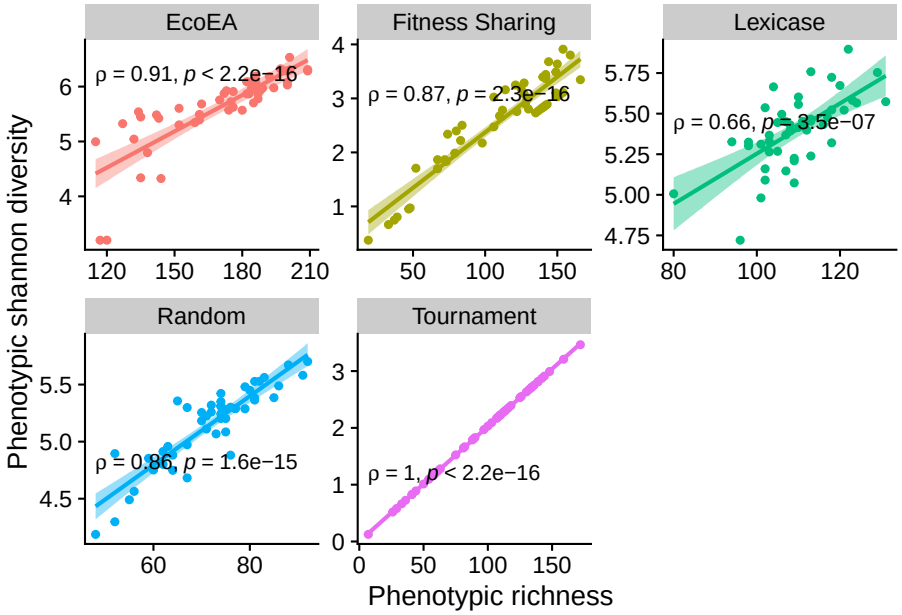
```
# Raincloud plot of final mean evolutionary distinctiveness
ggplot(
    final_data,
    aes(
      x=selection_name,
      y=mean_phenotype_evolutionary_distinctiveness,
      fill=selection_name
    )
  ) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8,
    scale="width"
  ) +
  geom_point(
```

```
    mapping=aes(color=selection_name),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_y_log10(
    name="Mean evolutionary distinctiveness"
  ) +
  scale_x_discrete(
    name="Selection"
  ) +
  scale_fill_discrete(
    name="Selection"
  ) +
  scale_color_discrete(
    name="Selection"
  ) +
  theme(legend.position = "none")
```



2.4 Phenotypic diversity

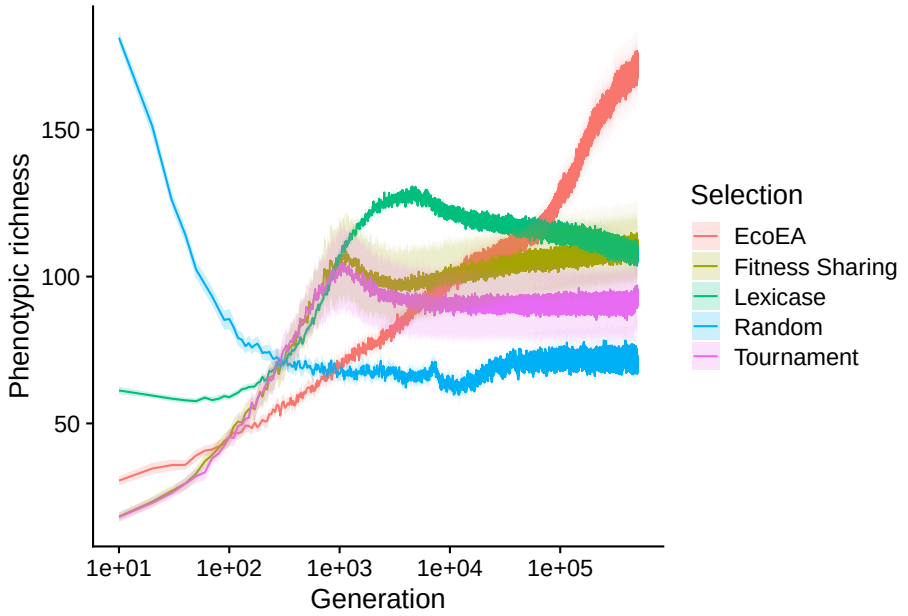
```
ggplot(
    data %>% filter(gen==500000),
    aes(
        y=phen_diversity,
        x=phen_num_taxa,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Phenotypic shannon diversity"
  ) +
  scale_x_continuous(
        name="Phenotypic richness",
        breaks = breaks_extended(4)
  ) +
  facet_wrap(
      ~selection_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```



2.4.2 Over time

```
ggplot(
    data,
    aes(
      x=gen,
      y=phen_num_taxa,
      color=selection_name,
      fill=selection_name
    )
  ) +
  stat_summary(geom="line", fun=mean) +
  stat_summary(
    geom="ribbon",
    fun.data="mean_cl_boot",
    fun.args=list(conf.int=0.95),
    alpha=0.2,
    linetype=0
  ) +
  scale_y_continuous(
    name="Phenotypic richness"
  ) +
  scale_x_log10(
    name="Generation"
```

```
  ) +
  scale_color_discrete("Selection") +
  scale_fill_discrete("Selection")
```

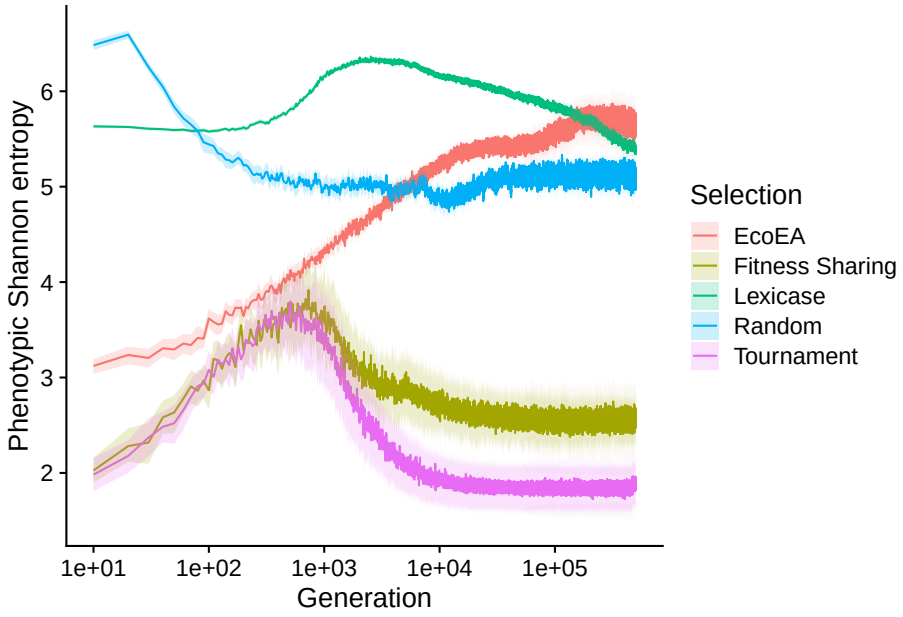


In contrast to the phylodiversity results, phenotypic richness in all selection schemes (even tournament selection) ultimately exceeds that of random selection. Eco-EA monotonically increases while lexicase selection reaches a maximum around the same time it reaches its fitness plateau. The only real similarity to the phylodiversity results is the behavior tournament selection and fitness sharing relative to each other.

We also looked at phenotypic shannon entropy:

```
ggplot(
    data,
    aes(
      x=gen,
      y=phen_diversity,
      color=selection_name,
      fill=selection_name
    )
  ) +
  stat_summary(geom="line", fun=mean) +
  stat_summary(
    geom="ribbon",
    fun.data="mean_cl_boot",
```

```
    fun.args=list(conf.int=0.95),
    alpha=0.2,
    linetype=0
  ) +
  scale_y_continuous(
    name="Phenotypic Shannon entropy"
  ) +
  scale_x_log10(
    name="Generation"
  ) +
  scale_color_discrete("Selection") +
  scale_fill_discrete("Selection")
```



2.4.3 Final

2.4.3.1 Richness

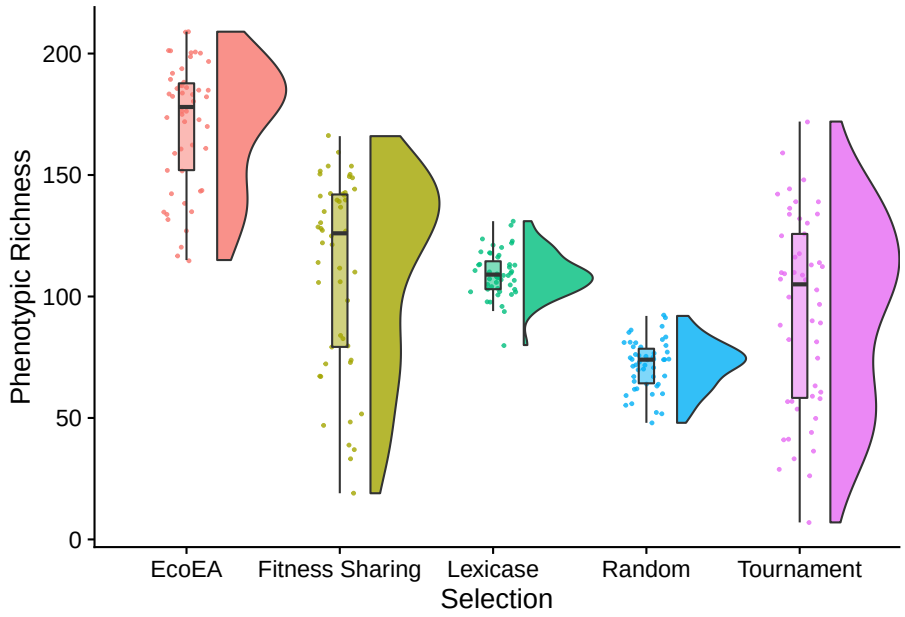
```
# Determine which conditions are significantly diferrent from each other
stat.test <- final_data %>%
  wilcox_test(phen_num_taxa ~ selection_name) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="selection_name",step.increase=1)
stat.test$label <- mapply(p_label,stat.test$p.adj)
```

.y.	group1	group2	n1	n2	statistic	p	p.adj	p.adj.signif
phen_num_taxa	EcoEA	Fitness Sharing	50	50	2249.0	0.00e+00	0.0e+00	****
phen_num_taxa	EcoEA	Lexicase	50	47	2319.0	0.00e+00	0.0e+00	****
phen_num_taxa	EcoEA	Random	50	50	2500.0	0.00e+00	0.0e+00	****
phen_num_taxa	EcoEA	Tournament	50	50	2378.5	0.00e+00	0.0e+00	****
phen_num_taxa	Fitness Sharing	Lexicase	50	47	1428.0	6.80e-02	6.8e-01	ns
phen_num_taxa	Fitness Sharing	Random	50	50	1973.0	6.00e-07	6.3e-06	****
phen_num_taxa	Fitness Sharing	Tournament	50	50	1585.0	2.10e-02	2.1e-01	ns
phen_num_taxa	Lexicase	Random	47	50	2339.5	0.00e+00	0.0e+00	****
phen_num_taxa	Lexicase	Tournament	47	50	1359.0	1.85e-01	1.0e+00	ns
phen_num_taxa	Random	Tournament	50	50	797.0	2.00e-03	2.0e-02	*

```
stat.test %>%
  kbl() %>%
  kable_styling(
    bootstrap_options = c(
      "striped",
      "hover",
      "condensed",
      "responsive"
    )
  ) %>%
  scroll_box(width="600px")
```

```
# Raincloud plot of final phenotypic diversity
final_phenotypic_fig <- ggplot(
    final_data,
    aes(
      x=selection_name,
      y=phen_num_taxa,
      fill=selection_name
    )
  ) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
    alpha = .8,
    scale="width"
  ) +
  geom_point(
    mapping=aes(color=selection_name),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
```

```
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_y_continuous(
    name="Phenotypic Richness"
  ) +
  scale_x_discrete(
    name="Selection"
  ) +
  scale_fill_discrete(
    name="Selection"
  ) +
  scale_color_discrete(
    name="Selection"
  ) +
  theme(legend.position = "none")
final_phenotypic_fig
```



Nothing particularly suprising here, but we should note that, based on the over time plot, this would look a lot different if we had selected a different time point.

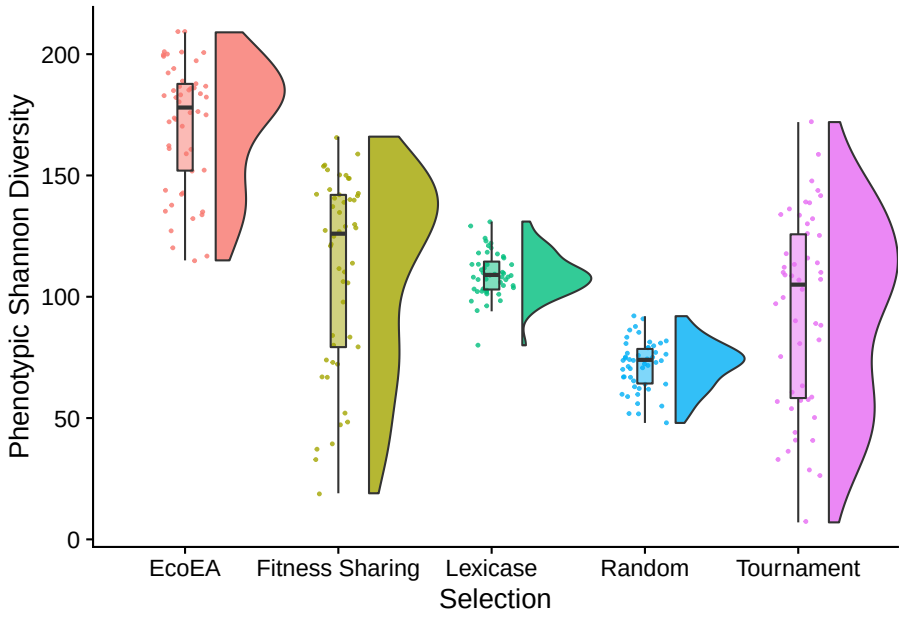
.y.	group1	group2	n1	n2	statistic	p	p.adj	p.adj.signif	y
phen_diversity	EcoEA	Fitness Sharing	50	50	2478.0	0.00e+00	0.00e+00	****	
phen_diversity	EcoEA	Lexicase	50	47	1772.0	1.66e-05	1.66e-04	***	
phen_diversity	EcoEA	Random	50	50	2089.5	0.00e+00	1.00e-07	****	
phen_diversity	EcoEA	Tournament	50	50	2496.0	0.00e+00	0.00e+00	****	
phen_diversity	Fitness Sharing	Lexicase	50	47	0.0	0.00e+00	0.00e+00	****	
phen_diversity	Fitness Sharing	Random	50	50	0.0	0.00e+00	0.00e+00	****	
phen_diversity	Fitness Sharing	Tournament	50	50	1856.0	2.99e-05	2.99e-04	***	
phen_diversity	Lexicase	Random	47	50	1714.0	1.01e-04	1.01e-03	**	
phen_diversity	Lexicase	Tournament	47	50	2350.0	0.00e+00	0.00e+00	****	
phen_diversity	Random	Tournament	50	50	2500.0	0.00e+00	0.00e+00	****	

2.4.3.2 Shannon diversity

```
# Determine which conditions are significantly diferrent from each other
stat.test <- final_data %>%
  wilcox_test(phen_diversity~ selection_name) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="selection_name",step.increase=1)
stat.test$label <- mapply(p_label,stat.test$p.adj)

stat.test %>%
  kbl() %>%
  kable_styling(
    bootstrap_options = c(
      "striped",
      "hover",
      "condensed",
      "responsive"
    )
  ) %>%
  scroll_box(width="600px")
```

```
# Raincloud plot of final phenotypic diversity
ggplot(
    final_data,
    aes(
      x=selection_name,
      y=phen_num_taxa,
      fill=selection_name
    )
  ) +
  geom_flat_violin(
    position = position_nudge(x = .2, y = 0),
```

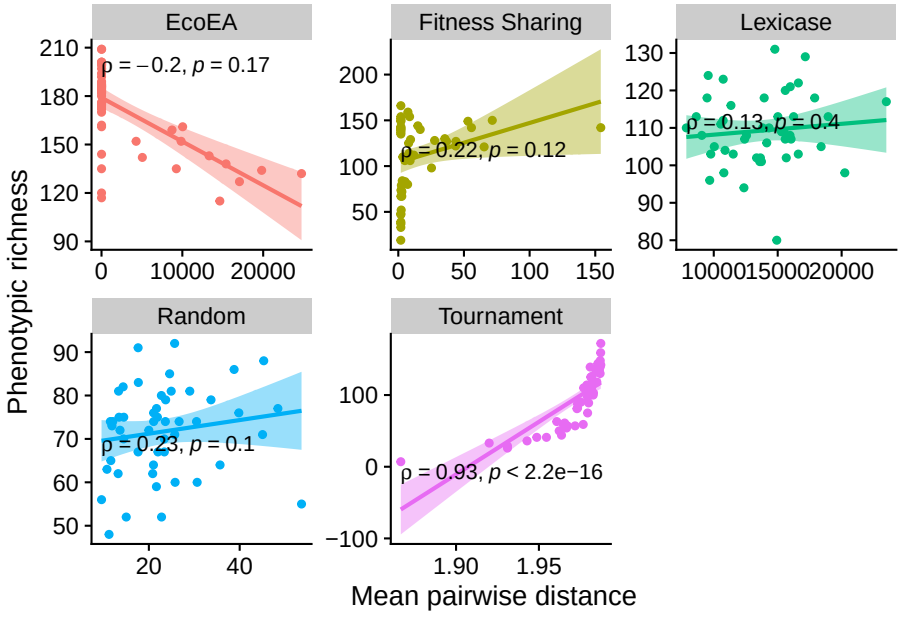
```
    alpha = .8,
    scale="width"
  ) +
  geom_point(
    mapping=aes(color=selection_name),
    position = position_jitter(width = .15),
    size = .5,
    alpha = 0.8
  ) +
  geom_boxplot(
    width = .1,
    outlier.shape = NA,
    alpha = 0.5
  ) +
  scale_y_continuous(
    name="Phenotypic Shannon Diversity"
  ) +
  scale_x_discrete(
    name="Selection"
  ) +
  scale_fill_discrete(
    name="Selection"
  ) +
  scale_color_discrete(
    name="Selection"
  ) +
  theme(legend.position = "none")
```



2.5 Relationship between phenotypic and phylogenetic diversity

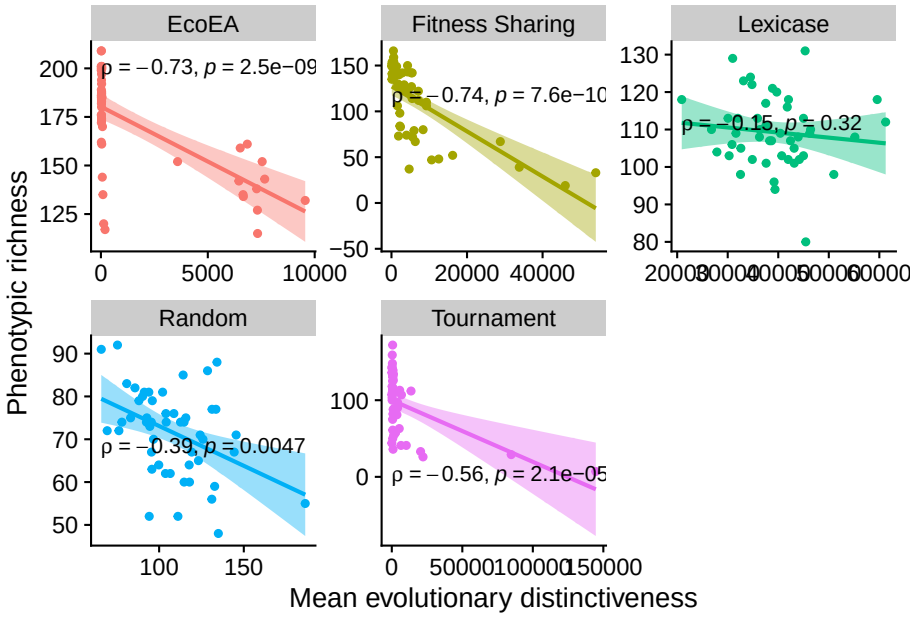
```
ggplot(
    data %>% filter(gen==500000),
    aes(
        y=phen_num_taxa,
        x=mean_phenotype_pairwise_distance,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Phenotypic richness"
  ) +
  scale_x_continuous(
        name="Mean pairwise distance",
        breaks = breaks_extended(4)
  ) +
  facet_wrap(
      ~selection_name, scales="free"
  ) +
```

```
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```



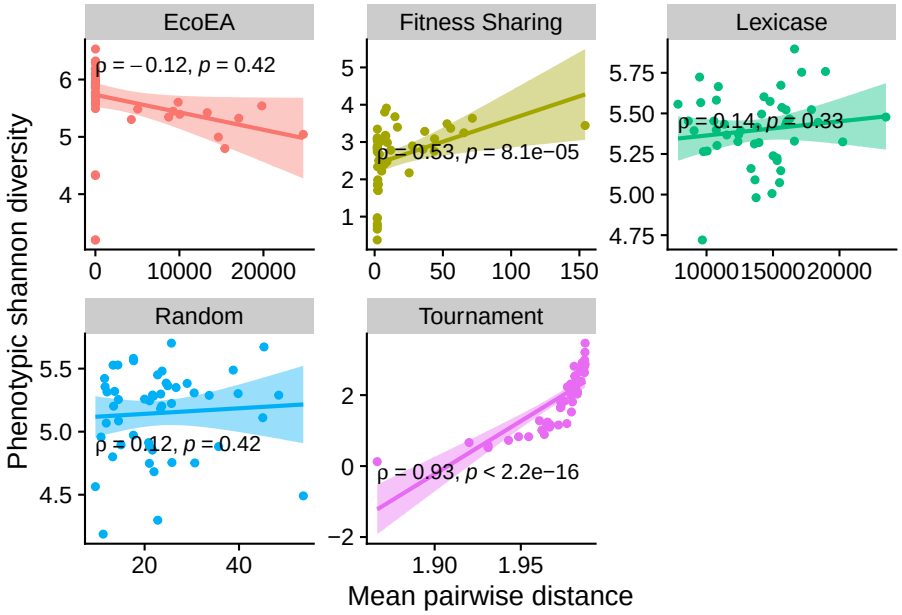

```
ggplot(
    data %>% filter(gen==500000),
    aes(
        y=phen_num_taxa,
        x=mean_phenotype_evolutionary_distinctiveness,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Phenotypic richness"
  ) +
  scale_x_continuous(
        name="Mean evolutionary distinctiveness",
        breaks = breaks_extended(4)
  ) +
```

```
  facet_wrap(
      ~selection_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```



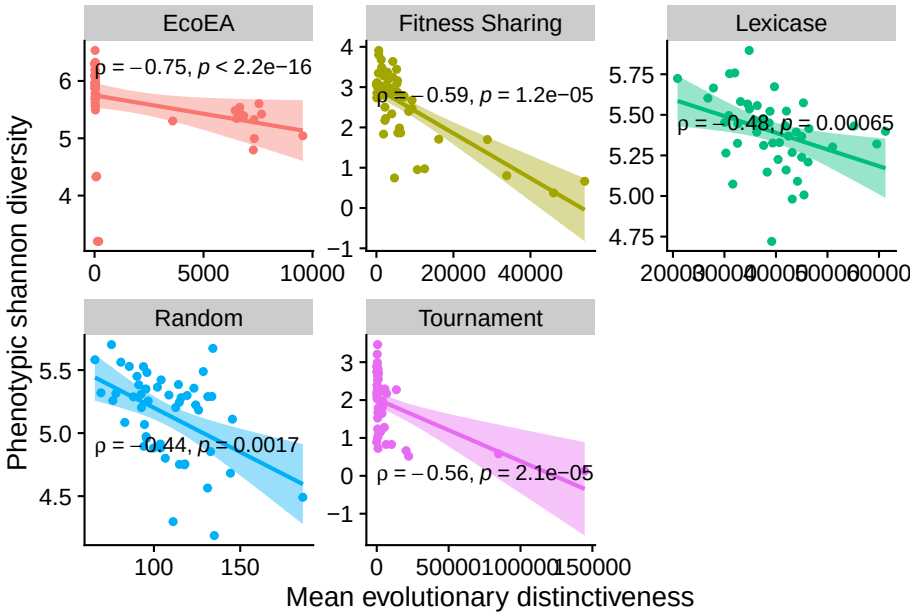

```
ggplot(
    data %>% filter(gen==500000),
    aes(
        y=phen_diversity,
        x=mean_phenotype_pairwise_distance,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Phenotypic shannon diversity"
  ) +
  scale_x_continuous(
```

```
        name="Mean pairwise distance",
        breaks = breaks_extended(4)
  ) +
  facet_wrap(
      ~selection_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```




```
ggplot(
    data %>% filter(gen==500000),
    aes(
        y=phen_diversity,
        x=mean_phenotype_evolutionary_distinctiveness,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
```

```
        name="Phenotypic shannon diversity"
  ) +
  scale_x_continuous(
        name="Mean evolutionary distinctiveness",
        breaks = breaks_extended(4)
  ) +
  facet_wrap(
      ~selection_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```

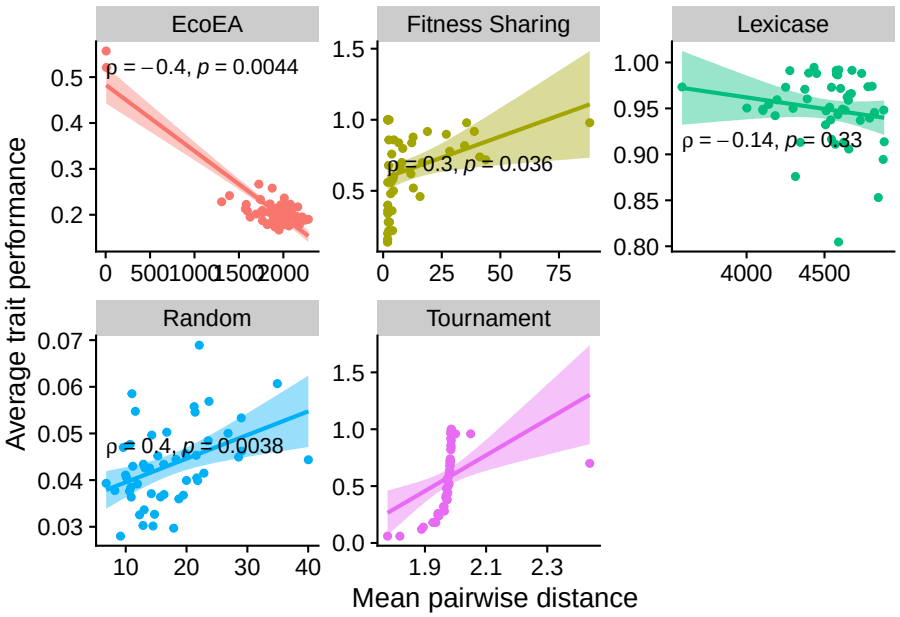


2.6 Relationship between diversity and success

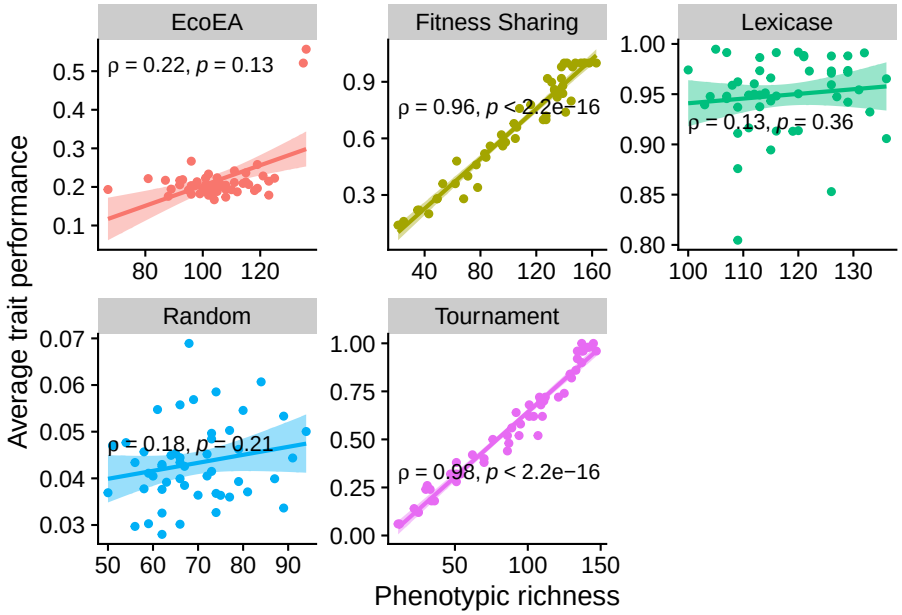
2.6.1 Earlier in run

```
ggplot(
    data %>% filter(gen==25000),
    aes(
```

```
        y=elite_trait_avg,
        x=mean_phenotype_pairwise_distance,
        color=selection_name,
        fill=selection_name
    )
) +
geom_point() +
  scale_y_continuous(
        name="Average trait performance"
) +
scale_x_continuous(
        name="Mean pairwise distance"
) +
facet_wrap(
     ~selection_name, scales="free"
) +
stat_smooth(
  method="lm"
) +
stat_cor(
  method="spearman", cor.coef.name = "rho", color="black"
) +
theme(legend.position = "none")
```

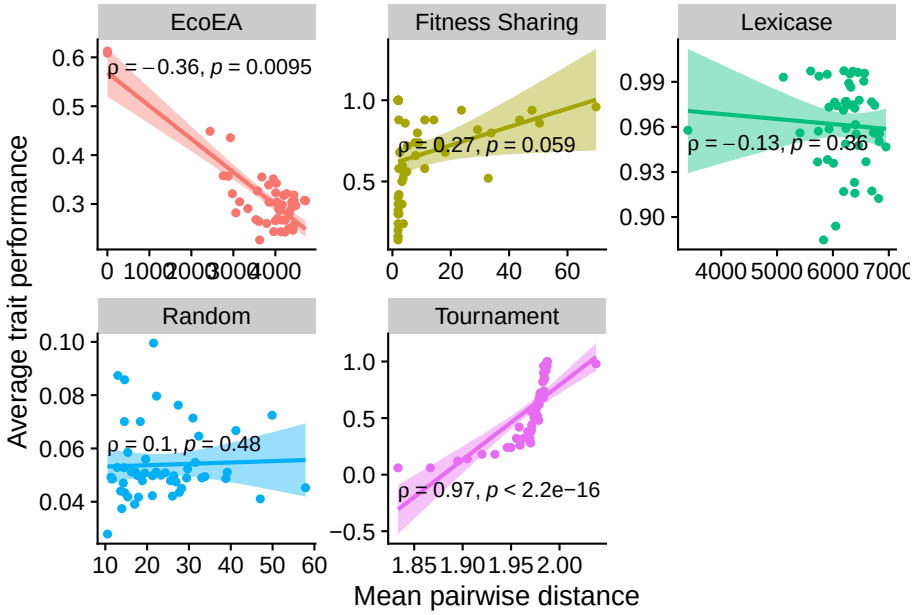


```
ggplot(
    data %>% filter(gen==25000),
    aes(
        y=elite_trait_avg,
        x=phen_num_taxa,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Average trait performance"
  ) +
  scale_x_continuous(
        name="Phenotypic richness"
  ) +
  facet_wrap(
      ~selection_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```



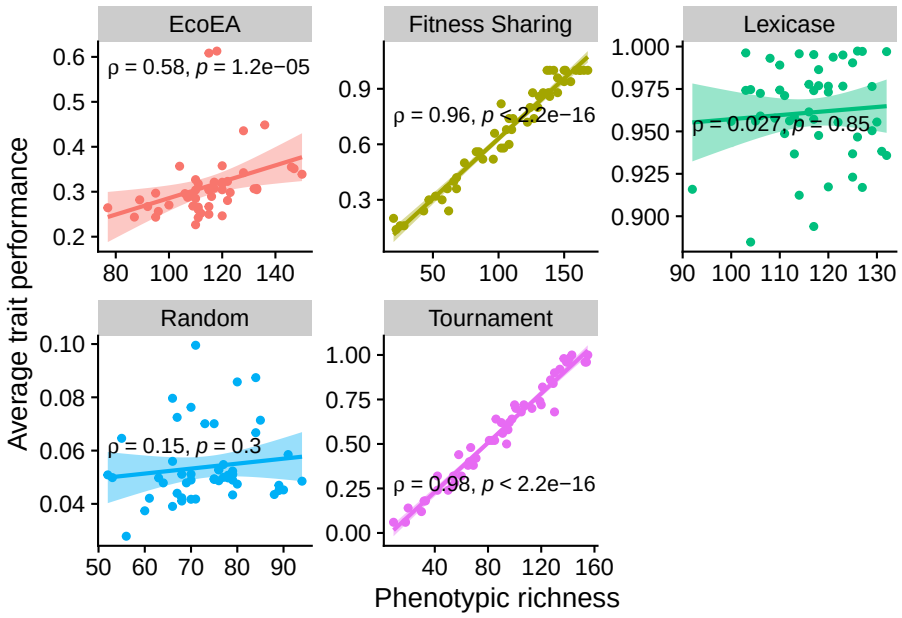

```
phylogney_vs_performance <- ggplot(
    data %>% filter(gen==50000),
    aes(
        y=elite_trait_avg,
        x=mean_phenotype_pairwise_distance,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Average trait performance"
  ) +
  scale_x_continuous(
        name="Mean pairwise distance"
  ) +
  facet_wrap(
      ~selection_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
```

```
  theme(legend.position = "none")

phylogney_vs_performance
```




```
richness_vs_performance <- ggplot(
    data %>% filter(gen==50000),
    aes(
        y=elite_trait_avg,
        x=phen_num_taxa,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Average trait performance"
  ) +
  scale_x_continuous(
        name="Phenotypic richness"
  ) +
  facet_wrap(
      ~selection_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
```

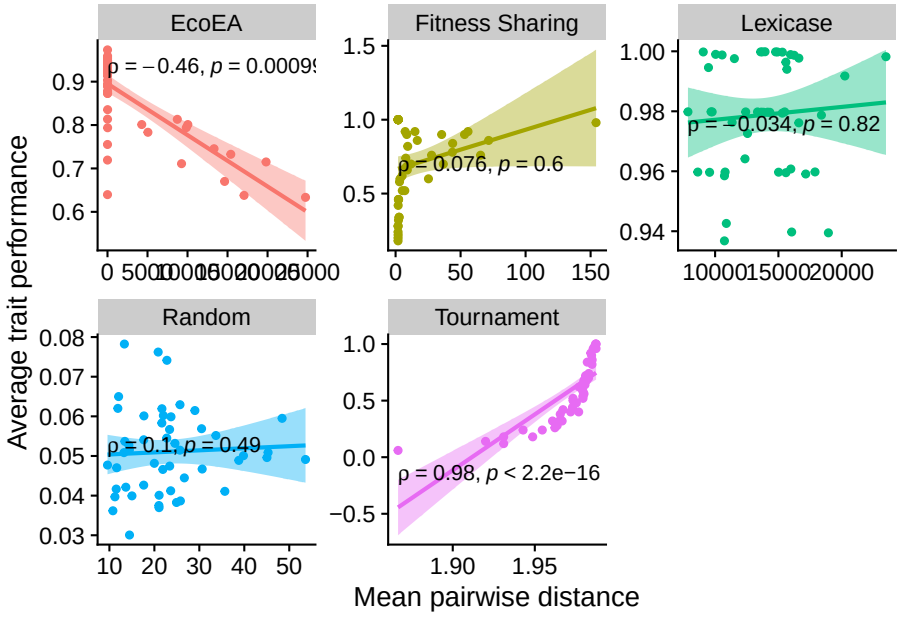
```
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")

richness_vs_performance
```



2.6.2 End of run

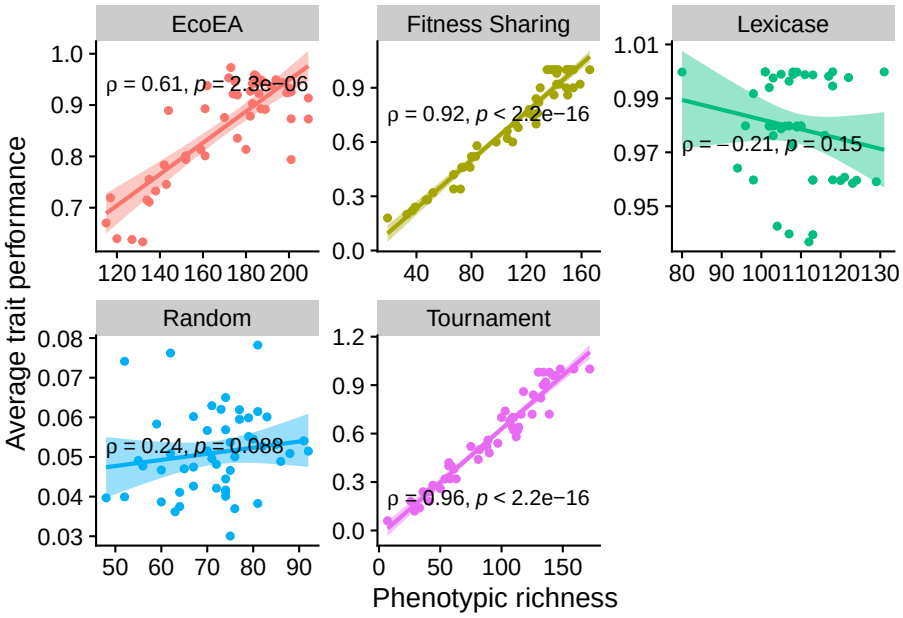
```
final_phylogney_vs_performance <- ggplot(
    final_data,
    aes(
        y=elite_trait_avg,
        x=mean_phenotype_pairwise_distance,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Average trait performance"
  ) +
  scale_x_continuous(
```

```
        name="Mean pairwise distance"
  ) +
  facet_wrap(
      ~selection_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")

final_phylogney_vs_performance
```




```
final_richness_vs_performance <- ggplot(
    final_data,
    aes(
        y=elite_trait_avg,
        x=phen_num_taxa,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
```

```
    scale_y_continuous(
        name="Average trait performance"
  ) +
  scale_x_continuous(
        name="Phenotypic richness"
  ) +
  facet_wrap(
      ~selection_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")

final_richness_vs_performance
```



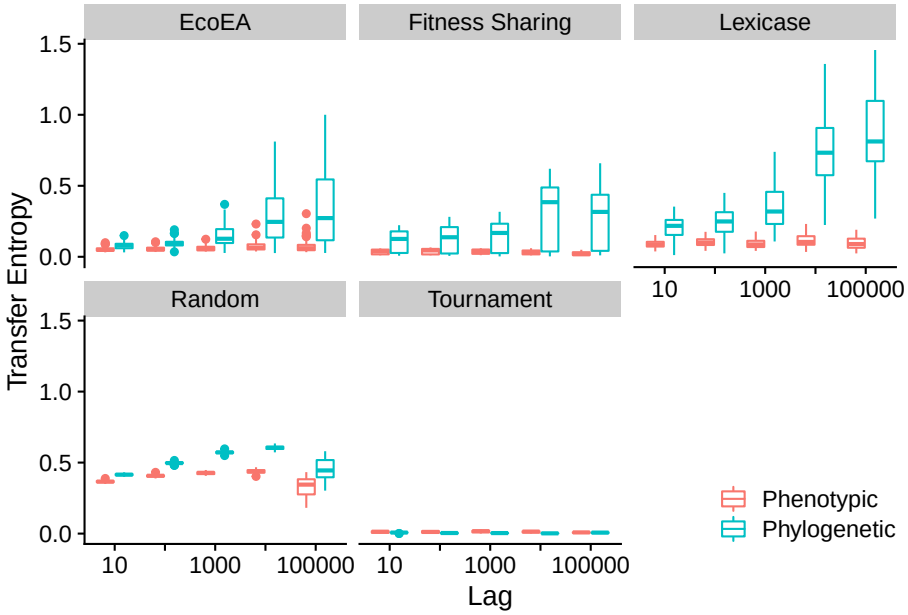
2.7 Causality analysis

2.7.1 Transfer entropy from diversity to fitness

2.7.1.1 Max pairwise distance vs. phenotypic richness

```
# Calculate transfer entropy for max pairwise distance
# Time points are 10 generations, so calculating lag 1 gives us lag 10
res <- data %>% group_by(directory, selection_name) %>%
summarise(
  fit_phylo_10 =     condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 10
                                    discretize(lag(max_phenotype_pairwise_distance, 1)),
                                    discretize(lag(elite_trait_avg, 1))),
  fit_phylo_100 =    condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 100
                                    discretize(lag(max_phenotype_pairwise_distance, 10)),
                                    discretize(lag(elite_trait_avg, 10))),
  fit_phylo_1000 =   condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 1000
                                    discretize(lag(max_phenotype_pairwise_distance, 100)),
                                    discretize(lag(elite_trait_avg, 100))),
  fit_phylo_10000 =  condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 10000
                                    discretize(lag(max_phenotype_pairwise_distance, 1000)),
                                    discretize(lag(elite_trait_avg, 1000))),
  fit_phylo_100000 = condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 100000
                                    discretize(lag(max_phenotype_pairwise_distance, 10000)),
                                    discretize(lag(elite_trait_avg, 10000))),
  fit_fit_10000 =    condinformation(discretize(elite_trait_avg), # Mutual info btwn. fit and lag
                                    discretize(lag(elite_trait_avg, 1000)),
                                    discretize(lag(max_phenotype_pairwise_distance, 1000))),
  fit_fit_100000 =   condinformation(discretize(elite_trait_avg), # Mutual info btwn. fit and lag
                                    discretize(lag(elite_trait_avg, 10000)),
                                    discretize(lag(max_phenotype_pairwise_distance, 10000))),
  fit_pheno_10 =     condinformation(discretize(elite_trait_avg), # TE pheno -> fit, lag 10
                                    discretize(lag(phen_num_taxa, 1)),
                                    discretize(lag(elite_trait_avg, 1))),
  fit_pheno_100 =    condinformation(discretize(elite_trait_avg),  # TE pheno -> fit, lag 100
                                    discretize(lag(phen_num_taxa, 10)),
                                    discretize(lag(elite_trait_avg, 10))),
  fit_pheno_1000 =   condinformation(discretize(elite_trait_avg),  # TE pheno -> fit, lag 1000
                                    discretize(lag(phen_num_taxa, 100)),
                                    discretize(lag(elite_trait_avg, 100))),
  fit_pheno_10000 =  condinformation(discretize(elite_trait_avg),  # TE pheno -> fit, lag 10000
                                    discretize(lag(phen_num_taxa, 1000)),
                                    discretize(lag(elite_trait_avg, 1000))),
  fit_pheno_100000 = condinformation(discretize(elite_trait_avg),  # TE pheno -> fit, lag 100000
                                    discretize(lag(phen_num_taxa, 10000)),
                                    discretize(lag(elite_trait_avg, 10000)))
```

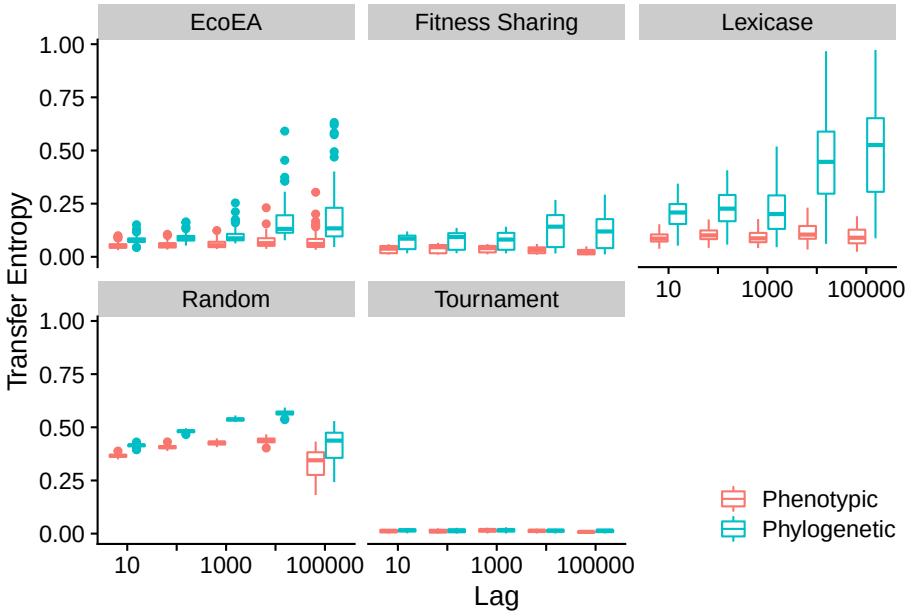
```
)

# Turn Transfer Entropy columns into rows
res <- res %>% pivot_longer(cols=contains("o_10"))
# Pull lag into a column
res$offset <- str_extract(res$name, "[:digit:]*$")
# Make column indicating direction of transfer entropy
res$Type <- case_when(str_detect(res$name, "phylo") ~ "Phylogenetic", TRUE ~ "Phenotypi

# Plot transfer entropy
ggplot(
  res %>% filter(str_detect(name, "fit_ph*")),
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~selection_name) +
  scale_x_discrete("Lag",labels=c("10","","1000","","100000")) +
  scale_y_continuous("Transfer Entropy") +
  theme(legend.position = c(1, 0),
        legend.justification = c(1, 0)) +
  scale_color_discrete("")
```



2.7.1.2 Mean pairwise distance vs. phenotypic richness

```
# Calculate transfer entropy for mean pairwise distance
# Time points are 10 generations, so calculating lag 1 gives us lag 10
res <- data %>% group_by(directory, selection_name) %>%
summarise(
  fit_phylo_10 =     condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 10
                                    discretize(lag(mean_phenotype_pairwise_distance, 1)),
                                    discretize(lag(elite_trait_avg, 1))),
  fit_phylo_100 =    condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 100
                                    discretize(lag(mean_phenotype_pairwise_distance, 10)),
                                    discretize(lag(elite_trait_avg, 10))),
  fit_phylo_1000 =   condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 1000
                                    discretize(lag(mean_phenotype_pairwise_distance, 100)),
                                    discretize(lag(elite_trait_avg, 100))),
  fit_phylo_10000 =  condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 10000
                                    discretize(lag(mean_phenotype_pairwise_distance, 1000)),
                                    discretize(lag(elite_trait_avg, 1000))),
  fit_phylo_100000 = condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 100000
                                    discretize(lag(mean_phenotype_pairwise_distance, 10000)),
                                    discretize(lag(elite_trait_avg, 10000))),
  fit_fit_10000 =    condinformation(discretize(elite_trait_avg), # Mutual info btwn. fit and lag
                                    discretize(lag(elite_trait_avg, 1000)),
                                    discretize(lag(mean_phenotype_pairwise_distance, 1000))),
```

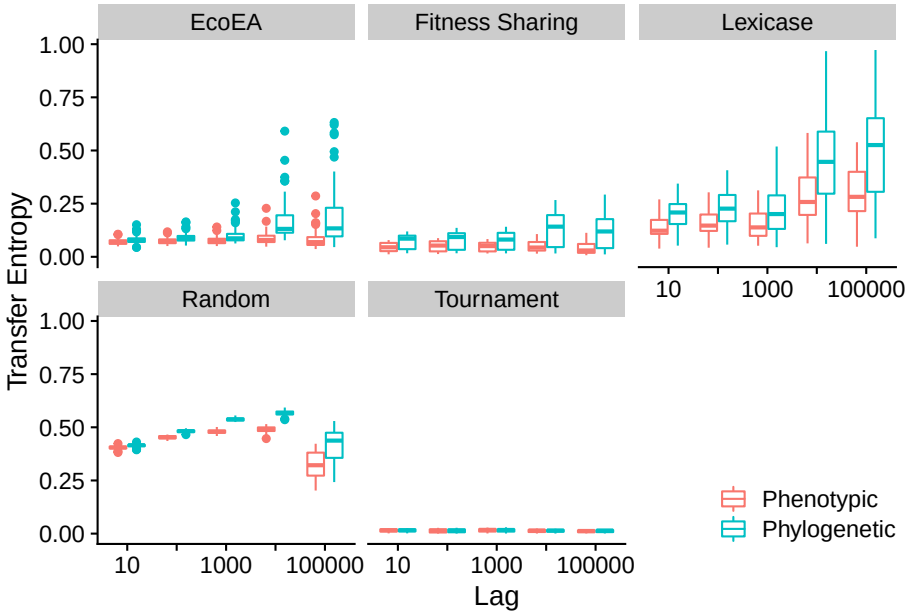
```
  fit_fit_100000 =   condinformation(discretize(elite_trait_avg), # Mutual info btwn.
                                     discretize(lag(elite_trait_avg, 10000)),
                                     discretize(lag(mean_phenotype_pairwise_distance,
  fit_pheno_10 =     condinformation(discretize(elite_trait_avg), # TE pheno -> fit, l
                                     discretize(lag(phen_num_taxa, 1)),
                                     discretize(lag(elite_trait_avg, 1))),
  fit_pheno_100 =    condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_num_taxa, 10)),
                                     discretize(lag(elite_trait_avg, 10))),
  fit_pheno_1000 =   condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_num_taxa, 100)),
                                     discretize(lag(elite_trait_avg, 100))),
  fit_pheno_10000 =  condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_num_taxa, 1000)),
                                     discretize(lag(elite_trait_avg, 1000))),
  fit_pheno_100000 = condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_num_taxa, 10000)),
                                     discretize(lag(elite_trait_avg, 10000)))
)

# Turn Transfer Entropy columns into rows
res <- res %>% pivot_longer(cols=contains("o_10"))
# Pull lag into a column
res$offset <- str_extract(res$name, "[:digit:]*$")
# Make column indicating direction of transfer entropy
res$Type <- case_when(str_detect(res$name, "phylo") ~ "Phylogenetic", TRUE ~ "Phenotypi

# Plot transfer entropy
ggplot(
  res %>% filter(str_detect(name, "fit_ph*")),
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~selection_name) +
  scale_x_discrete("Lag",labels=c("10","","1000","","100000")) +
  scale_y_continuous("Transfer Entropy") +
  theme(legend.position = c(1, 0),
        legend.justification = c(1, 0)) +
  scale_color_discrete("")
```



2.7.1.3 Mean pairwise distance vs. shannon entropy

```
# Calculate transfer entropy for mean pairwise distance
# Time points are 10 generations, so calculating lag 1 gives us lag 10
res <- data %>% group_by(directory, selection_name) %>%
summarise(
  fit_phylo_10 =      condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 10
                                     discretize(lag(mean_phenotype_pairwise_distance, 1)),
                                     discretize(lag(elite_trait_avg, 1))),
  fit_phylo_100 =     condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 100
                                     discretize(lag(mean_phenotype_pairwise_distance, 10)),
                                     discretize(lag(elite_trait_avg, 10))),
  fit_phylo_1000 =    condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 1000
                                     discretize(lag(mean_phenotype_pairwise_distance, 100)),
                                     discretize(lag(elite_trait_avg, 100))),
  fit_phylo_10000 =   condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 10000
                                     discretize(lag(mean_phenotype_pairwise_distance, 1000)),
                                     discretize(lag(elite_trait_avg, 1000))),
  fit_phylo_100000 = condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 100000
                                     discretize(lag(mean_phenotype_pairwise_distance, 10000)),
                                     discretize(lag(elite_trait_avg, 10000))),
  fit_fit_10000 =     condinformation(discretize(elite_trait_avg), # Mutual info btwn. fit and lag
                                     discretize(lag(elite_trait_avg, 1000)),
                                     discretize(lag(mean_phenotype_pairwise_distance, 1000))),
```

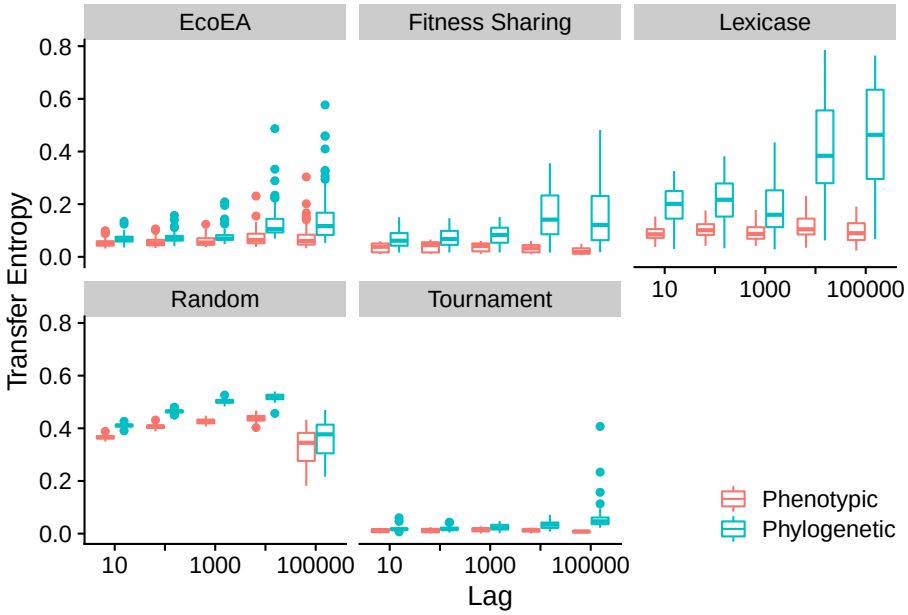
```
  fit_fit_100000 =   condinformation(discretize(elite_trait_avg), # Mutual info btwn.
                                     discretize(lag(elite_trait_avg, 10000)),
                                     discretize(lag(mean_phenotype_pairwise_distance,
  fit_pheno_10 =     condinformation(discretize(elite_trait_avg), # TE pheno -> fit, l
                                     discretize(lag(phen_diversity, 1)),
                                     discretize(lag(elite_trait_avg, 1))),
  fit_pheno_100 =    condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_diversity, 10)),
                                     discretize(lag(elite_trait_avg, 10))),
  fit_pheno_1000 =   condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_diversity, 100)),
                                     discretize(lag(elite_trait_avg, 100))),
  fit_pheno_10000 =  condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_diversity, 1000)),
                                     discretize(lag(elite_trait_avg, 1000))),
  fit_pheno_100000 = condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_diversity, 10000)),
                                     discretize(lag(elite_trait_avg, 10000)))
)

# Turn Transfer Entropy columns into rows
res <- res %>% pivot_longer(cols=contains("o_10"))
# Pull lag into a column
res$offset <- str_extract(res$name, "[:digit:]*$")
# Make column indicating direction of transfer entropy
res$Type <- case_when(str_detect(res$name, "phylo") ~ "Phylogenetic", TRUE ~ "Phenotyp

# Plot transfer entropy
ggplot(
  res %>% filter(str_detect(name, "fit_ph*")),
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~selection_name) +
  scale_x_discrete("Lag",labels=c("10","","1000","","100000")) +
  scale_y_continuous("Transfer Entropy") +
  theme(legend.position = c(1, 0),
        legend.justification = c(1, 0)) +
  scale_color_discrete("")
```



2.7.1.4 Mean evolutionary distinctiveness vs. phenotypic richness

```
# Calculate transfer entropy for mean evolutionary distinctiveness
# Time points are 10 generations, so calculating lag 1 gives us lag 10
res <- data %>% group_by(directory, selection_name) %>%
summarise(
  fit_phylo_10 =      condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 10
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
                                     discretize(lag(elite_trait_avg, 1))),
  fit_phylo_100 =     condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 100
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
                                     discretize(lag(elite_trait_avg, 10))),
  fit_phylo_1000 =    condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 1000
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
                                     discretize(lag(elite_trait_avg, 100))),
  fit_phylo_10000 =   condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 10000
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
                                     discretize(lag(elite_trait_avg, 1000))),
  fit_phylo_100000 = condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 100000
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
                                     discretize(lag(elite_trait_avg, 10000))),
  fit_fit_10000 =     condinformation(discretize(elite_trait_avg), # Mutual info btwn. fit and lag
                                     discretize(lag(elite_trait_avg, 1000)),
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
```

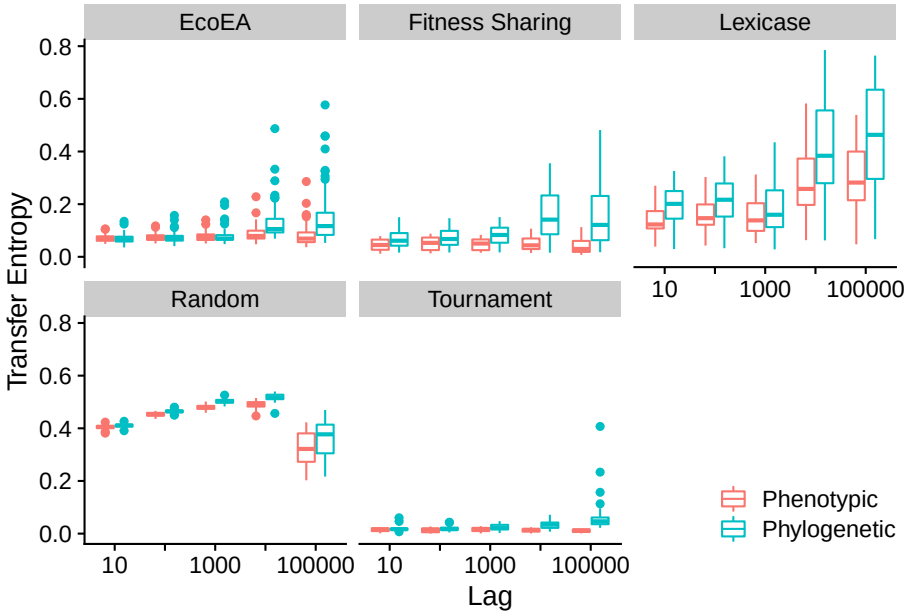
```
  fit_fit_100000 =   condinformation(discretize(elite_trait_avg), # Mutual info btwn.
                                     discretize(lag(elite_trait_avg, 10000)),
                                     discretize(lag(mean_phenotype_evolutionary_distinc
  fit_pheno_10 =     condinformation(discretize(elite_trait_avg), # TE pheno -> fit, l
                                     discretize(lag(phen_num_taxa, 1)),
                                     discretize(lag(elite_trait_avg, 1))),
  fit_pheno_100 =    condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_num_taxa, 10)),
                                     discretize(lag(elite_trait_avg, 10))),
  fit_pheno_1000 =   condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_num_taxa, 100)),
                                     discretize(lag(elite_trait_avg, 100))),
  fit_pheno_10000 =  condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_num_taxa, 1000)),
                                     discretize(lag(elite_trait_avg, 1000))),
  fit_pheno_100000 = condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_num_taxa, 10000)),
                                     discretize(lag(elite_trait_avg, 10000)))
)

# Turn Transfer Entropy columns into rows
res <- res %>% pivot_longer(cols=contains("o_10"))
# Pull lag into a column
res$offset <- str_extract(res$name, "[:digit:]*$")
# Make column indicating direction of transfer entropy
res$Type <- case_when(str_detect(res$name, "phylo") ~ "Phylogenetic", TRUE ~ "Phenotypi

# Plot transfer entropy
ggplot(
  res %>% filter(str_detect(name, "fit_ph*")),
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~selection_name) +
  scale_x_discrete("Lag",labels=c("10","","1000","","100000")) +
  scale_y_continuous("Transfer Entropy") +
  theme(legend.position = c(1, 0),
        legend.justification = c(1, 0)) +
  scale_color_discrete("")
```



2.7.1.5 Mean evolutionary distinctiveness vs. shannon entropy

```
# Calculate transfer entropy for mean evolutionary distinctiveness
# Time points are 10 generations, so calculating lag 1 gives us lag 10
res <- data %>% group_by(directory, selection_name) %>%
summarise(
  fit_phylo_10 =      condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 10
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
                                     discretize(lag(elite_trait_avg, 1))),
  fit_phylo_100 =     condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 100
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
                                     discretize(lag(elite_trait_avg, 10))),
  fit_phylo_1000 =    condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 1000
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
                                     discretize(lag(elite_trait_avg, 100))),
  fit_phylo_10000 =   condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 10000
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
                                     discretize(lag(elite_trait_avg, 1000))),
  fit_phylo_100000 = condinformation(discretize(elite_trait_avg), # TE phylo -> fit, lag 100000
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
                                     discretize(lag(elite_trait_avg, 10000))),
  fit_fit_10000 =     condinformation(discretize(elite_trait_avg), # Mutual info btwn. fit and lag
                                     discretize(lag(elite_trait_avg, 1000)),
                                     discretize(lag(mean_phenotype_evolutionary_distinctiveness,
```

```
  fit_fit_100000 =   condinformation(discretize(elite_trait_avg), # Mutual info btwm. j
                                     discretize(lag(elite_trait_avg, 10000)),
                                     discretize(lag(mean_phenotype_evolutionary_distinc
  fit_pheno_10 =     condinformation(discretize(elite_trait_avg), # TE pheno -> fit, la
                                     discretize(lag(phen_diversity, 1)),
                                     discretize(lag(elite_trait_avg, 1))),
  fit_pheno_100 =    condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_diversity, 10)),
                                     discretize(lag(elite_trait_avg, 10))),
  fit_pheno_1000 =   condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_diversity, 100)),
                                     discretize(lag(elite_trait_avg, 100))),
  fit_pheno_10000 =  condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_diversity, 1000)),
                                     discretize(lag(elite_trait_avg, 1000))),
  fit_pheno_100000 = condinformation(discretize(elite_trait_avg),  # TE pheno -> fit,
                                     discretize(lag(phen_diversity, 10000)),
                                     discretize(lag(elite_trait_avg, 10000)))
)

# Turn Transfer Entropy columns into rows
res <- res %>% pivot_longer(cols=contains("o_10"))
# Pull lag into a column
res$offset <- str_extract(res$name, "[:digit:]*$")
# Make column indicating direction of transfer entropy
res$Type <- case_when(str_detect(res$name, "phylo") ~ "Phylogenetic", TRUE ~ "Phenotypi

# Plot transfer entropy
ggplot(
  res %>% filter(str_detect(name, "fit_ph*")),
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~selection_name) +
  scale_x_discrete("Lag",labels=c("10","","1000","","100000")) +
  scale_y_continuous("Transfer Entropy") +
  theme(legend.position = c(1, 0),
        legend.justification = c(1, 0)) +
  scale_color_discrete("")
```

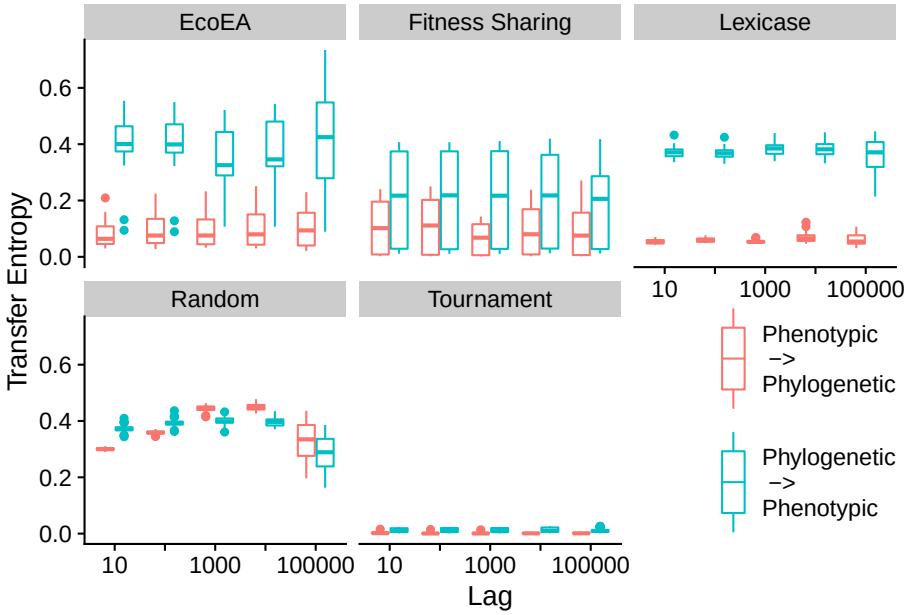


2.7.2 Transfer entropy between types of diversity

2.7.2.1 Max pairwise distance and phenotypic richness

```
res <- data %>% group_by(directory, selection_name) %>%
summarise(
  phen_phylo_10 =       condinformation(discretize(phen_num_taxa),
                                       discretize(lag(max_phenotype_pairwise_distance, 1)),
                                       discretize(lag(phen_num_taxa, 1))),
  phen_phylo_100 =      condinformation(discretize(phen_num_taxa),
                                       discretize(lag(max_phenotype_pairwise_distance, 10)),
                                       discretize(lag(phen_num_taxa, 10))),
  pheno_phylo_1000 =    condinformation(discretize(phen_num_taxa),
                                       discretize(lag(max_phenotype_pairwise_distance, 100)),
                                       discretize(lag(phen_num_taxa, 100))),
  pheno_phylo_10000 =   condinformation(discretize(phen_num_taxa),
                                       discretize(lag(max_phenotype_pairwise_distance, 1000)),
                                       discretize(lag(phen_num_taxa, 1000))),
  pheno_phylo_100000 = condinformation(discretize(phen_num_taxa),
                                       discretize(lag(max_phenotype_pairwise_distance, 10000)),
                                       discretize(lag(phen_num_taxa, 10000))),

  phylo_pheno_10 =      condinformation(discretize(max_phenotype_pairwise_distance),
                                       discretize(lag(phen_num_taxa, 1)),
```

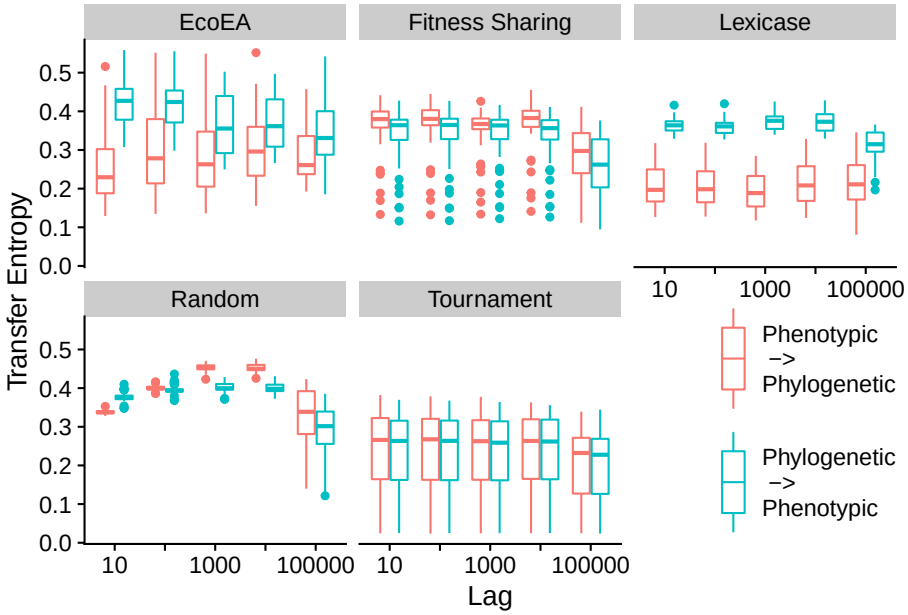
```
                                        discretize(lag(max_phenotype_pairwise_distance,
  phylo_pheno_100 =    condinformation(discretize(max_phenotype_pairwise_distance),
                                        discretize(lag(phen_num_taxa, 10)),
                                        discretize(lag(max_phenotype_pairwise_distance,
  phylo_pheno_1000 =   condinformation(discretize(max_phenotype_pairwise_distance),
                                        discretize(lag(phen_num_taxa, 100)),
                                        discretize(lag(max_phenotype_pairwise_distance,
  phylo_pheno_10000 =  condinformation(discretize(max_phenotype_pairwise_distance),
                                        discretize(lag(phen_num_taxa, 1000)),
                                        discretize(lag(max_phenotype_pairwise_distance,
  phylo_pheno_100000 = condinformation(discretize(max_phenotype_pairwise_distance),
                                        discretize(lag(phen_num_taxa, 10000)),
                                        discretize(lag(max_phenotype_pairwise_distance,
)

# Turn Transfer Entropy columns into rows
res <- res %>% pivot_longer(cols=contains("phylo"))
# Pull lag into a column
res$offset <- str_extract(res$name, "[:digit:]*$")
# Make column indicating direction of transfer entropy
res$Type <- case_when(str_detect(res$name, "phylo_pheno") ~ "\nPhenotypic\n\t->\nPhylog

ggplot(
  res,
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~selection_name) +
  scale_x_discrete("Lag",labels=c("10","","1000","","100000")) +
  scale_y_continuous("Transfer Entropy") +
  theme(legend.position = c(1, 0),
        legend.justification = c(1, 0)) +
  scale_color_discrete("")
```



2.7.2.2 Mean pairwise distance and phenotypic richness

```
res <- data %>% group_by(directory, selection_name) %>%
summarise(
  phen_phylo_10 =       condinformation(discretize(phen_num_taxa),
                                       discretize(lag(mean_phenotype_pairwise_distance, 1)),
                                       discretize(lag(phen_num_taxa, 1))),
  phen_phylo_100 =      condinformation(discretize(phen_num_taxa),
                                       discretize(lag(mean_phenotype_pairwise_distance, 10)),
                                       discretize(lag(phen_num_taxa, 10))),
  pheno_phylo_1000 =    condinformation(discretize(phen_num_taxa),
                                       discretize(lag(mean_phenotype_pairwise_distance, 100)),
                                       discretize(lag(phen_num_taxa, 100))),
  pheno_phylo_10000 =   condinformation(discretize(phen_num_taxa),
                                       discretize(lag(mean_phenotype_pairwise_distance, 1000)),
                                       discretize(lag(phen_num_taxa, 1000))),
  pheno_phylo_100000 = condinformation(discretize(phen_num_taxa),
                                       discretize(lag(mean_phenotype_pairwise_distance, 10000)),
                                       discretize(lag(phen_num_taxa, 10000))),

  phylo_pheno_10 =      condinformation(discretize(mean_phenotype_pairwise_distance),
                                       discretize(lag(phen_num_taxa, 1)),
                                       discretize(lag(mean_phenotype_pairwise_distance, 1))),
  phylo_pheno_100 =     condinformation(discretize(mean_phenotype_pairwise_distance),
```

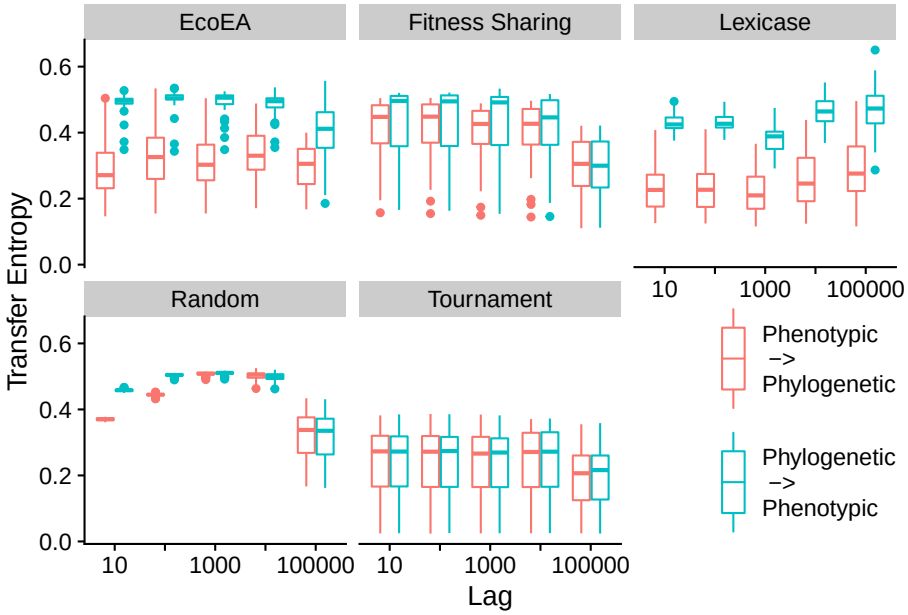
```
                                        discretize(lag(phen_num_taxa, 10)),
                                        discretize(lag(mean_phenotype_pairwise_distance
  phylo_pheno_1000 =   condinformation(discretize(mean_phenotype_pairwise_distance),
                                        discretize(lag(phen_num_taxa, 100)),
                                        discretize(lag(mean_phenotype_pairwise_distance
  phylo_pheno_10000 =  condinformation(discretize(mean_phenotype_pairwise_distance),
                                        discretize(lag(phen_num_taxa, 1000)),
                                        discretize(lag(mean_phenotype_pairwise_distance
  phylo_pheno_100000 = condinformation(discretize(mean_phenotype_pairwise_distance),
                                        discretize(lag(phen_num_taxa, 10000)),
                                        discretize(lag(mean_phenotype_pairwise_distance
)

# Turn Transfer Entropy columns into rows
res <- res %>% pivot_longer(cols=contains("phylo"))
# Pull lag into a column
res$offset <- str_extract(res$name, "[:digit:]*$")
# Make column indicating direction of transfer entropy
res$Type <- case_when(str_detect(res$name, "phylo_pheno") ~ "\nPhenotypic\n\t->\nPhylog

ggplot(
  res,
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~selection_name) +
  scale_x_discrete("Lag",labels=c("10","","1000","","100000")) +
  scale_y_continuous("Transfer Entropy") +
  theme(legend.position = c(1, 0),
        legend.justification = c(1, 0)) +
  scale_color_discrete("")
```



2.7.2.3 Mean pairwise distance and shannon diversity

```
res <- data %>% group_by(directory, selection_name) %>%
summarise(
  phen_phylo_10 =       condinformation(discretize(phen_diversity),
                                       discretize(lag(mean_phenotype_pairwise_distance, 1)),
                                       discretize(lag(phen_diversity, 1))),
  phen_phylo_100 =      condinformation(discretize(phen_diversity),
                                       discretize(lag(mean_phenotype_pairwise_distance, 10)),
                                       discretize(lag(phen_diversity, 10))),
  pheno_phylo_1000 =    condinformation(discretize(phen_diversity),
                                       discretize(lag(mean_phenotype_pairwise_distance, 100)),
                                       discretize(lag(phen_diversity, 100))),
  pheno_phylo_10000 =   condinformation(discretize(phen_diversity),
                                       discretize(lag(mean_phenotype_pairwise_distance, 1000)),
                                       discretize(lag(phen_diversity, 1000))),
  pheno_phylo_100000 = condinformation(discretize(phen_diversity),
                                       discretize(lag(mean_phenotype_pairwise_distance, 10000)),
                                       discretize(lag(phen_diversity, 10000))),

  phylo_pheno_10 =      condinformation(discretize(mean_phenotype_pairwise_distance),
                                       discretize(lag(phen_diversity, 1)),
                                       discretize(lag(mean_phenotype_pairwise_distance, 1))),
  phylo_pheno_100 =     condinformation(discretize(mean_phenotype_pairwise_distance),
```

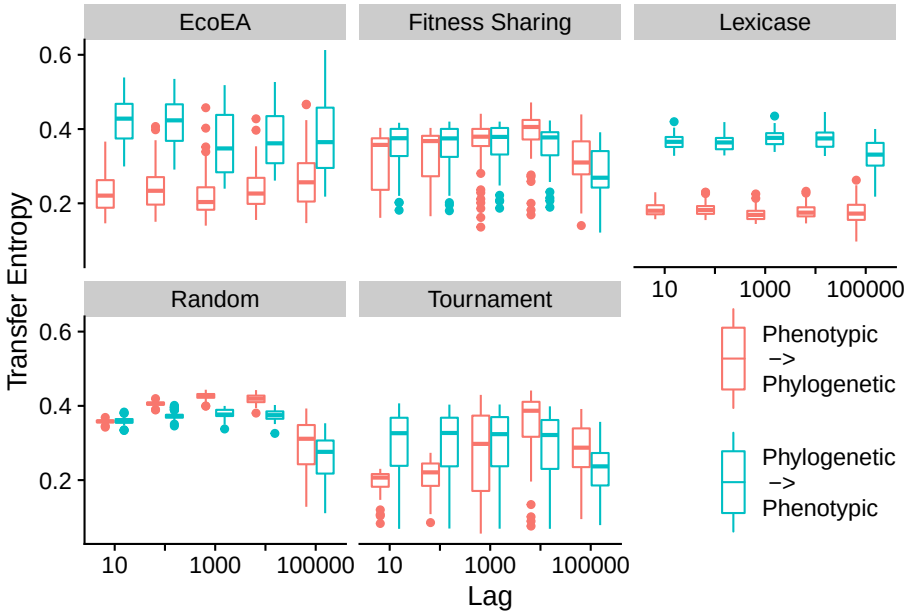
```
                                        discretize(lag(phen_diversity, 10)),
                                        discretize(lag(mean_phenotype_pairwise_distance
  phylo_pheno_1000 =   condinformation(discretize(mean_phenotype_pairwise_distance),
                                        discretize(lag(phen_diversity, 100)),
                                        discretize(lag(mean_phenotype_pairwise_distance
  phylo_pheno_10000 =  condinformation(discretize(mean_phenotype_pairwise_distance),
                                        discretize(lag(phen_diversity, 1000)),
                                        discretize(lag(mean_phenotype_pairwise_distance
  phylo_pheno_100000 = condinformation(discretize(mean_phenotype_pairwise_distance),
                                        discretize(lag(phen_diversity, 10000)),
                                        discretize(lag(mean_phenotype_pairwise_distance
)

# Turn Transfer Entropy columns into rows
res <- res %>% pivot_longer(cols=contains("phylo"))
# Pull lag into a column
res$offset <- str_extract(res$name, "[:digit:]*$")
# Make column indicating direction of transfer entropy
res$Type <- case_when(str_detect(res$name, "phylo_pheno") ~ "\nPhenotypic\n\t->\nPhylog

ggplot(
  res,
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~selection_name) +
  scale_x_discrete("Lag",labels=c("10","","1000","","100000")) +
  scale_y_continuous("Transfer Entropy") +
  theme(legend.position = c(1, 0),
        legend.justification = c(1, 0)) +
  scale_color_discrete("")
```



2.7.2.4 Mean evolutionary distinctiveness and phenotypic richness

```
res <- data %>% group_by(directory, selection_name) %>%
summarise(
  phen_phylo_10 =       condinformation(
                           discretize(phen_num_taxa),
                           discretize(lag(mean_phenotype_evolutionary_distinctiveness, 1)),
                           discretize(lag(phen_num_taxa, 1))),
  phen_phylo_100 =      condinformation(
                           discretize(phen_num_taxa),
                           discretize(lag(mean_phenotype_evolutionary_distinctiveness, 10)),
                           discretize(lag(phen_num_taxa, 10))),
  pheno_phylo_1000 =    condinformation(
                           discretize(phen_num_taxa),
                           discretize(lag(mean_phenotype_evolutionary_distinctiveness, 100)),
                           discretize(lag(phen_num_taxa, 100))),
  pheno_phylo_10000 =   condinformation(
                           discretize(phen_num_taxa),
                           discretize(lag(mean_phenotype_evolutionary_distinctiveness, 1000)),
                           discretize(lag(phen_num_taxa, 1000))),
  pheno_phylo_100000 = condinformation(
                           discretize(phen_num_taxa),
                           discretize(lag(mean_phenotype_evolutionary_distinctiveness, 10000)),
                           discretize(lag(phen_num_taxa, 10000))),
```

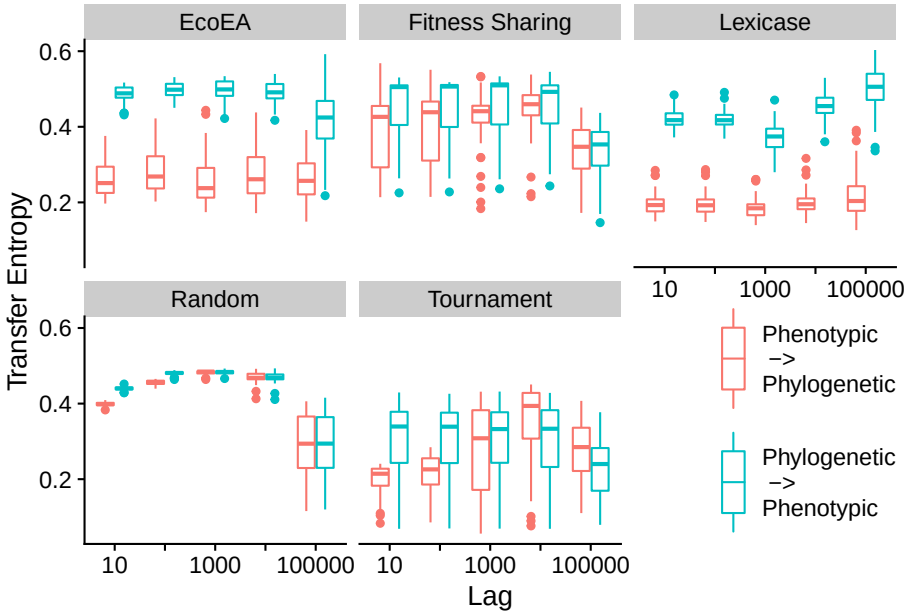
```
  phylo_pheno_10 =     condinformation(
                          discretize(mean_phenotype_evolutionary_distinctiveness),
                          discretize(lag(phen_num_taxa, 1)),
                          discretize(lag(mean_phenotype_evolutionary_distinctiveness,
  phylo_pheno_100 =    condinformation(
                          discretize(mean_phenotype_evolutionary_distinctiveness),
                          discretize(lag(phen_num_taxa, 10)),
                          discretize(lag(mean_phenotype_evolutionary_distinctiveness,
  phylo_pheno_1000 =   condinformation(
                          discretize(mean_phenotype_evolutionary_distinctiveness),
                          discretize(lag(phen_num_taxa, 100)),
                          discretize(lag(mean_phenotype_evolutionary_distinctiveness,
  phylo_pheno_10000 =  condinformation(
                          discretize(mean_phenotype_evolutionary_distinctiveness),
                          discretize(lag(phen_num_taxa, 1000)),
                          discretize(lag(mean_phenotype_evolutionary_distinctiveness,
  phylo_pheno_100000 = condinformation(
                          discretize(mean_phenotype_evolutionary_distinctiveness),
                          discretize(lag(phen_num_taxa, 10000)),
                          discretize(lag(mean_phenotype_evolutionary_distinctiveness,
)

# Turn Transfer Entropy columns into rows
res <- res %>% pivot_longer(cols=contains("phylo"))
# Pull lag into a column
res$offset <- str_extract(res$name, "[:digit:]*$")
# Make column indicating direction of transfer entropy
res$Type <- case_when(str_detect(res$name, "phylo_pheno") ~ "\nPhenotypic\n\t->\nPhylog

ggplot(
  res,
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~selection_name) +
  scale_x_discrete("Lag",labels=c("10","","1000","","100000")) +
  scale_y_continuous("Transfer Entropy") +
  theme(legend.position = c(1, 0),
        legend.justification = c(1, 0)) +
  scale_color_discrete("")
```



2.7.2.5 Mean evolutionary distinctiveness and shannon diversity

```
res <- data %>% group_by(directory, selection_name) %>%
summarise(
  phen_phylo_10 =       condinformation(
                           discretize(phen_diversity),
                           discretize(lag(mean_phenotype_evolutionary_distinctiveness, 1)),
                           discretize(lag(phen_diversity, 1))),
  phen_phylo_100 =      condinformation(
                           discretize(phen_diversity),
                           discretize(lag(mean_phenotype_evolutionary_distinctiveness, 10)),
                           discretize(lag(phen_diversity, 10))),
  pheno_phylo_1000 =    condinformation(
                           discretize(phen_diversity),
                           discretize(lag(mean_phenotype_evolutionary_distinctiveness, 100)),
                           discretize(lag(phen_diversity, 100))),
  pheno_phylo_10000 =   condinformation(
                           discretize(phen_diversity),
                           discretize(lag(mean_phenotype_evolutionary_distinctiveness, 1000)),
                           discretize(lag(phen_diversity, 1000))),
  pheno_phylo_100000 = condinformation(
                           discretize(phen_diversity),
                           discretize(lag(mean_phenotype_evolutionary_distinctiveness, 10000)),
                           discretize(lag(phen_diversity, 10000))),
```

```
  phylo_pheno_10 =     condinformation(
                          discretize(mean_phenotype_evolutionary_distinctiveness),
                          discretize(lag(phen_diversity, 1)),
                          discretize(lag(mean_phenotype_evolutionary_distinctiveness,
  phylo_pheno_100 =    condinformation(
                          discretize(mean_phenotype_evolutionary_distinctiveness),
                          discretize(lag(phen_diversity, 10)),
                          discretize(lag(mean_phenotype_evolutionary_distinctiveness,
  phylo_pheno_1000 =   condinformation(
                          discretize(mean_phenotype_evolutionary_distinctiveness),
                          discretize(lag(phen_diversity, 100)),
                          discretize(lag(mean_phenotype_evolutionary_distinctiveness,
  phylo_pheno_10000 =  condinformation(
                          discretize(mean_phenotype_evolutionary_distinctiveness),
                          discretize(lag(phen_diversity, 1000)),
                          discretize(lag(mean_phenotype_evolutionary_distinctiveness,
  phylo_pheno_100000 = condinformation(
                          discretize(mean_phenotype_evolutionary_distinctiveness),
                          discretize(lag(phen_diversity, 10000)),
                          discretize(lag(mean_phenotype_evolutionary_distinctiveness,
)

# Turn Transfer Entropy columns into rows
res <- res %>% pivot_longer(cols=contains("phylo"))
# Pull lag into a column
res$offset <- str_extract(res$name, "[:digit:]*$")
# Make column indicating direction of transfer entropy
res$Type <- case_when(str_detect(res$name, "phylo_pheno") ~ "\nPhenotypic\n\t->\nPhylog

ggplot(
  res,
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~selection_name) +
  scale_x_discrete("Lag",labels=c("10","","1000","","100000")) +
  scale_y_continuous("Transfer Entropy") +
  theme(legend.position = c(1, 0),
        legend.justification = c(1, 0)) +
  scale_color_discrete("")
```


EcoEA
Fitness Sharing
Lexicase
Random
Tournament
Transfer Entropy
0.6
0.4
0.2
10
1000
100000
Lag
Phenotypic -> Phylogenetic
Phylogenetic -> Phenotypic

Chapter 3

Other fitness landscapes

3.1 Setup

```
library(ggplot2)
library(tidyverse)
library(knitr)
library(cowplot)
library(viridis)
library(RColorBrewer)
library(rstatix)
library(ggsignif)
library(Hmisc)
library(kableExtra)
source("https://gist.githubusercontent.com/benmarwick/2a1bb0133ff568cbe28d/raw/fb53bd97121f7f9ce9
library(readr)
library(stringr)
library(ggpubr)
library(infotheo)
library(osfr)
```

These analyses were conducted in the following computing environment:

```
print(version)
```

```
##                _
## platform       x86_64-pc-linux-gnu
## arch           x86_64
## os             linux-gnu
## system         x86_64, linux-gnu
## status
```

```
## major          4
## minor          0.4
## year           2021
## month          02
## day            15
## svn rev        80002
## language       R
## version.string R version 4.0.4 (2021-02-15)
## nickname       Lost Library Book
```

```
# Labeler for stats annotations
p_label <- function(p_value) {
  threshold = 0.0001
  if (p_value < threshold) {
    return(paste0("p < ", threshold))
  } else {
    return(paste0("p = ", p_value))
  }
}

# Significance threshold
alpha <- 0.05

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

```
osf_retrieve_file("p79hx") %>% osf_download(conflicts = "skip")  # Download data from
```

```
## # A tibble: 1 x 4
##   name                   id                local_path                meta
##   <chr>                  <chr>             <chr>                     <list>
## 1 complex_fitness_lands~ 612fe4d84e5ee501~ ./complex_fitness_lands~ <named list~
```

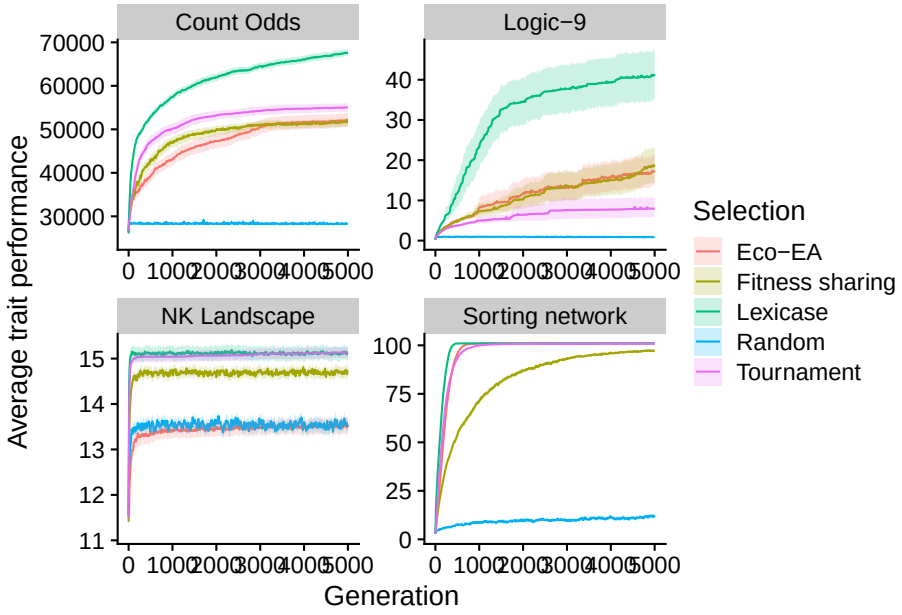
```
data_loc <- "complex_fitness_landscapes.csv"

data <- read_csv(data_loc, na=c("NONE", "NA", ""))

data <- data %>%
  filter(N==20, generation %%10 == 0) %>%
  mutate(
  selection_name = as.factor(case_when(
    SELECTION == 0 ~ "Tournament",
    SELECTION == 1 ~ "Fitness sharing",
    SELECTION == 2 ~ "Lexicase",
    SELECTION == 3 ~ "Eco-EA",
    SELECTION == 4 ~ "Random",
```

```
  )),
  problem_name = as.factor(case_when(
    PROBLEM == 0 ~ "NK Landscape",
    PROBLEM == 1 ~ "Count Odds",
    PROBLEM == 2 ~ "Real-valued optimization",
    PROBLEM == 3 ~ "Sorting network",
    PROBLEM == 4 ~ "Logic-9"
  ))
)

final_data <- filter(data, generation==max(data$generation))
```

3.2 Performance

3.2.1 Over time

```
ggplot(
    data,
    aes(
      x=generation,
      y=max_performance,
      color=selection_name,
      fill=selection_name
    )
  ) +
  stat_summary(geom="line", fun=mean) +
  stat_summary(
    geom="ribbon",
    fun.data="mean_cl_boot",
    fun.args=list(conf.int=0.95),
    alpha=0.2,
    linetype=0
  ) +
  scale_y_continuous(
    name="Average trait performance"
  ) +
  scale_x_continuous(
    name="Generation"
  ) +
  scale_color_discrete("Selection") +
  scale_fill_discrete("Selection") +
  facet_wrap(~problem_name, scales="free")
```

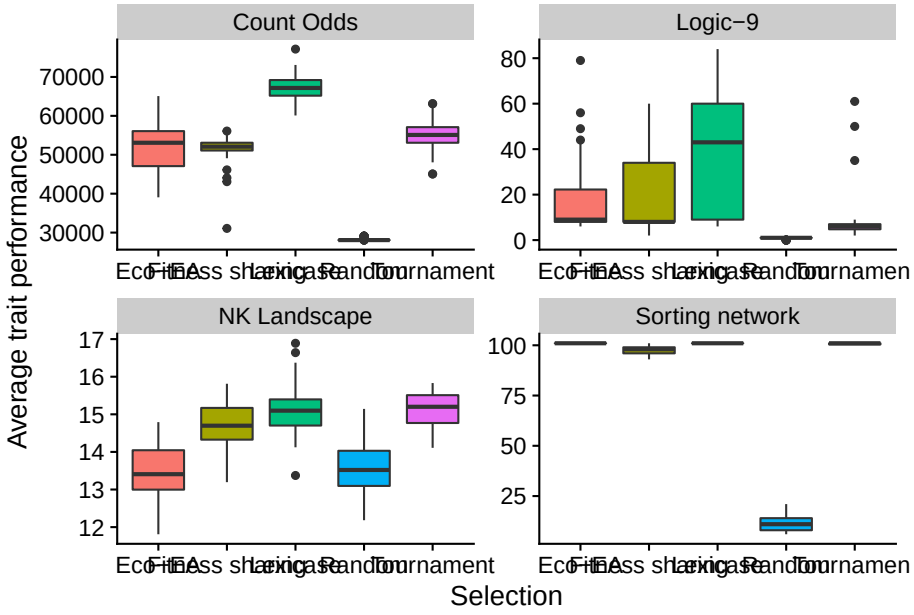


3.2.2 Final

```
# Compute manual labels for geom_signif
stat.test <- final_data %>%
  wilcox_test(max_performance ~ selection_name) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="selection_name",step.increase=1)
#stat.test$manual_position <- stat.test$y.position * .5
#stat.test$manual_position <- c(110, 150, 170, 170, 130, 110)
stat.test$label <- mapply(p_label,stat.test$p.adj)
```

```
ggplot(
    final_data,
    aes(
      x=selection_name,
      y=max_performance,
      fill=selection_name
    )
  ) +
  geom_boxplot() +
  scale_y_continuous(
    name="Average trait performance"
  ) +
```

```
  scale_x_discrete(
    name="Selection"
  ) +
  scale_fill_discrete(
    name="Selection"
  ) +
  scale_color_discrete(
    name="Selection"
  ) +
  theme(legend.position="none") +
  facet_wrap(~problem_name, scales="free")
```

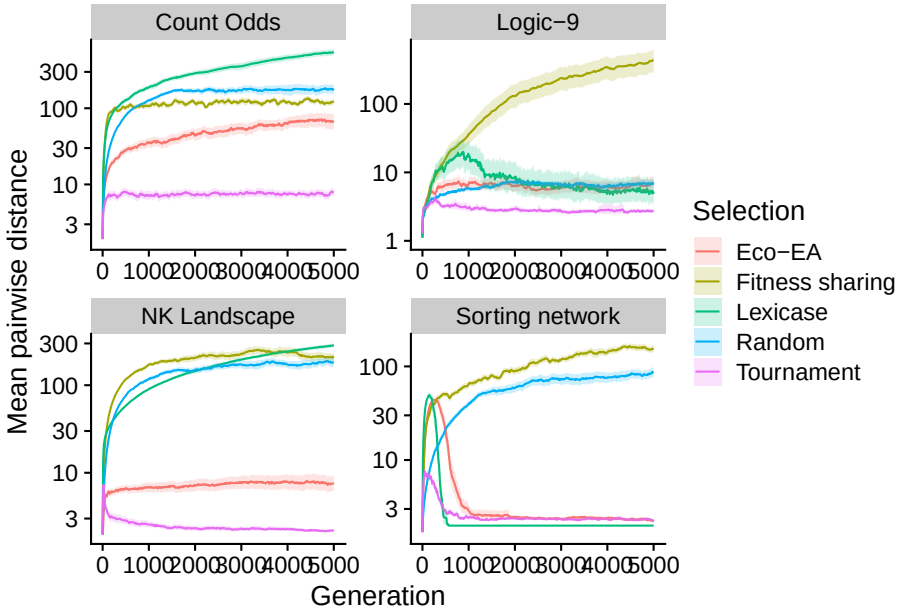



```
stat.test %>%
  kbl() %>%
  kable_styling(
    bootstrap_options = c(
      "striped",
      "hover",
      "condensed",
      "responsive"
    )
  ) %>%
  scroll_box(width="600px")
```

.y.	group1	group2	n1	n2	statistic	p	p.adj
max_performance	Eco-EA	Fitness sharing	240	240	29310.5	7.37e-01	1.000000
max_performance	Eco-EA	Lexicase	240	240	23075.0	1.46e-04	0.001460
max_performance	Eco-EA	Random	240	240	39696.5	0.00e+00	0.000000
max_performance	Eco-EA	Tournament	240	240	29346.0	7.18e-01	1.000000
max_performance	Fitness sharing	Lexicase	240	240	22328.5	2.01e-05	0.000201
max_performance	Fitness sharing	Random	240	240	41293.0	0.00e+00	0.000000
max_performance	Fitness sharing	Tournament	240	240	27500.5	3.92e-01	1.000000
max_performance	Lexicase	Random	240	240	44132.5	0.00e+00	0.000000
max_performance	Lexicase	Tournament	240	240	34831.5	6.61e-05	0.000661
max_performance	Random	Tournament	240	240	18254.0	0.00e+00	0.000000

3.3 Phylogenetic diversity

```
ggplot(
    data,
    aes(
      x=generation,
      y=mean_phenotype_pairwise_distance,
      color=selection_name,
      fill=selection_name
    )
  ) +
  stat_summary(geom="line", fun=mean) +
  stat_summary(
    geom="ribbon",
    fun.data="mean_cl_boot",
    fun.args=list(conf.int=0.95),
    alpha=0.2,
    linetype=0
  ) +
  scale_y_log10(
    name="Mean pairwise distance"
  ) +
  scale_x_continuous(
    name="Generation"
  ) +
  scale_color_discrete("Selection") +
  scale_fill_discrete("Selection") +
  facet_wrap(~problem_name, scales = "free")
```

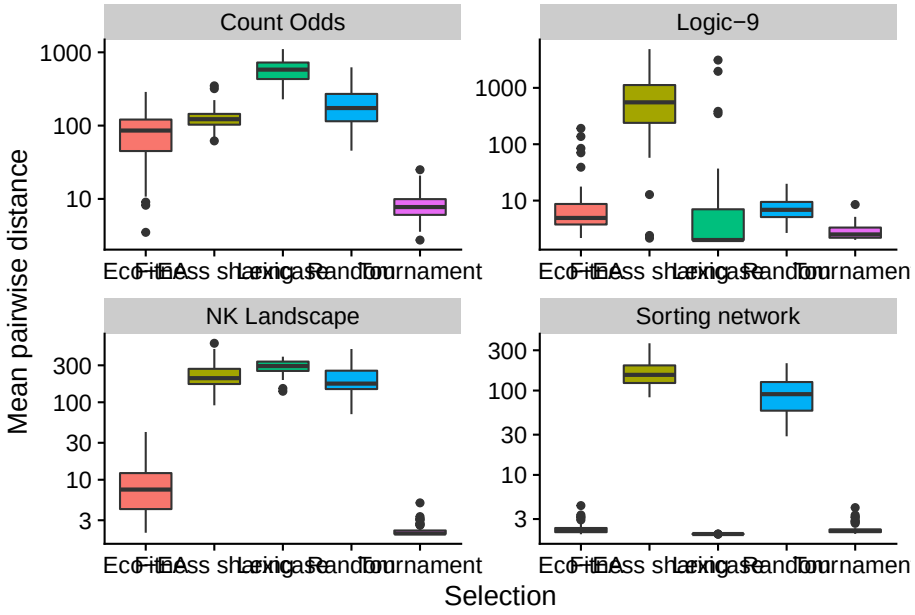


3.3.2 Final

```
# Compute manual labels for geom_signif
stat.test <- final_data %>%
  wilcox_test(mean_phenotype_pairwise_distance ~ selection_name) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="selection_name",step.increase=1)
#stat.test$manual_position <- stat.test$y.position * .5
#stat.test$manual_position <- c(110, 150, 170, 170, 130, 110)
stat.test$label <- mapply(p_label,stat.test$p.adj)
```

```
ggplot(
    final_data,
    aes(
      x=selection_name,
      y=mean_phenotype_pairwise_distance,
      fill=selection_name
    )
  ) +
  geom_boxplot() +
  scale_y_log10(
    name="Mean pairwise distance"
  ) +
```

```
  scale_x_discrete(
    name="Selection"
  ) +
  scale_fill_discrete(
    name="Selection"
  ) +
  scale_color_discrete(
    name="Selection"
  ) +
  theme(legend.position = "none") +
    facet_wrap(~problem_name, scales = "free")
```

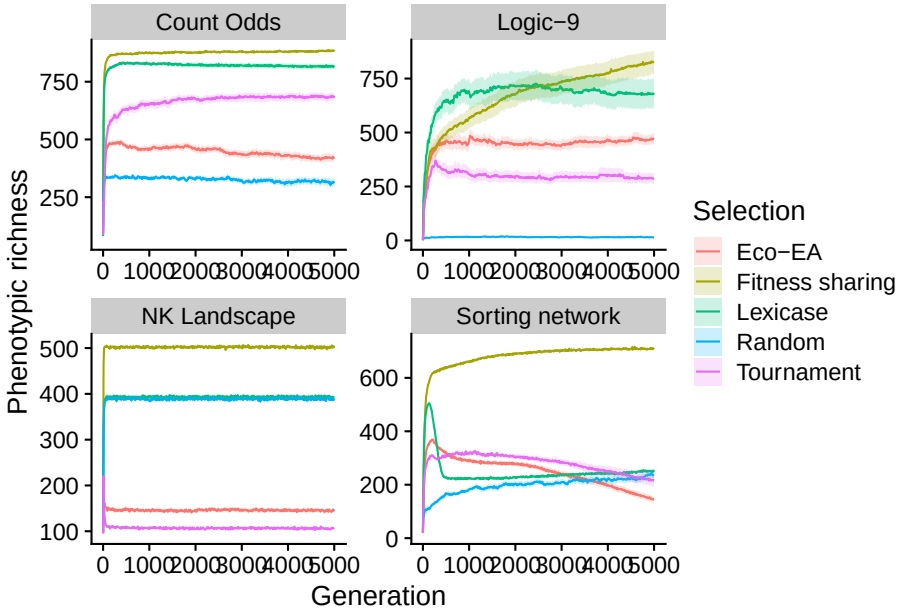



```
stat.test %>%
  kbl() %>%
  kable_styling(
    bootstrap_options = c(
      "striped",
      "hover",
      "condensed",
      "responsive"
    )
  ) %>%
  scroll_box(width="600px")
```

.y.	group1	group2	n1	n2	statistic	p	
mean_phenotype_pairwise_distance	Eco-EA	Fitness sharing	240	240	2831.0	0.00e+00	0.00
mean_phenotype_pairwise_distance	Eco-EA	Lexicase	240	240	25159.0	1.70e-02	0.17
mean_phenotype_pairwise_distance	Eco-EA	Random	240	240	9552.0	0.00e+00	0.00
mean_phenotype_pairwise_distance	Eco-EA	Tournament	240	240	43052.5	0.00e+00	0.00
mean_phenotype_pairwise_distance	Fitness sharing	Lexicase	240	240	33683.0	1.00e-03	0.01
mean_phenotype_pairwise_distance	Fitness sharing	Random	240	240	40738.0	0.00e+00	0.00
mean_phenotype_pairwise_distance	Fitness sharing	Tournament	240	240	57314.0	0.00e+00	0.00
mean_phenotype_pairwise_distance	Lexicase	Random	240	240	28272.0	8.49e-01	1.00
mean_phenotype_pairwise_distance	Lexicase	Tournament	240	240	34801.0	7.84e-05	0.00
mean_phenotype_pairwise_distance	Random	Tournament	240	240	54927.0	0.00e+00	0.00

3.4 Phenotypic diversity

```
ggplot(
    data,
    aes(
      x=generation,
      y=phenotype_num_taxa,
      color=selection_name,
      fill=selection_name
    )
  ) +
  stat_summary(geom="line", fun=mean) +
  stat_summary(
    geom="ribbon",
    fun.data="mean_cl_boot",
    fun.args=list(conf.int=0.95),
    alpha=0.2,
    linetype=0
  ) +
  scale_y_continuous(
    name="Phenotypic richness"
  ) +
  scale_x_continuous(
    name="Generation"
  ) +
  scale_color_discrete("Selection") +
  scale_fill_discrete("Selection") +
  facet_wrap(~problem_name, scales = "free")
```

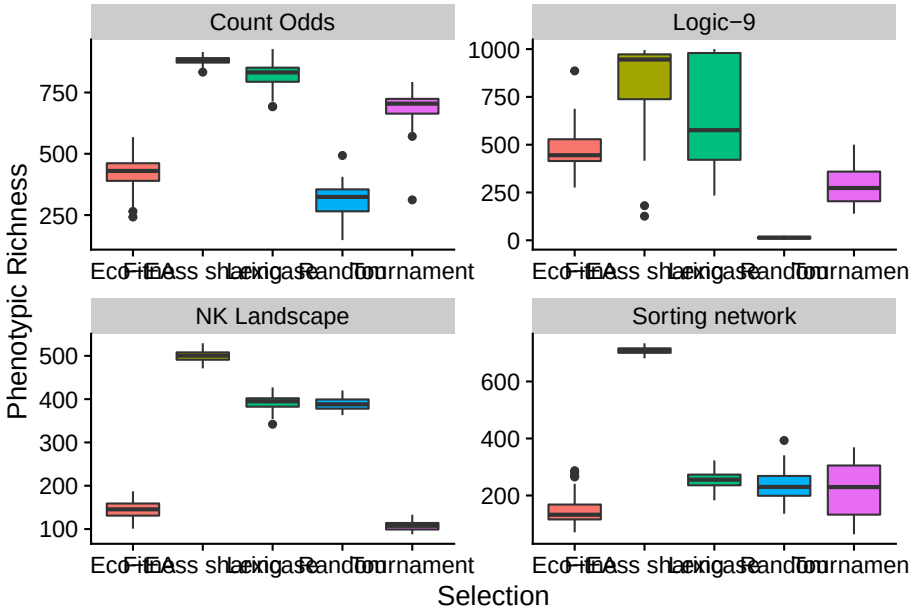


3.4.2 Final

```
# Compute manual labels for geom_signif
stat.test <- final_data %>%
  wilcox_test(phenotype_num_taxa ~ selection_name) %>%
  adjust_pvalue(method = "bonferroni") %>%
  add_significance() %>%
  add_xy_position(x="selection_name",step.increase=1)
#stat.test$manual_position <- stat.test$y.position * .5
#stat.test$manual_position <- c(110, 150, 170, 170, 130, 110)
stat.test$label <- mapply(p_label,stat.test$p.adj)
```

```
ggplot(
    final_data,
    aes(
      x=selection_name,
      y=phenotype_num_taxa,
      fill=selection_name
    )
  ) +
  geom_boxplot() +
  scale_y_continuous(
    name="Phenotypic Richness"
  ) +
```

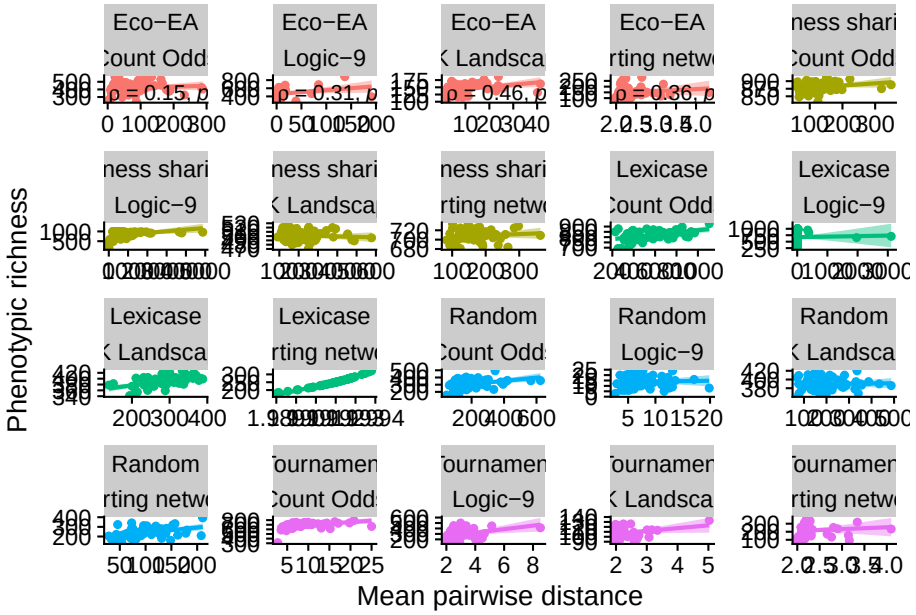
```
  scale_x_discrete(
    name="Selection"
  ) +
  scale_fill_discrete(
    name="Selection"
  ) +
  scale_color_discrete(
    name="Selection"
  ) +
  theme(legend.position = "none") +
  facet_wrap(~problem_name, scales = "free")
```




```
stat.test %>%
  kbl() %>%
  kable_styling(
    bootstrap_options = c(
      "striped",
      "hover",
      "condensed",
      "responsive"
    )
  ) %>%
  scroll_box(width="600px")
```

.y.	group1	group2	n1	n2	statistic	p	p.ad
phenotype_num_taxa	Eco-EA	Fitness sharing	240	240	2406.0	0.000000	0.0000
phenotype_num_taxa	Eco-EA	Lexicase	240	240	14833.5	0.000000	0.0000
phenotype_num_taxa	Eco-EA	Random	240	240	34261.5	0.000326	0.0032
phenotype_num_taxa	Eco-EA	Tournament	240	240	29587.5	0.604000	1.0000
phenotype_num_taxa	Fitness sharing	Lexicase	240	240	42195.5	0.000000	0.0000
phenotype_num_taxa	Fitness sharing	Random	240	240	57231.0	0.000000	0.0000
phenotype_num_taxa	Fitness sharing	Tournament	240	240	51342.0	0.000000	0.0000
phenotype_num_taxa	Lexicase	Random	240	240	47218.0	0.000000	0.0000
phenotype_num_taxa	Lexicase	Tournament	240	240	43850.0	0.000000	0.0000
phenotype_num_taxa	Random	Tournament	240	240	25692.5	0.041000	0.4100

```
ggplot(
    final_data,
    aes(
        y=phenotype_num_taxa,
        x=mean_phenotype_pairwise_distance,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Phenotypic richness"
  ) +
  scale_x_continuous(
        name="Mean pairwise distance"
  ) +
  facet_wrap(
      ~selection_name*problem_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```

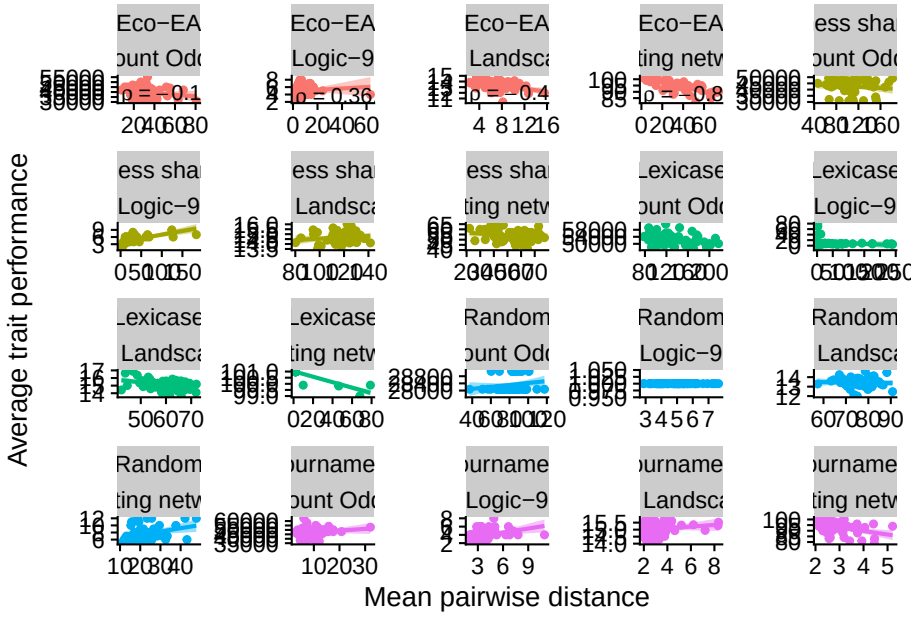


3.6 Relationship between diversity and success

3.6.1 Earlier in run

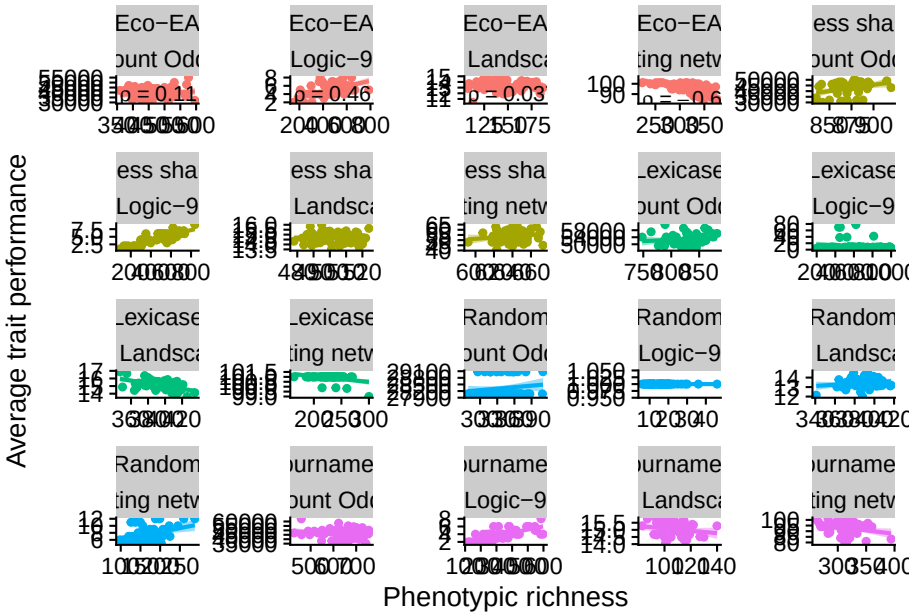
```
ggplot(
    data %>% filter(generation==500),
    aes(
        y=max_performance,
        x=mean_phenotype_pairwise_distance,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Average trait performance"
  ) +
  scale_x_continuous(
        name="Mean pairwise distance"
  ) +
  facet_wrap(
      ~selection_name*problem_name, scales="free"
  ) +
  stat_smooth(
```

```
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```



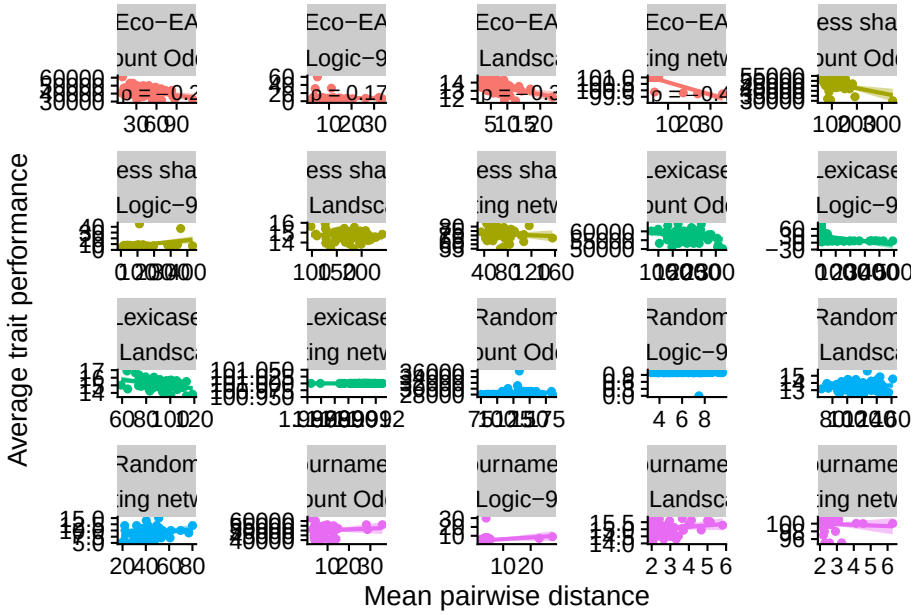

```
ggplot(
    data %>% filter(generation==500),
    aes(
        y=max_performance,
        x=phenotype_num_taxa,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Average trait performance"
  ) +
  scale_x_continuous(
        name="Phenotypic richness"
  ) +
  facet_wrap(
      ~selection_name*problem_name, scales="free"
```

```
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```



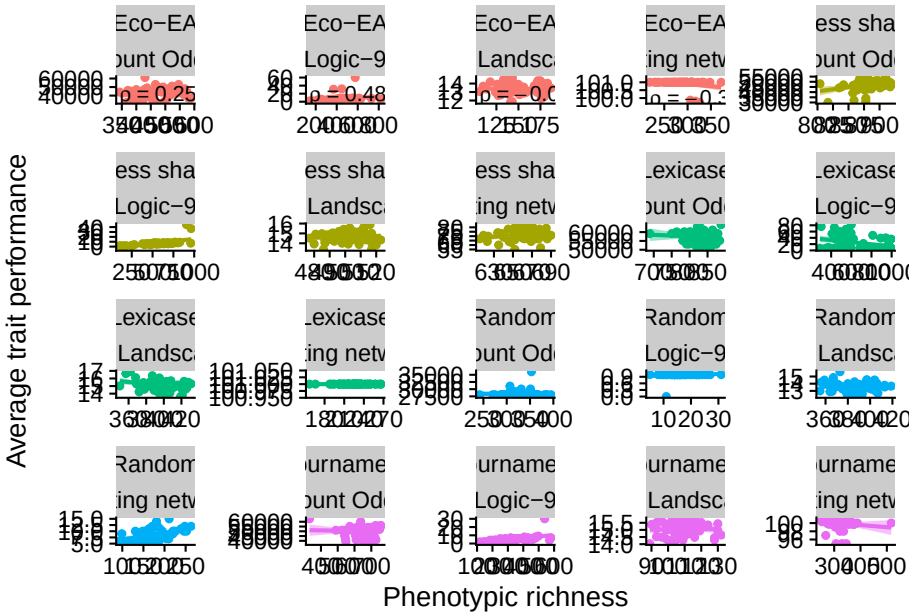

```
phylogney_vs_performance <- ggplot(
    data %>% filter(generation==1000),
    aes(
        y=max_performance,
        x=mean_phenotype_pairwise_distance,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Average trait performance"
  ) +
  scale_x_continuous(
        name="Mean pairwise distance"
  ) +
```

```
  facet_wrap(
      ~selection_name*problem_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")

phylogney_vs_performance
```




```
ggplot(
    data %>% filter(generation==1000),
    aes(
        y=max_performance,
        x=phenotype_num_taxa,
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Average trait performance"
```

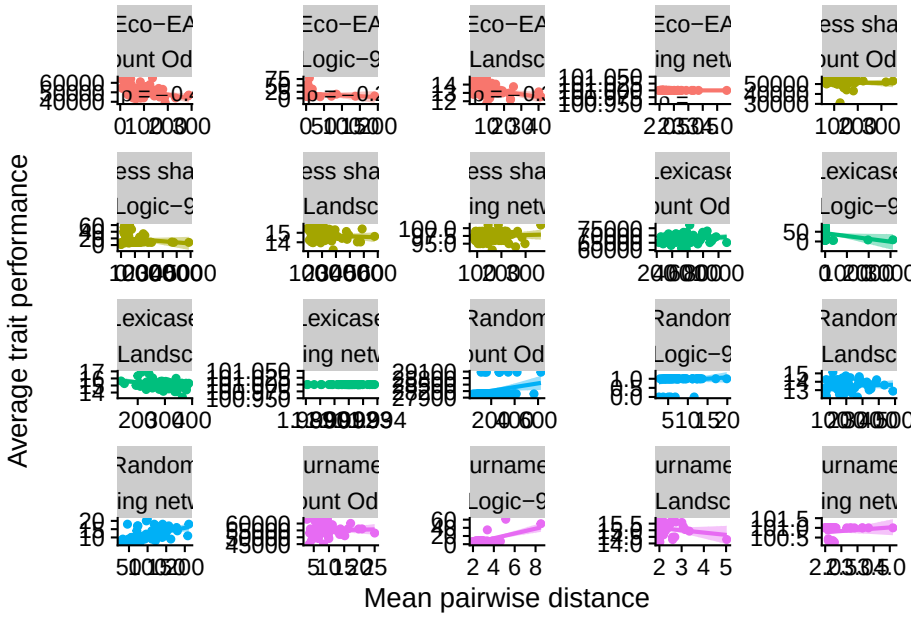
```
  ) +
  scale_x_continuous(
        name="Phenotypic richness"
  ) +
  facet_wrap(
      ~selection_name*problem_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```



3.6.2 End of run

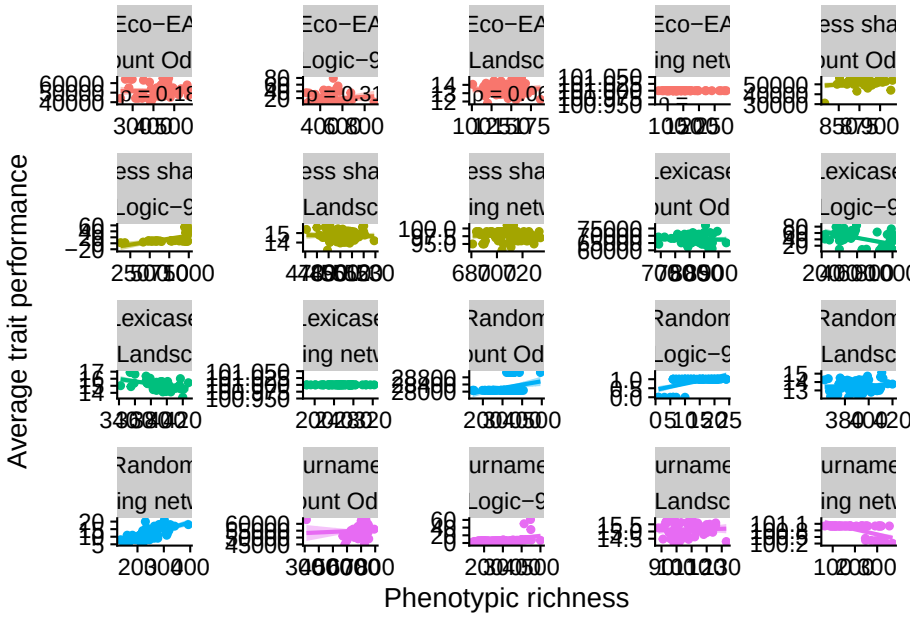
```
ggplot(
    final_data,
    aes(
        y=max_performance,
        x=mean_phenotype_pairwise_distance,
        color=selection_name,
        fill=selection_name
```

```
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Average trait performance"
  ) +
  scale_x_continuous(
        name="Mean pairwise distance"
  ) +
  facet_wrap(
      ~selection_name*problem_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```



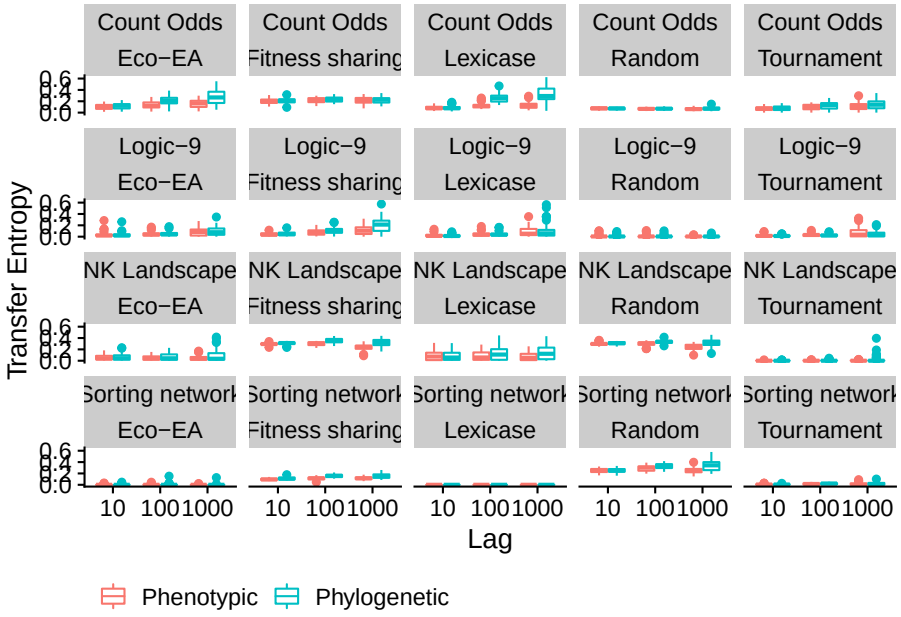

```
ggplot(
    final_data,
    aes(
        y=max_performance,
        x=phenotype_num_taxa,
```

```
        color=selection_name,
        fill=selection_name
    )
  ) +
  geom_point() +
    scale_y_continuous(
        name="Average trait performance"
  ) +
  scale_x_continuous(
        name="Phenotypic richness"
  ) +
  facet_wrap(
      ~selection_name*problem_name, scales="free"
  ) +
  stat_smooth(
    method="lm"
  ) +
  stat_cor(
    method="spearman", cor.coef.name = "rho", color="black"
  ) +
  theme(legend.position = "none")
```



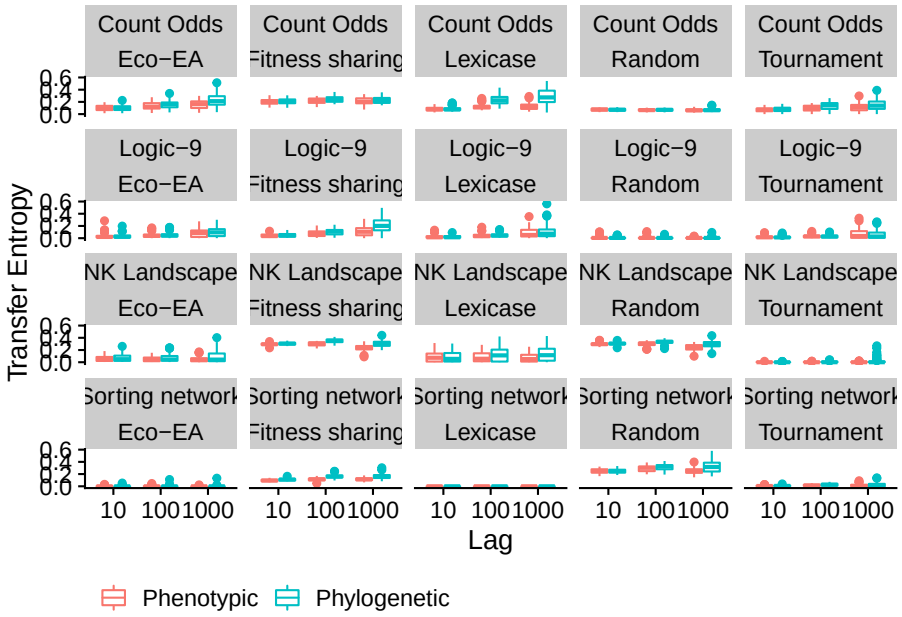
3.7 Causality analysis

```
res <- data %>% group_by(SEED, selection_name, problem_name) %>%
summarise(
  fit_phylo_10 = condinformation(discretize(max_performance), discretize(lag(max_phenot
  fit_phylo_100 = condinformation(discretize(max_performance), discretize(lag(max_pheno
  fit_phylo_1000 = condinformation(discretize(max_performance), discretize(lag(max_phen
    fit_pheno_10 = condinformation(discretize(max_performance), discretize(lag(phenotyp
      fit_pheno_100 = condinformation(discretize(max_performance), discretize(lag(pheno
      fit_pheno_1000 = condinformation(discretize(max_performance), discretize(lag(phen
      )

res <- res %>% pivot_longer(cols=contains("o_10"))
res$offset <- str_extract(res$name, "[:digit:]*$")
res$Type <- case_when(str_detect(res$name, "phylo") ~ "Phylogenetic", TRUE ~ "Phenotypi

ggplot(
  res %>% filter(str_detect(name, "fit_ph*")),
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~problem_name*selection_name) +
  scale_x_discrete("Lag") +
  scale_y_continuous("Transfer Entropy") +
  scale_color_discrete("") +
  theme(legend.position = "bottom")
```



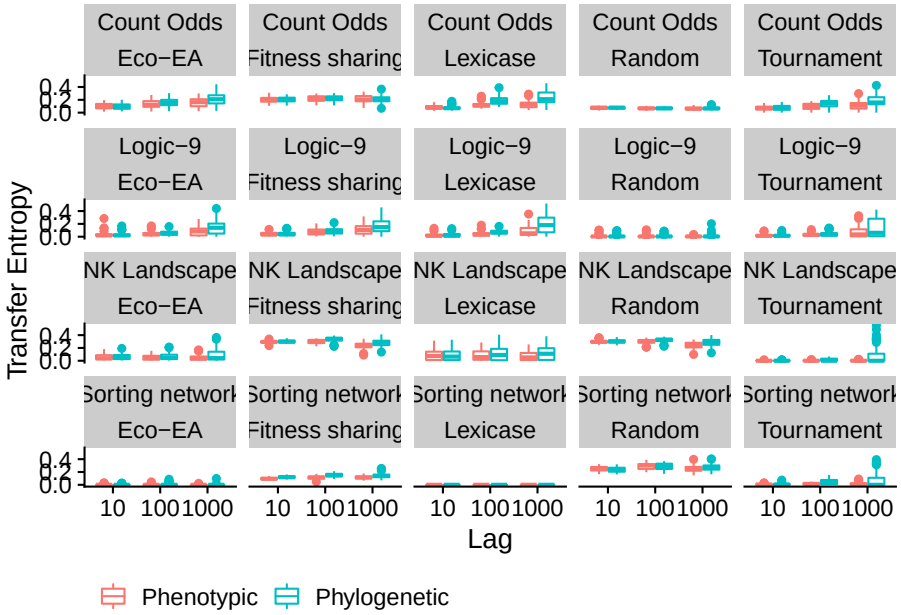

```
res <- data %>% group_by(SEED, selection_name, problem_name) %>%
summarise(
  fit_phylo_10 = condinformation(discretize(max_performance), discretize(lag(mean_phenotype_pairw
  fit_phylo_100 = condinformation(discretize(max_performance), discretize(lag(mean_phenotype_pair
  fit_phylo_1000 = condinformation(discretize(max_performance), discretize(lag(mean_phenotype_pai
    fit_pheno_10 = condinformation(discretize(max_performance), discretize(lag(phenotype_num_taxa
      fit_pheno_100 = condinformation(discretize(max_performance), discretize(lag(phenotype_num_t
      fit_pheno_1000 = condinformation(discretize(max_performance), discretize(lag(phenotype_num_
      )

res <- res %>% pivot_longer(cols=contains("o_10"))
res$offset <- str_extract(res$name, "[:digit:]*$")
res$Type <- case_when(str_detect(res$name, "phylo") ~ "Phylogenetic", TRUE ~ "Phenotypic")

ggplot(
  res %>% filter(str_detect(name, "fit_ph*")),
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~problem_name*selection_name) +
  scale_x_discrete("Lag") +
```

```
  scale_y_continuous("Transfer Entropy") +
  scale_color_discrete("") +
  theme(legend.position = "bottom")
```



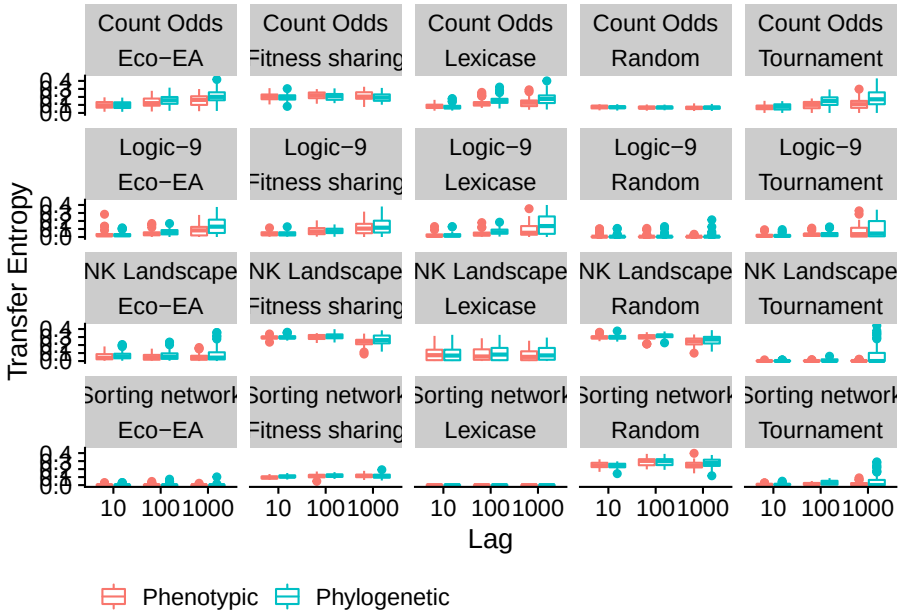

```
res <- data %>% group_by(SEED, selection_name, problem_name) %>%
summarise(
  fit_phylo_10 = condinformation(discretize(max_performance), discretize(lag(mean_phen
  fit_phylo_100 = condinformation(discretize(max_performance), discretize(lag(mean_phe
  fit_phylo_1000 = condinformation(discretize(max_performance), discretize(lag(mean_ph
    fit_pheno_10 = condinformation(discretize(max_performance), discretize(lag(phenoty
      fit_pheno_100 = condinformation(discretize(max_performance), discretize(lag(phen
      fit_pheno_1000 = condinformation(discretize(max_performance), discretize(lag(phe
      )

res <- res %>% pivot_longer(cols=contains("o_10"))
res$offset <- str_extract(res$name, "[:digit:]*$")
res$Type <- case_when(str_detect(res$name, "phylo") ~ "Phylogenetic", TRUE ~ "Phenotyp

ggplot(
  res %>% filter(str_detect(name, "fit_ph*")),
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
```

```
  ) +
  geom_boxplot() +
  facet_wrap(~problem_name*selection_name) +
  scale_x_discrete("Lag") +
  scale_y_continuous("Transfer Entropy") +
  scale_color_discrete("") +
  theme(legend.position = "bottom")
```



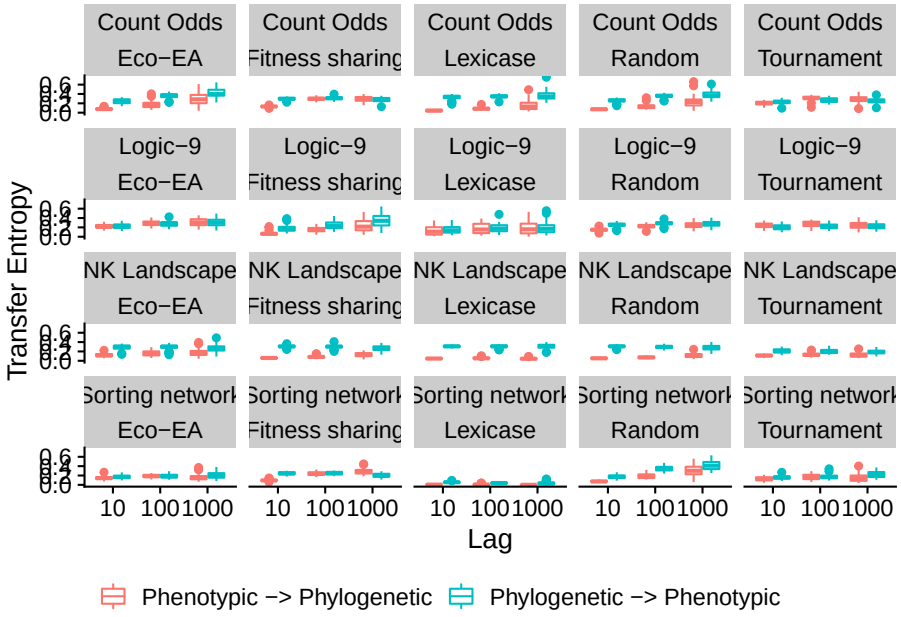

```
res <- data %>% group_by(SEED, selection_name, problem_name) %>%
summarise(
  fit_phylo_10 = condinformation(discretize(max_performance), discretize(lag(variance_phenotype_e
  fit_phylo_100 = condinformation(discretize(max_performance), discretize(lag(variance_phenotype_
  fit_phylo_1000 = condinformation(discretize(max_performance), discretize(lag(variance_phenotype
    fit_pheno_10 = condinformation(discretize(max_performance), discretize(lag(phenotype_num_taxa
      fit_pheno_100 = condinformation(discretize(max_performance), discretize(lag(phenotype_num_t
      fit_pheno_1000 = condinformation(discretize(max_performance), discretize(lag(phenotype_num_
      )

res <- res %>% pivot_longer(cols=contains("o_10"))
res$offset <- str_extract(res$name, "[:digit:]*$")
res$Type <- case_when(str_detect(res$name, "phylo") ~ "Phylogenetic", TRUE ~ "Phenotypic")

ggplot(
  res %>% filter(str_detect(name, "fit_ph*")),
  aes(
```

```
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~problem_name*selection_name) +
  scale_x_discrete("Lag") +
  scale_y_continuous("Transfer Entropy") +
  scale_color_discrete("") +
  theme(legend.position = "bottom")
```




```
res <- data %>% group_by(SEED, selection_name, problem_name) %>%
summarise(
  phen_phylo_10 = condinformation(discretize(phenotype_num_taxa), discretize(lag(max_pl
  phen_phylo_100 = condinformation(discretize(phenotype_num_taxa), discretize(lag(max_p
  pheno_phylo_1000 = condinformation(discretize(phenotype_num_taxa), discretize(lag(ma
    phylo_pheno_10 = condinformation(discretize(max_phenotype_pairwise_distance), disc
      phylo_pheno_100 = condinformation(discretize(max_phenotype_pairwise_distance), di
      phylo_pheno_1000 = condinformation(discretize(max_phenotype_pairwise_distance), c
      )

res <- res %>% pivot_longer(cols=contains("phylo"))
res$offset <- str_extract(res$name, "[:digit:]*$")
res$Type <- case_when(str_detect(res$name, "phylo_pheno") ~ "Phenotypic -> Phylogenetic
```

```
ggplot(
  res,
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~problem_name*selection_name) +
  scale_x_discrete("Lag") +
  scale_y_continuous("Transfer Entropy") +
  scale_color_discrete("") +
  theme(legend.position = "bottom")
```




```
res <- data %>% group_by(SEED, selection_name, problem_name) %>%
summarise(
  phen_phylo_10 = condinformation(discretize(phenotype_num_taxa), discretize(lag(max_phenotype_ev
  phen_phylo_100 = condinformation(discretize(phenotype_num_taxa), discretize(lag(max_phenotype_e
  pheno_phylo_1000 = condinformation(discretize(phenotype_num_taxa), discretize(lag(max_phenotype
    phylo_pheno_10 = condinformation(discretize(max_phenotype_evolutionary_distinctiveness), disc
      phylo_pheno_100 = condinformation(discretize(max_phenotype_evolutionary_distinctiveness), d
      phylo_pheno_1000 = condinformation(discretize(max_phenotype_evolutionary_distinctiveness),
      )
```

```
res <- res %>% pivot_longer(cols=contains("phylo"))
res$offset <- str_extract(res$name, "[:digit:]*$")
res$Type <- case_when(str_detect(res$name, "phylo_pheno") ~ "Phenotypic -> Phylogeneti

ggplot(
  res,
  aes(
    x=as.factor(offset),
    y=value,
    color=Type
    )
  ) +
  geom_boxplot() +
  facet_wrap(~problem_name*selection_name) +
  scale_x_discrete("Lag") +
  scale_y_continuous("Transfer Entropy") +
  scale_color_discrete("") +
  theme(legend.position = "bottom")
```