FS Analysis

This script conducts the statistical analyses and creates the visualizations presented in PAPER TITLE.

TO DO: - adjust font sizes for all figures to 9-10pt for axis titles and 7-8pt for axis tick labels - Modify Size for all figures & make adjustments to fix layout scaling issues after resize

Outline:

1 - Prepare the workspace (scrub the environment, load required packages, set working directories, create helper functions, specify plotting control values).

```
# Environment & Notebook Setup ----------------------------------------------------
rm(list = ls())
options(scipen = 999)
knitr::opts_chunk$set(tidy = TRUE, collapse = TRUE)

# Set Working Directory Paths -----------------------------------------------------
DIR_counts_in <- "~/Box Sync/CB_VF_Shared/Wet_Lab/Projects/Fluctuating_Selection_Project/FS_Code_Supple
DIR_evo_formatted <- "~/Box Sync/CB_VF_Shared/Wet_Lab/Projects/Fluctuating_Selection_Project/FS_Code_Su
DIR_fit_formatted <- "~/Box Sync/CB_VF_Shared/Wet_Lab/Projects/Fluctuating_Selection_Project/FS_Code_Su
DIR_out <- "~/Box Sync/CB_VF_Shared/Wet_Lab/Projects/Fluctuating_Selection_Project/FS_Code_Supplement/F

# Load Packages -------------------------------------------------------------------
require(pwr)
```

```
## Loading required package: pwr
```

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
require(cowplot)
```

```
## Loading required package: cowplot
```

```
require(lme4)
```

```
## Loading required package: lme4
```

```
## Loading required package: Matrix
```

```
require(lmerTest)
```

```
## Loading required package: lmerTest
```

```
##
## Attaching package: 'lmerTest'
```

```
## The following object is masked from 'package:lme4':
##
##     lmer
```

```
## The following object is masked from 'package:stats':
##
##     step
```

```
require(ggthemes)
```

```
## Loading required package: ggthemes
```

```
##
## Attaching package: 'ggthemes'
```

```
## The following object is masked from 'package:cowplot':
##
##     theme_map
```

```
require(sjPlot)
```

```
## Loading required package: sjPlot
```

```
##
## Attaching package: 'sjPlot'
```

```
## The following objects are masked from 'package:cowplot':
##
##     plot_grid, save_plot
```

```
# Specify Helper Functions ----------------------------------------------------
# get_legend ----------------------------
#   Source: https://stackoverflow.com/questions/12041042/how-to-plot-just-the-legends-in-ggplot2
#   Description: scrapes figure legend for later plotting in multipannel plot
get_legend<-function(myggplot){
  tmp <- ggplot_gtable(ggplot_build(myggplot))
  leg <- which(sapply(tmp$grobs, function(x) x$name) == "guide-box")
  legend <- tmp$grobs[[leg]]
  return(legend)
}


# http://www.sthda.com/english/wiki/ggplot2-quick-correlation-matrix-heatmap-r-software-and-data-visual
# Get lower triangle of the correlation matrix
  get_lower_tri<-function(cormat){
    cormat[upper.tri(cormat)] <- NA
    return(cormat)
  }


# %ni% ----------------------------------
#   Description: "not in", opposite of %in% operator.
'%ni%' <- Negate('%in%')



# Specify Plotting Control Values ---------------------------------------------
dichrompalette_br_cy <- RColorBrewer::brewer.pal(n = 11, name = "BrBG")[c(11,10,9,8,3,2,1)]
mypalette <- dichrompalette_br_cy; rm(dichrompalette_br_cy)
```

2 - Library Counts (total counts, total counts less reference, average counts per barcode per fit assay).

```
setwd(DIR_counts_in)
load("metadatapluscounts.rdata")
temp <- myd[myd$sample.type == "MPA_Plate", 125:237]  # retain fitness assay entries only.
temp <- temp[, which(colnames(temp) %ni% c("d2C5", "d2C9"))]  # omit barcode d2C5 and d2C9 columns, the
```

```r
lowcut <- 20
temp[temp <= lowcut] <- NA  # Data supported by <= 20 counts are unreliable -> NA.
print(paste0(sum(temp, na.rm = T), " total counts (reference included)"))
## [1] "96807316 total counts (reference included)"
temp <- temp[, which(colnames(temp) %ni% c("d1C2"))]  # omit reference barcode column, reference makes
print(paste0(sum(temp, na.rm = T), " total counts (reference omitted)"))
## [1] "52853350 total counts (reference omitted)"

# hist of counts frequency
pA <- ggplot(data = data.frame(unlist(temp))) + geom_histogram(aes(x = unlist(temp)),
    colour = "black", fill = "gray90", bins = 50) + scale_x_continuous(name = "Counts",
    expand = expansion(mult = c(0, 0.02))) + scale_y_continuous(name = "Frequency",
    expand = expansion(mult = c(0, 0.02))) + theme_classic() + theme(plot.tag = element_text(face = "bo
    labs(tag = "A")
pA <- ggplotGrob(pA)
## Warning: Removed 3491 rows containing non-finite values (stat_bin).

# hist of log counts frequency (w/ density plot overlay)
pB <- ggplot(data = data.frame(log(unlist(temp))), aes(x = log(unlist(temp)))) +
    geom_histogram(aes(y = ..density..), colour = "black", fill = "gray90",
        bins = 50) + geom_density(alpha = 0.2, fill = NA) + scale_x_continuous(name = "Log Counts",
    limits = c(0, 15), expand = expansion(mult = c(0, 0.02))) + scale_y_continuous(name = "Frequency",
    expand = expansion(mult = c(0, 0.02))) + theme_classic() + theme(plot.tag = element_text(face = "bo
    labs(tag = "B")
pB <- ggplotGrob(pB)
## Warning: Removed 3491 rows containing non-finite values (stat_bin).
## Warning: Removed 3491 rows containing non-finite values (stat_density).
## Warning: Removed 2 rows containing missing values (geom_bar).

temp2 <- colMeans(temp, na.rm = T)
print("Summary: counts per barcode across 110 fitness assay sample pools")
## [1] "Summary: counts per barcode across 110 fitness assay sample pools"
summary(temp2)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   360.8  1966.2  3442.2  5655.6  5862.8 59582.9
print(paste0("Counts per barcode is extremely variable across barcodes due to treatment effects in fitn
    sd(temp2)))
## [1] "Counts per barcode is extremely variable across barcodes due to treatment effects in fitness as

# hist of barcode mean counts frequency
pC <- ggplot(data = data.frame(unlist(temp2))) + geom_histogram(aes(x = unlist(temp2)),
    colour = "black", fill = "gray90", bins = 50) + scale_x_continuous(name = "Counts",
    expand = expansion(mult = c(0, 0.02))) + scale_y_continuous(name = "Frequency",
    expand = expansion(mult = c(0, 0.02))) + theme_classic() + theme(plot.tag = element_text(face = "bo
    labs(tag = "C")
pC <- ggplotGrob(pC)


# hist of barcode mean log counts frequency (w/ density plot overlay)
pD <- ggplot(data = data.frame(log(unlist(temp2))), aes(x = log(unlist(temp2)))) +
    geom_histogram(aes(y = ..density..), colour = "black", fill = "gray90",
        bins = 50) + geom_density(alpha = 0.2, fill = NA) + scale_x_continuous(name = "Log Counts",
    limits = c(0, 15), expand = expansion(mult = c(0, 0.02))) + scale_y_continuous(name = "Frequency",
```

```
    expand = expansion(mult = c(0, 0.02))) + theme_classic() + theme(plot.tag = element_text(face = "bo
    labs(tag = "D")
pD <- ggplotGrob(pD)
## Warning: Removed 2 rows containing missing values (geom_bar).

# output viz
setwd(DIR_out)
pdf("SSS_CountsHists.pdf", height = 8.25, width = 8.25)
cowplot::plot_grid(pA, pB, pC, pD, nrow = 2, align = "v")
dev.off()
## pdf
##   2
rm(myd, temp, temp2, lowcut, pA, pB, pC, pD)
```

3 - Barcode Cross-Contamination (Reference purity, Single-well controls).

```
# Reference purity.
# ------------------------------------------------------------
setwd(DIR_fit_formatted)
load("swref.rdata")
temp <- swref[, c("ppects", "ppccts", "ppcctsm", "cc", "ccm")]
rm(swref)

tem <- as.data.frame(cbind(mean(temp$ppects), sd(temp$ppects)/sqrt(length(temp$ppects)),
    mean(temp$ppccts), sd(temp$ppccts)/sqrt(length(temp$ppccts)), mean(temp$ppcctsm),
    sd(temp$ppcctsm)/sqrt(length(temp$ppcctsm)), mean(temp$cc), (sd(temp$cc)/sqrt(length(temp$cc))),
    mean(temp$ccm), (sd(temp$ccm)/sqrt(length(temp$ccm))), length(temp$ppects)))
colnames(tem) <- c("ec_mean", "ec_se", "uct_mean", "uct_se", "ucm_mean", "ucm_se",
    "ccpt_mean", "ccpt_se", "ccpm_mean", "ccpm_se", "n_reps")
tem$day <- NA
tem$plateloc <- NA
tem$group <- "ref"


# single well controls
# ------------------------------------------------------------
setwd(DIR_evo_formatted)
load("swref.rdata")
temp <- swref[, c("ppects", "ppccts", "ppcctsm", "cc", "ccm", "replicate")]
rm(swref)
temp <- temp[c(1:(nrow(temp) - 3)), ]  # remove sulf samples (not used in this manuscript)

temp1s <- temp[c((nrow(temp) - 2):nrow(temp)), ]
temp <- temp[c(1:(nrow(temp) - 3)), ]
temp1s <- as.data.frame(cbind(temp1s$ppects, NA, temp1s$ppccts, NA, temp1s$ppcctsm,
    NA, temp1s$cc, NA, temp1s$ccm, NA, 1, 50, c("A1", "E6", "F8"), "COPR"))
colnames(temp1s) <- colnames(tem)
tem <- rbind(tem, temp1s)
rm(temp1s)

i <- 1
days <- c(rep(0, times = 9), rep(50, times = 9))
platelocs <- rep(c("A1", "A1", "A1", "E6", "E6", "E6", "F8", "F8", "F8"), times = 2)
groups <- c(rep("ancestral", times = 9), rep("SALT", times = 9))
```

```
while (i <= (nrow(temp) - 2)) {
    te <- temp[i:(i + 2), ]
    te <- as.data.frame(cbind(mean(te$ppects), sd(te$ppects)/sqrt(length(te$ppects)),
        mean(te$ppccts), sd(te$ppccts)/sqrt(length(te$ppccts)), mean(te$ppcctsm),
        sd(te$ppcctsm)/sqrt(length(te$ppcctsm)), mean(te$cc), (sd(te$cc)/sqrt(length(te$cc))),
        mean(te$ccm), (sd(te$ccm)/sqrt(length(te$ccm))), length(te$ppects)))
    colnames(te) <- c("ec_mean", "ec_se", "uct_mean", "uct_se", "ucm_mean",
        "ucm_se", "ccpt_mean", "ccpt_se", "ccpm_mean", "ccpm_se", "n_reps")
    te$day <- days[i]
    te$plateloc <- platelocs[i]
    te$group <- groups[i]
    tem <- rbind(tem, te)
    rm(te)
    i <- i + 3
}
mycc <- tem
rm(tem, days, platelocs, groups, temp, i)
for (i in 1:(ncol(mycc) - 2)) {
    mycc[, i] <- as.numeric(mycc[, i])
}
rm(i)
setwd(DIR_out)
write.csv(mycc, file = "SSS_CC.csv")

# reference BC [%]
mycc[1, ]$ccpt_mean
## [1] 0.0005962786
mycc[1, ]$ccpt_se
## [1] 0.000120813

# ancestral [%]
mean(mycc[mycc$group == "ancestral", ]$ccpt_mean)
## [1] 0.0001115948
sd(mycc[mycc$group == "ancestral", ]$ccpt_mean)/sqrt(length(mycc[mycc$group ==
    "ancestral", ]$ccpt_mean))
## [1] 0.00004195045

# Day 50 SALT [%]
mean(mycc[mycc$group == "SALT", ]$ccpt_mean)
## [1] 0.0003028522
sd(mycc[mycc$group == "SALT", ]$ccpt_mean)/sqrt(length(mycc[mycc$group == "SALT",
    ]$ccpt_mean))
## [1] 0.0000702448

# Day 50 COPR [%]
mean(mycc[mycc$group == "COPR", ]$ccpt_mean)
## [1] 0.007649521
sd(mycc[mycc$group == "COPR", ]$ccpt_mean)/sqrt(length(mycc[mycc$group == "COPR",
    ]$ccpt_mean))
## [1] 0.005716901

# Day 50 COMBINED [%] Day 50 COPR [%]
mean(mycc[mycc$group %in% c("SALT", "COPR"), ]$ccpt_mean)
```

```
## [1] 0.003976187
sd(mycc[mycc$group %in% c("SALT", "COPR"), ]$ccpt_mean)/sqrt(length(mycc[mycc$group %in%
    c("SALT", "COPR"), ]$ccpt_mean))
## [1] 0.003039121
rm(mycc)
```

3.5 - extinction and extirpation

```
setwd(DIR_fit_formatted)
load("FitAssayData.rdata")  # load fitness assay data, we will use it from here forward.

alive <- function(x) {
    sum(!is.na(x))
}  # throw-away funciton to sum extant barcodes below.
chems <- list(c("SALT_A", "SALT_B"), c("COPR_A", "COPR_B"))  # loop controls
outnames <- c("SALT", "COPR")

# calculate extinct/extirpated and surviving lineages in each treatment X
# assay environment and create formatted tables for output indicating n and
# proportion of treatment total n in each category.
for (i in 1:length(chems)) {
    m <- myrcw[myrcw$epo %in% chems[[i]], ]
    m$ext00 <- is.na(m$dw_00)
    m$ext80 <- is.na(m$dw_80)
    m$ext080 <- rowSums(m[, c("ext00", "ext80")])
    m$ali080 <- apply(m[, c("dw_00", "dw_80")], 1, alive)
    m$ali080[m$ali080 < 2] <- FALSE
    m$ali080[m$ali080 == 2] <- TRUE
    m$ali080 <- as.logical(m$ali080)
    m$ext080[m$ext080 < 2] <- FALSE
    m$ext080[m$ext080 == 2] <- TRUE
    m$ext080 <- as.logical(m$ext080)
    m <- as.matrix(cbind(tapply(m$ali080, m$treat, sum), tapply(m$ext080, m$treat,
        sum), tapply(m$ext00 - m$ext080, m$treat, sum), tapply(m$ext80 - m$ext080,
        m$treat, sum), round(tapply(m$ali080, m$treat, sum)/c(32, 32, 32, 32,
        30, 32, 30), 2), round(tapply(m$ext080, m$treat, sum)/c(32, 32, 32,
        32, 30, 32, 30), 2), round(tapply(m$ext00 - m$ext080, m$treat, sum)/c(32,
        32, 32, 32, 30, 32, 30), 2), round(tapply(m$ext80 - m$ext080, m$treat,
        sum)/c(32, 32, 32, 32, 30, 32, 30), 2)))
    colnames(m) <- c("pres_both", "ext_both", "ext_00_only", "ext_80_only",
        "pres_both_p", "ext_both_p", "ext_00_only_p", "ext_80_only_p")
    setwd(DIR_out)
    write.csv(m, file = paste0("SSS_Extinction", outnames[i], ".csv"))
}
rm(m, chems, i, outnames, alive)
```

4 - Power (indiv bc fitness change, treatment effects).

```
# power to detect fitness increase / decrease (t-tests)
# ------------------------ d = m1 - m2 / q; m1 is mean group 1, m2 is mean
# group 2, q is common standard deviation in the two groups here m1 is
# change in fitness and m2 is 0. mean q is used and is the weighted mean of
# all population standard deviation of the four replicate sets in each row
# of the dataset.
```

```r
# prep frame & calculate psd -----------
temp <- myrc[!is.na(myrc$fit_e_1) & !is.na(myrc$fit_e_2) & !is.na(myrc$fit_e_3) &
    !is.na(myrc$fit_e_4) & !is.na(myrc$fit_a) & !is.na(myrc$dw) & !is.na(myrc$reads_dw),
    ]
temp$dw_1 <- temp$fit_e_1 - temp$fit_a
temp$dw_2 <- temp$fit_e_2 - temp$fit_a
temp$dw_3 <- temp$fit_e_3 - temp$fit_a
temp$dw_4 <- temp$fit_e_4 - temp$fit_a
temp$psd_r1 <- ((temp$dw - temp$dw_1)^2)/4
temp$psd_r2 <- ((temp$dw - temp$dw_2)^2)/4
temp$psd_r3 <- ((temp$dw - temp$dw_3)^2)/4
temp$psd_r4 <- ((temp$dw - temp$dw_4)^2)/4
temp$psd_m <- rowMeans(cbind(temp$psd_r1, temp$psd_r2, temp$psd_r3, temp$psd_r4),
    na.rm = T)
temp$psd <- sqrt(temp$psd_m)
psd <- weighted.mean(temp$psd, temp$reads_dw, na.rm = T)
rm(temp)

# report power -------------------------
print(paste0(pwr.t.test(d = 0.01/psd, n = 4, sig.level = 0.05)$power * 100,
    "% power to detect a 1% fitness change for an average BC"))
## [1] "25.992098103152% power to detect a 1% fitness change for an average BC"
print(paste0("80% power to detect a ", pwr.t.test(power = 0.8, n = 4, sig.level = 0.05)$d *
    psd * 100, "% fitness change for an average BC"))
## [1] "80% power to detect a 2.16292232223626% fitness change for an average BC"

# create data for plotting -------------
powerBC <- matrix(nrow = 100, ncol = 5)
colnames(powerBC) <- c("fitchange", "psd", "n", "power", "sig.level")
powerBC <- as.data.frame(powerBC)
powerBC$fitchange <- seq(0, 0.04, length.out = 100)  # fitness changes to calculate power for.
powerBC$psd <- psd  # psd calculated above
powerBC$n <- 4  # repliacates
powerBC$sig.level <- 0.05  # significance level
for (i in 1:nrow(powerBC)) {
    powerBC$power[i] <- pwr.t.test(d = powerBC$fitchange[i]/powerBC$psd[i],
        n = powerBC$n[i], sig.level = powerBC$sig.level[i])$power
}
rm(i)  # calculate power
setwd(DIR_out)
write.csv(powerBC, file = "SSS_powerBCTable.csv")

# Generate plot panel A ----------------
pA <- ggplot(powerBC, aes(x = fitchange, y = power)) + geom_segment(x = pwr.t.test(power = 0.8,
    n = 4, sig.level = 0.05)$d * psd, xend = pwr.t.test(power = 0.8, n = 4,
    sig.level = 0.05)$d * psd, y = 0, yend = 0.8, lty = "dotdash", color = "gray",
    size = 0.25) + geom_segment(x = 0, xend = pwr.t.test(power = 0.8, n = 4,
    sig.level = 0.05)$d * psd, y = 0.8, yend = 0.8, lty = "dotdash", color = "gray",
    size = 0.25) + geom_point(x = pwr.t.test(power = 0.8, n = 4, sig.level = 0.05)$d *
    psd, y = 0.8, pch = 19, size = 3, color = "darkgray") + geom_text(x = 2.5/100,
    y = 0.75, label = paste0(round(pwr.t.test(power = 0.8, n = 4, sig.level = 0.05)$d *
        psd * 100, 2), ", 80"), color = "darkgray", size = 2) + geom_line() +
    scale_x_continuous(name = "Fitness Change\n (for Individual Strains) \n ",
```

```
        breaks = seq(0, 0.04, by = 0.005), labels = paste0(seq(0, 0.04, by = 0.005) *
            100, "%"), limits = c(0, 0.04), expand = expansion(mult = c(0, 0.02))) +
    scale_y_continuous(name = "Power", limits = c(0, 1), breaks = seq(0, 1,
        by = 0.25), labels = paste0(seq(0, 1, by = 0.25) * 100, "%"), expand = expansion(mult = c(0,
        0.02))) + theme_classic() + labs(tag = "A") + theme(axis.text.x = element_text(angle = 45,
    hjust = 1), plot.tag = element_text(face = "bold"))



# power to detect Treatment effects (lmer model)
# ------------------------------ f2 (effect size) = 1/rmse (where rmse =
# root mean square error among replicate measures of dw).  obtain rmse via
# rmse = sqrt(mean(residuals(MODEL)^2))*100. Use pwr.f2.test in pwr package
# to obtain power to detect treatment effects between treatments with a
# TOTAL number of barcodes equal to v between them.

# prep frame & calc effect size (f2) ---
temp <- my[my$mpasp %in% c(0, 0.8), ]  # Restrict to SALT & COPR dw data in 0.0 and 0.8 environments
temp$ID <- paste0(temp$bcpID, temp$mpasp)
myrmse <- sqrt(mean(residuals(lm(temp$dw ~ temp$ID + 0, weights = I(log(temp$re_dw))))^2)) *
    100
myrmse  # this is the rmse from the model*100 --> effect size of 1 corresonds to an 2.41% (rmse) fitnes
## [1] 2.418516
myeffectsize <- 1/myrmse
myeffectsize
## [1] 0.4134767
rm(temp)  # effect size of 1 is an 2.41% (rmse) fitness cahnge, so for a fitness change of 1(%), expect

# report power -------------------------
print(paste0(pwr.f2.test(u = 1, v = ((28 * 2) - 1 - 1), f2 = myeffectsize, sig.level = 0.05)$power *
    100, "% power to detect a 1% fitness change between treatments (assuming no ext lineages)"))
## [1] "99.7152356944258% power to detect a 1% fitness change between treatments (assuming no ext linea
print(paste0("80% power to detect a ", pwr.f2.test(u = 1, v = ((28 * 2) - 1 -
    1), power = 0.8, sig.level = 0.05)$f2 * myrmse, "% fitness change between treatments (assuming no e
## [1] "80% power to detect a 0.351456784470059% fitness change between treatments (assuming no ext lin
print(paste0(pwr.f2.test(u = 1, v = ((10 * 2) - 1 - 1), f2 = myeffectsize, sig.level = 0.05)$power *
    100, "% power to detect a 1% fitness difference between treatments (worst case scenario)"))
## [1] "77.6249436580094% power to detect a 1% fitness difference between treatments (worst case scenari
print(paste0("80% power to detect a ", pwr.f2.test(u = 1, v = ((10 * 2) - 1 -
    1), power = 0.8, sig.level = 0.05)$f2 * myrmse, "% fitness change for an average BC (assuming 50% e
## [1] "80% power to detect a 1.06138509814466% fitness change for an average BC (assuming 50% ext linea

# create data for plotting -------------
powerTR <- matrix(nrow = 200, ncol = 5)
colnames(powerTR) <- c("u", "v", "power", "f2", "sig.level")
powerTR <- as.data.frame(powerTR)
powerTR$v <- c(rep((28 * 2) - 1 - 1, times = 100), rep((10 * 2) - 1 - 1, times = 100))
powerTR$u <- 1
powerTR$n <- powerTR$v + 2
powerTR$n <- factor(powerTR$n, levels = c(56, 20))
powerTR$f2 <- seq(0, myeffectsize * 1.5, length.out = 100)
powerTR$sig.level <- 0.05
for (i in 1:nrow(powerTR)) {
```

```
    powerTR$power[i] <- pwr.f2.test(u = powerTR$u[i], v = powerTR$v[i], f2 = powerTR$f2[i],
        sig.level = powerTR$sig.level[i])$power
}
rm(i)
powerTR$f2 <- powerTR$f2 * myrmse
setwd(DIR_out)
write.csv(powerTR, file = "SSS_powerTRTable.csv")

# Generate plot panel B ----------------
pB <- ggplot(powerTR, aes(x = f2, y = power, group = n)) + geom_segment(x = pwr.f2.test(u = 1,
    v = ((28 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, xend = pwr.f2.test(u = 1,
    v = ((28 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, y = 0,
    yend = 0.8, lty = "dotdash", color = "gray", size = 0.25) + geom_segment(x = pwr.f2.test(u = 1,
    v = ((10 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, xend = pwr.f2.test(u = 1,
    v = ((10 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, y = 0,
    yend = 0.8, lty = "dotdash", color = "gray", size = 0.25) + geom_segment(x = 0,
    xend = pwr.f2.test(u = 1, v = ((10 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 *
        myrmse, y = 0.8, yend = 0.8, lty = "dotdash", color = "gray", size = 0.25) +
    geom_point(x = pwr.f2.test(u = 1, v = ((28 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 *
        myrmse, y = 0.8, pch = 19, size = 3, color = "darkgray") + geom_point(x = pwr.f2.test(u = 1,
    v = ((10 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, y = 0.8,
    pch = 19, size = 3, color = "darkgray") + geom_text(x = 0.2, y = 0.85, label = paste0(round(pwr.f2.
    v = ((28 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, 2),
    ", 80"), color = "darkgray", size = 2) + geom_text(x = 1.22, y = 0.75, label = paste0(round(pwr.f2.
    v = ((10 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, 2),
    ", 80    "), color = "darkgray", size = 2) + geom_line(aes(lty = n)) + scale_x_continuous(name = "F
    breaks = seq(0, 1.5, by = 0.25), labels = paste0(seq(0, 1.5, by = 0.25),
        "%"), limits = c(0, 1.5), expand = expansion(mult = c(0, 0.02))) + scale_y_continuous(name = "P
    limits = c(0, 1), breaks = seq(0, 1, by = 0.25), labels = paste0(seq(0,
        1, by = 0.25) * 100, "%"), expand = expansion(mult = c(0, 0.02))) +
    theme_classic() + labs(tag = "B") + theme(legend.position = c(0.9, 0.3)) +
    theme(axis.text.x = element_text(angle = 45, hjust = 1), plot.tag = element_text(face = "bold"))


# plot panels A&B in a single figure ---
setwd(DIR_out)
pdf(file = "SSS_Power.pdf", width = 3.345, height = 6.69)
cowplot::plot_grid(pA, pB, nrow = 2)
dev.off()  # one column wide, equal height
## pdf
##   2
rm(pA, pB, powerBC, powerTR, myeffectsize, myrmse, psd)
```

5 - Treatment effects on fitness (statistics)

```
# setup
treatlist <- c("EH0", "EH0_40", "EH40", "EH20_60", "EH0_80", "EH40_80", "EH80")
treattreatlist <- paste0("treat", treatlist)
myres <- matrix(nrow = 7, ncol = 7)
rownames(myres) <- treattreatlist
colnames(myres) <- rownames(myres)
chems <- list(c("SALT_A", "SALT_B"), c("COPR_A", "COPR_B"))
envs <- c(0, 0.8)
setwd(DIR_out)  # outputting with this block, so adjust dir.
```

```
# run models for each chemical x assay environment testing treatments
# against one another
for (g in 1:length(chems)) {
    for (h in 1:length(envs)) {
        myres_p <- myres
        myres_est <- myres
        for (i in 1:length(treatlist)) {
            m <- my[my$mpasp == envs[h] & my$epo %in% chems[[g]] & !is.na(my$dw) &
                !is.na(my$re_dw), ]
            m$treat <- relevel(m$treat, ref = treatlist[i])
            mod <- lmer(dw ~ treat + (1 | bcpID), weights = log(re_dw), data = m)
            print(tab_model(mod, digits = 5, digits.p = 8, show.stat = T, wrap.labels = 100,
                file = paste0("SSS_lmer_TreatEffects_", strsplit(chems[[g]][1],
                  "_")[[1]][1], "_", envs[h], ".html")))
            for (j in 1:length(treattreatlist)) {
                if (treatlist[j] != treatlist[i]) {
                  myres_est[treattreatlist[i], treattreatlist[j]] <- summary(mod)$coefficients[treattrea
                    "Estimate"]
                  myres_p[treattreatlist[i], treattreatlist[j]] <- summary(mod)$coefficients[treattreatl
                    "Pr(>|t|)"]
                }
            }
        }
        colnames(myres_est) <- c("EH0", "EH0_40", "EH40", "EH20_60", "EH0_80",
            "EH40_80", "EH80")
        rownames(myres_est) <- colnames(myres_est)
        assign(paste0("myres_est_", strsplit(chems[[g]][1], "_")[[1]][1], "_",
            envs[h]), myres_est)
        colnames(myres_p) <- c("EH0", "EH0_40", "EH40", "EH20_60", "EH0_80",
            "EH40_80", "EH80")
        rownames(myres_p) <- colnames(myres_est)
        assign(paste0("myres_p_", strsplit(chems[[g]][1], "_")[[1]][1], "_",
            envs[h]), myres_p)
    }
}
rm(m, mod, g, h, i, j, treatlist, treattreatlist, myres_est, myres_p, weights)


# run models for each chemical x assay environment testing treatments
# against 0.
myres0_est <- myres[, 1:4]
colnames(myres0_est) <- c("SALT_0.0", "SALT_0.8", "COPR_0.0", "COPR_0.8")
myres0_p <- myres0_est
counter <- 1
for (g in 1:length(chems)) {
    for (h in 1:length(envs)) {
        m <- my[my$mpasp == envs[h] & my$epo %in% chems[[g]] & !is.na(my$dw),
            ]
        mod <- lmer(dw ~ treat + 0 + (1 | bcpID), weights = log(re_dw), data = m)
        myres0_est[, counter] <- summary(mod)$coefficients[, 1]
        myres0_p[, counter] <- summary(mod)$coefficients[, 5]
        print(tab_model(mod, digits = 5, digits.p = 8, show.stat = T, wrap.labels = 100,
            file = paste0("SSS_lmer_TreatEffects_VS0_", strsplit(chems[[g]][1],
```

```
                "_")[[1]][1], "_", envs[h], ".html")))
        counter <- counter + 1
    }
}
rm(chems, m, mod, g, h, envs, myres, counter, weights)
```

```
# get datasets -------------------------
mAA <- myrc[myrc$mpasp == 0 & myrc$epo %in% c("SALT_A", "SALT_B") & !is.na(myrc$dw) &
    !is.na(myrc$reads_dw), ]
mCC <- myrc[myrc$mpasp == 0.8 & myrc$epo %in% c("SALT_A", "SALT_B") & !is.na(myrc$dw) &
    !is.na(myrc$reads_dw), ]
mBB <- myrc[myrc$mpasp == 0 & myrc$epo %in% c("COPR_A", "COPR_B") & !is.na(myrc$dw) &
    !is.na(myrc$reads_dw), ]
mDD <- myrc[myrc$mpasp == 0.8 & myrc$epo %in% c("COPR_A", "COPR_B") & !is.na(myrc$dw) &
    !is.na(myrc$reads_dw), ]


# control block ------------------------
fitaxdatlimSALT <- c(-0.15 * 100, 0.45 * 100)  #; min(c(mAA$dw, mCC$dw)); max(c(mAA$dw, mCC$dw))
fitaxdatlimCOPR <- c(-0.1 * 100, 0.55 * 100)  #; min(c(mBB$dw, mDD$dw)); max(c(mBB$dw, mDD$dw))

fitaxlablim <- c(-0.1 * 100, 0.5 * 100)
boxSALT <- c(0.2 * 100, 0.49 * 100)  #; (boxSALT[2]-boxSALT[1]) / (fitaxdatlimSALT[2] - fitaxdatlimSALT
boxCOPR <- c(0.275 * 100, 0.59 * 100)  #; (boxCOPR[2]-boxCOPR[1]) / (fitaxdatlimCOPR[2] - fitaxdatlimCO

treatlabSALT <- "NaCl Evol. Treatment"
treatlabCOPR <- expression(bold(CuSO["4"] * "  Evol. Treatment"))
fitnesslabCM <- "Fitness Change in CM (%)"
fitnesslabSALT <- "Fitness Change in CM + NaCl (%)"
fitnesslabCOPR <- expression(bold("Fitness Change in CM + " * CuSO["4"] * " (%)"))


# inset backer -------------------------
myres_backer <- myres_est_SALT_0
myres_backer[myres_backer < 0] <- -1
myres_backer[myres_backer > 0 | is.na(myres_backer)] <- 1
myres_backer <- round(myres_backer, 2)
myres_backer <- get_lower_tri(myres_backer)
melt_myres_backer <- reshape::melt(myres_backer)
melt_myres_backer$X1 <- factor(melt_myres_backer$X1, levels = c("EH0", "EH0_40",
    "EH40", "EH20_60", "EH0_80", "EH40_80", "EH80"))
melt_myres_backer$X2 <- factor(melt_myres_backer$X2, levels = c("EH0", "EH0_40",
    "EH40", "EH20_60", "EH0_80", "EH40_80", "EH80"))


# prep p and est values ----------------
lsuffs <- c("AA", "CC", "BB", "DD")
lintabs <- c("SALT_0", "SALT_0.8", "COPR_0", "COPR_0.8")
for (i in 1:4) {
    assign("myres_est_temp", get(paste0("myres_est_", lintabs[i])))
    assign("myres_p_temp", get(paste0("myres_p_", lintabs[i])))
    myres_est_temp[which(myres_p_temp > 0.05)] <- NA
    myres_est_temp <- round(myres_est_temp * 100, 0)
    myres_est_temp <- get_lower_tri(myres_est_temp)  # uncomment for just lower triangle
```

```
    melt_myres_est_temp <- reshape::melt(myres_est_temp)
    melt_myres_est_temp$X1 <- factor(melt_myres_est_temp$X1, levels = c("EH0",
        "EH0_40", "EH40", "EH20_60", "EH0_80", "EH40_80", "EH80"))
    melt_myres_est_temp$X2 <- factor(melt_myres_est_temp$X2, levels = c("EH0",
        "EH0_40", "EH40", "EH20_60", "EH0_80", "EH40_80", "EH80"))
    assign(paste0("melt_myres_est_", lsuffs[i]), melt_myres_est_temp)
}
rm(melt_myres_est_temp, myres_est_COPR_0, myres_est_COPR_0.8, myres_est_SALT_0,
    myres_est_SALT_0.8, myres_est_temp, myres_p_COPR_0, myres_p_COPR_0.8, myres_p_SALT_0,
    myres_p_SALT_0.8, myres_p_temp, i, lintabs, lsuffs)


viopanel <- function(dwdata, statsdata, backerdata, fitlims, insetlims, fitaxlablim,
    treatlab, fitlab, tag, usepalette, dcol = "dw", fitaxstep = 0.1 * 100) {
    pZZ <- ggplot(dwdata, aes(x = treat, y = dwdata[, dcol] * 100, group = treat,
        color = treat)) + geom_hline(yintercept = 0, lty = "dotted") + geom_violin(fill = "gray90",
        color = "gray30", draw_quantiles = NULL, trim = T) + geom_jitter(width = 0.15,
        height = 0, alpha = 0.35) + geom_rug(sides = "l", alpha = 0.7, length = unit(0.015,
        "npc")) + geom_point(stat = "summary", fun = "mean", color = "black",
        cex = 1.5) + geom_point(stat = "summary", fun = "median", color = "black",
        cex = 3, pch = 1) + geom_errorbar(stat = "summary", fun.data = "mean_se",
        color = "black", width = 0, lwd = 0.5) + scale_y_continuous(breaks = seq(fitaxlablim[1],
        fitaxlablim[2], by = fitaxstep), limits = c(fitlims[1], fitlims[2]),
        labels = scales::number_format(accuracy = 0.01 * 100)) + scale_color_manual(values = usepalette
        coord_flip() + ylab(fitlab) + xlab(treatlab) + labs(tag = tag) + theme_classic() +
        theme(axis.title = element_text(size = 12, face = "bold"), axis.title.y = element_text(margin =
            5, 0, 0, "pt")), axis.title.x = element_text(margin = margin(5,
            0, 0, 0, "pt")), axis.text = element_text(size = 10, face = "bold"),
            plot.tag = element_text(face = "bold", size = 20), legend.position = "none")
    ppZZ <- ggplot(backerdata, aes(x = X1, y = X2, fill = value)) + geom_tile(show.legend = F) +
        scale_y_discrete(position = "right") + theme_tufte(base_family = "Helvetica") +
        theme(axis.ticks = element_blank(), axis.title = element_blank(), axis.text.x = element_text(ang
            hjust = 1, vjust = 0.5, margin = margin(t = -2.5, b = -2.5), size = 6),
            axis.text.y.right = element_text(angle = 0, hjust = 0, vjust = 0.5,
                margin = margin(l = -2.5, r = -2.5), size = 6)) + scale_fill_gradient2(low = "transparer
        high = "gray80", mid = "gray80", midpoint = 0, limit = c(0, 1), space = "Lab",
        name = "Estimate", na.value = "transparent")
    pppZZ <- ggplot(statsdata, aes(x = X1, y = X2, fill = value)) + geom_tile(show.legend = F) +
        geom_text(aes(X1, X2, label = value), color = "black", size = 2, angle = 0,
            fontface = "bold") + scale_y_discrete(position = "right") + theme_tufte(base_family = "Helv
        theme(axis.ticks = element_blank(), axis.title = element_blank(), axis.text.x = element_text(ang
            hjust = 1, vjust = 0.5, margin = margin(t = -2.5, b = -2.5), size = 6,
            face = "bold"), axis.text.y.right = element_text(angle = 0, hjust = 0,
            vjust = 0.5, margin = margin(l = -2.5, r = -2.5), size = 6, face = "bold")) +
        scale_fill_gradient2(low = "blue", high = "red", mid = "white", midpoint = 0,
            space = "Lab", name = "Estimate", na.value = "transparent", limit = c(min(statsdata$value),
                max(statsdata$value)))
    pZZ <- pZZ + annotation_custom(ggplotGrob(ppZZ), xmin = 0.5, xmax = 3.5,
        ymin = insetlims[1], ymax = insetlims[2])
    rm(ppZZ)
    pZZ <- pZZ + annotation_custom(ggplotGrob(pppZZ), xmin = 0.5, xmax = 3.5,
        ymin = insetlims[1], ymax = insetlims[2])
    rm(pppZZ)
```

```
    return(pZZ)
}

pAA <- viopanel(dwdata = mAA, statsdata = melt_myres_est_AA, backerdata = melt_myres_backer,
    fitlims = fitaxdatlimSALT, insetlims = boxSALT, fitaxlablim = fitaxlablim,
    treatlab = treatlabSALT, fitlab = fitnesslabCM, tag = "A", usepalette = mypalette)
## Warning: Removed 36 rows containing missing values (geom_text).
pCC <- viopanel(dwdata = mCC, statsdata = melt_myres_est_CC, backerdata = melt_myres_backer,
    fitlims = fitaxdatlimSALT, insetlims = boxSALT, fitaxlablim = fitaxlablim,
    treatlab = treatlabSALT, fitlab = fitnesslabSALT, tag = "C", usepalette = mypalette)
## Warning: Removed 31 rows containing missing values (geom_text).
pBB <- viopanel(dwdata = mBB, statsdata = melt_myres_est_BB, backerdata = melt_myres_backer,
    fitlims = fitaxdatlimCOPR, insetlims = boxCOPR, fitaxlablim = fitaxlablim,
    treatlab = treatlabCOPR, fitlab = fitnesslabCM, tag = "B", usepalette = mypalette)
## Warning: Removed 40 rows containing missing values (geom_text).
pDD <- viopanel(dwdata = mDD, statsdata = melt_myres_est_DD, backerdata = melt_myres_backer,
    fitlims = fitaxdatlimCOPR, insetlims = boxCOPR, fitaxlablim = fitaxlablim,
    treatlab = treatlabCOPR, fitlab = fitnesslabCOPR, tag = "D", usepalette = mypalette)
## Warning: Removed 34 rows containing missing values (geom_text).

setwd(DIR_out)
pdf(file = "SSS_TreatEffectViolins.pdf", width = 11, height = 8.5)
cowplot::plot_grid(pAA, pBB, pCC, pDD, nrow = 2)
dev.off()  # one column wide, equal height
## pdf
##   2

rm(mAA, mBB, mCC, mDD, melt_myres_est_AA, melt_myres_est_BB, melt_myres_est_CC,
    melt_myres_est_DD, myres_backer, myres0_est, myres0_p, pAA, pBB, pCC, pDD,
    boxCOPR, boxSALT, fitaxdatlimCOPR, fitaxdatlimSALT, fitaxlablim, fitnesslabCM,
    fitnesslabCOPR, fitnesslabSALT, treatlabCOPR, treatlabSALT)
```

Geomean fit calcs

```
# setup
treatlist <- c("EH0", "EH0_40", "EH40", "EH20_60", "EH0_80", "EH40_80", "EH80")
treattreatlist <- paste0("treat", treatlist)
myres <- matrix(nrow = 7, ncol = 7)
rownames(myres) <- treattreatlist
colnames(myres) <- rownames(myres)
chems <- list(c("SALT_A", "SALT_B"), c("COPR_A", "COPR_B"))
setwd(DIR_out)  # outputting with this block, so adjust dir.


# run models for each chemical x assay environment testing treatments
# against one another
for (g in 1:length(chems)) {
    myres_p_mn <- myres
    myres_est_mn <- myres
    myres_p_spread <- myres
    myres_est_spread <- myres
    for (i in 1:length(treatlist)) {
        m <- myrcw[myrcw$epo %in% chems[[g]] & !is.na(myrcw$dw_00_80) & !is.na(myrcw$reads_dw_00_80),
            ]
```

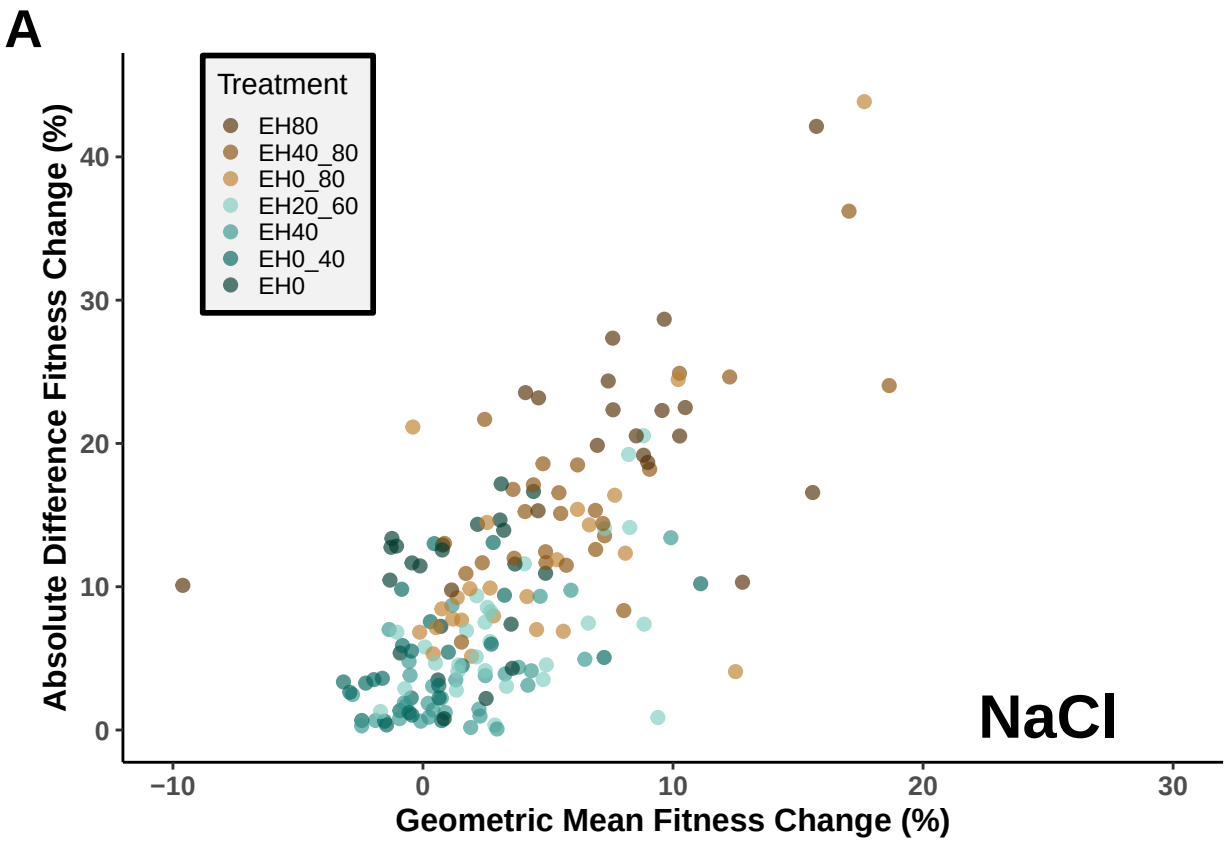
```
        m$spread_dw_00_80 <- m$absdif_dw_00_80
        m$treat <- relevel(m$treat, ref = treatlist[i])
        mod <- lm(dw_00_80 ~ treat, weights = log(reads_dw_00_80), data = m)  # mean
        mod2 <- lm(spread_dw_00_80 ~ treat, weights = log(reads_dw_00_80), data = m)  # spread
        print(tab_model(mod, digits = 5, digits.p = 8, show.stat = T, wrap.labels = 100,
            file = paste0("SSS_lm_TreatEffects_geomean", strsplit(chems[[g]][1],
                "_")[[1]][1], ".html")))
        print(tab_model(mod2, digits = 5, digits.p = 8, show.stat = T, wrap.labels = 100,
            file = paste0("SSS_lm_TreatEffects_spread", strsplit(chems[[g]][1],
                "_")[[1]][1], ".html")))
        for (j in 1:length(treattreatlist)) {
            if (treatlist[j] != treatlist[i]) {
                myres_est_mn[treattreatlist[i], treattreatlist[j]] <- summary(mod)$coefficients[treattr
                  "Estimate"]
                myres_p_mn[treattreatlist[i], treattreatlist[j]] <- summary(mod)$coefficients[treattrea
                  "Pr(>|t|)"]

                myres_est_spread[treattreatlist[i], treattreatlist[j]] <- summary(mod2)$coefficients[tr
                  "Estimate"]
                myres_p_spread[treattreatlist[i], treattreatlist[j]] <- summary(mod2)$coefficients[trea
                  "Pr(>|t|)"]
            }
        }
    }
    colnames(myres_est_mn) <- c("EH0", "EH0_40", "EH40", "EH20_60", "EH0_80",
        "EH40_80", "EH80")
    rownames(myres_est_mn) <- colnames(myres_est_mn)
    assign(paste0("myres_est_mn_", strsplit(chems[[g]][1], "_")[[1]][1]), myres_est_mn)
    colnames(myres_p_mn) <- c("EH0", "EH0_40", "EH40", "EH20_60", "EH0_80",
        "EH40_80", "EH80")
    rownames(myres_p_mn) <- colnames(myres_est_mn)
    assign(paste0("myres_p_mn_", strsplit(chems[[g]][1], "_")[[1]][1]), myres_p_mn)

    colnames(myres_est_spread) <- c("EH0", "EH0_40", "EH40", "EH20_60", "EH0_80",
        "EH40_80", "EH80")
    rownames(myres_est_spread) <- colnames(myres_est_spread)
    assign(paste0("myres_est_spread_", strsplit(chems[[g]][1], "_")[[1]][1]),
        myres_est_spread)
    colnames(myres_p_spread) <- c("EH0", "EH0_40", "EH40", "EH20_60", "EH0_80",
        "EH40_80", "EH80")
    rownames(myres_p_spread) <- colnames(myres_est_spread)
    assign(paste0("myres_p_spread_", strsplit(chems[[g]][1], "_")[[1]][1]),
        myres_p_spread)

}
rm(m, mod, g, i, j, treattreatlist, myres_est_mn, myres_p_mn, myres_est_spread,
    myres_p_spread)


# run models for each chemical x assay environment testing treatments
# against 0.
myres0_est <- myres[, 1:4]
colnames(myres0_est) <- c("SALT_MN", "SALT_VAR", "COPR_MN", "COPR_VAR")
```

```
myres0_p <- myres0_est
counter <- 1
for (g in 1:length(chems)) {
    m <- myrcw[myrcw$epo %in% chems[[g]] & !is.na(myrcw$dw_00_80) & !is.na(myrcw$reads_dw_00_80),
        ]
    m$spread_dw_00_80 <- m$absdif_dw_00_80

    mod <- lm(dw_00_80 ~ treat + 0, weights = log(reads_dw_00_80), data = m)
    myres0_est[, counter] <- summary(mod)$coefficients[, 1]
    myres0_p[, counter] <- summary(mod)$coefficients[, 4]

    mod <- lm(spread_dw_00_80 ~ treat + 0, weights = log(reads_dw_00_80), data = m)
    myres0_est[, counter + 1] <- summary(mod)$coefficients[, 1]
    myres0_p[, counter + 1] <- pt(coef(summary(mod))[, 3], mod$df, lower = FALSE)

    print(tab_model(mod, digits = 5, digits.p = 8, show.stat = T, wrap.labels = 100,
        file = paste0("SSS_lmer_TreatEffects_VS0_", strsplit(chems[[g]][1],
            "_")[[1]][1], ".html")))
    counter <- counter + 2
}
rm(chems, m, mod, g, myres, counter, mod2, treatlist, myres0_p, myres0_est)
```

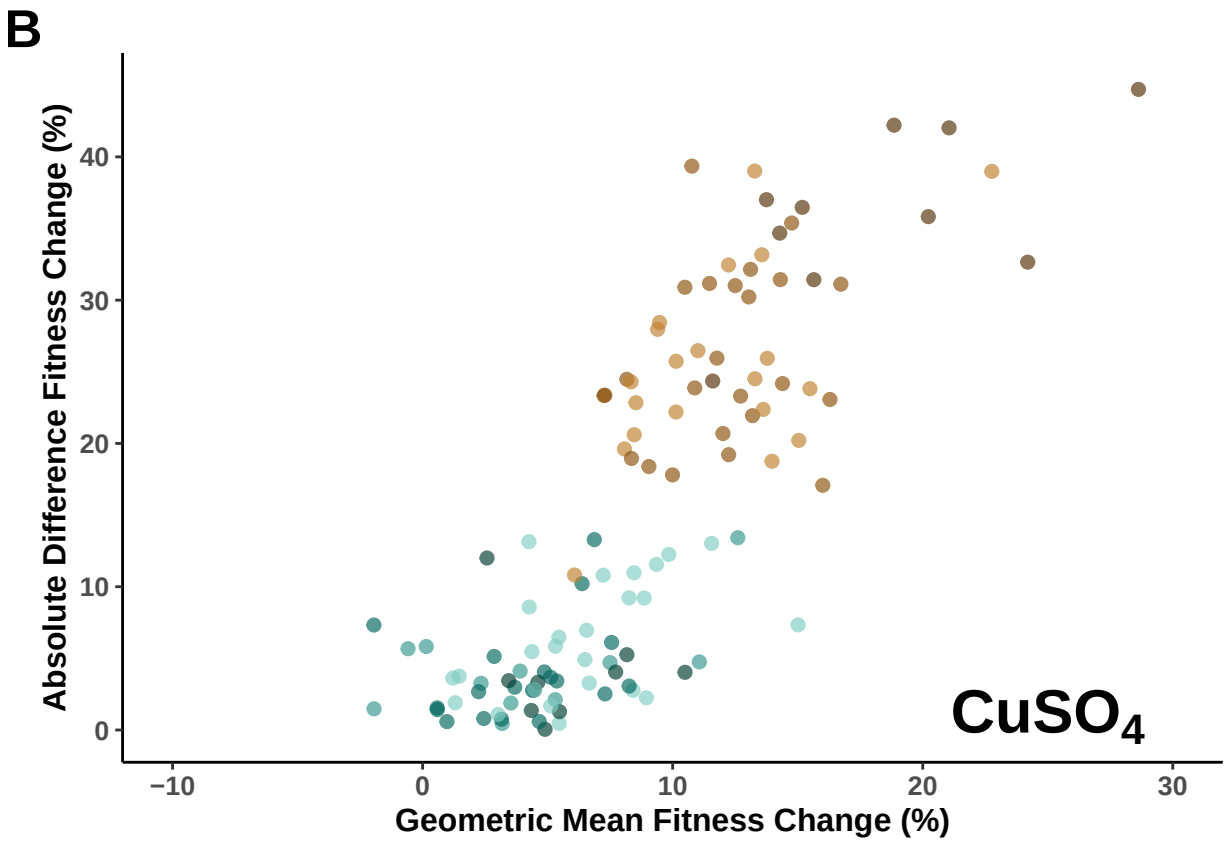
```
m1 <- myrcw[myrcw$epo %in% c("SALT_A", "SALT_B") & !is.na(myrcw$dw_00_80) &
    !is.na(myrcw$reads_dw_00_80), ]
m2 <- myrcw[myrcw$epo %in% c("COPR_A", "COPR_B") & !is.na(myrcw$dw_00_80) &
    !is.na(myrcw$reads_dw_00_80), ]
m1$spread_dw_00_80 <- m1$absdif_dw_00_80
m2$spread_dw_00_80 <- m2$absdif_dw_00_80

# control block -------------------------
fitaxdatlimMN <- c(-0.1 * 100, 0.3 * 100)  #; min(myrcw$dw_00_80, na.rm = T); max(myrcw$dw_00_80, na.rm
fitaxlablimMN <- c(-0.1 * 100, 0.3 * 100)
boxMN <- c(0.125 * 100, 0.32 * 100)  #; (boxMN[2]-boxMN[1]) / (fitaxdatlimMN[2] - fitaxdatlimMN[1])

fitaxdatlimSPREAD <- c(0, 0.45 * 100)  #; min(myrcw$absdif_dw_00_80, na.rm = T); max(myrcw$absdif_dw_00
fitaxlablimSPREAD <- c(0, 0.45 * 100)
boxSPREAD <- c(0.228 * 100, 0.449 * 100)  #; (boxSPREAD[2]-boxSPREAD[1]) / (fitaxdatlimSPREAD[2] - fita

treatlabSALT <- "NaCl Evol. Treatment"
treatlabCOPR <- expression(bold(CuSO["4"] * "  Evol. Treatment"))
fitnesslab1 <- "Geometric Mean Fitness Change (%)"
fitnesslab2 <- "Absolute Difference Fitness Change (%)"

for (i in c("SALT", "COPR")) {
    for (j in c("mn", "spread")) {
        assign("myres_est_temp", get(paste0("myres_est_", j, "_", i)))
        assign("myres_p_temp", get(paste0("myres_p_", j, "_", i)))
        myres_est_temp[which(myres_p_temp > 0.05)] <- NA
        myres_est_temp <- round(myres_est_temp * 100, 0)
        myres_est_temp <- get_lower_tri(myres_est_temp)  # uncomment for just lower triangle
        melt_myres_est_temp <- reshape::melt(myres_est_temp)
        melt_myres_est_temp$X1 <- factor(melt_myres_est_temp$X1, levels = c("EH0",
            "EH0_40", "EH40", "EH20_60", "EH0_80", "EH40_80", "EH80"))
        melt_myres_est_temp$X2 <- factor(melt_myres_est_temp$X2, levels = c("EH0",
```

```
            "EH0_40", "EH40", "EH20_60", "EH0_80", "EH40_80", "EH80"))
        assign(paste0("melt_myres_est_", j, "_", i), melt_myres_est_temp)
    }
}

pAA <- viopanel(m1, melt_myres_est_mn_SALT, melt_myres_backer, fitlims = fitaxdatlimMN,
    insetlims = boxMN, fitaxlablim = fitaxlablimMN, treatlab = treatlabSALT,
    fitlab = fitnesslab1, tag = "A", usepalette = mypalette, dcol = "dw_00_80")
## Warning: Removed 33 rows containing missing values (geom_text).
pBB <- viopanel(m1, melt_myres_est_spread_SALT, melt_myres_backer, fitlims = fitaxdatlimSPREAD,
    insetlims = boxSPREAD, fitaxlablim = fitaxlablimSPREAD, treatlab = treatlabSALT,
    fitlab = fitnesslab2, tag = "B", usepalette = mypalette, dcol = "absdif_dw_00_80",
    fitaxstep = 0.1 * 100)
## Warning: Removed 32 rows containing missing values (geom_text).
pCC <- viopanel(m2, melt_myres_est_mn_COPR, melt_myres_backer, fitlims = fitaxdatlimMN,
    insetlims = boxMN, fitaxlablim = fitaxlablimMN, treatlab = treatlabCOPR,
    fitlab = fitnesslab1, tag = "C", usepalette = mypalette, dcol = "dw_00_80")
## Warning: Removed 34 rows containing missing values (geom_text).
pDD <- viopanel(m2, melt_myres_est_spread_COPR, melt_myres_backer, fitlims = fitaxdatlimSPREAD,
    insetlims = boxSPREAD, fitaxlablim = fitaxlablimSPREAD, treatlab = treatlabCOPR,
    fitlab = fitnesslab2, tag = "D", usepalette = mypalette, dcol = "absdif_dw_00_80",
    fitaxstep = 0.1 * 100)
## Warning: Removed 35 rows containing missing values (geom_text).

setwd(DIR_out)
pdf(file = "SSS_TreatEffectViolins_GEOSPREAD.pdf", width = 11, height = 8.5)
cowplot::plot_grid(pAA, pBB, pCC, pDD, nrow = 2)
dev.off()  # one column wide, equal height
## pdf
##   2

rm(pAA, pBB, pCC, pDD, melt_myres_backer, melt_myres_est_mn_COPR, melt_myres_est_mn_SALT,
    melt_myres_est_spread_COPR, melt_myres_est_spread_SALT, melt_myres_est_temp,
    myres_p_temp, myres_p_spread_SALT, myres_p_spread_COPR, myres_p_mn_SALT,
    myres_p_mn_COPR, myres_est_temp, myres_est_spread_SALT, myres_est_spread_COPR,
    myres_est_mn_SALT, myres_est_mn_COPR, i, j, treatlabCOPR, treatlabSALT,
    fitnesslab1, fitnesslab2, boxMN, boxSPREAD, fitaxlablimMN, fitaxlablimSPREAD)
```

```
pAA <- ggplot(data = m1, aes(x = dw_00_80 * 100, y = spread_dw_00_80 * 100,
    group = treat, color = treat)) + geom_point(alpha = 0.66, pch = 19, cex = 2) +
    scale_x_continuous(breaks = seq(fitaxdatlimMN[1], fitaxdatlimMN[2], by = 0.1 *
        100), limits = c(fitaxdatlimMN[1], fitaxdatlimMN[2]), labels = scales::number_format(accuracy =
        100)) + scale_y_continuous(breaks = seq(fitaxdatlimSPREAD[1], fitaxdatlimSPREAD[2],
    by = 0.1 * 100), limits = c(fitaxdatlimSPREAD[1], fitaxdatlimSPREAD[2]),
    labels = scales::number_format(accuracy = 0.01 * 100)) + scale_color_manual(values = mypalette,
    name = "Treatment") + ylab("Absolute Difference Fitness Change (%)") + xlab("Geometric Mean Fitness
    labs(tag = "A") + theme_classic() + theme(axis.title = element_text(size = 12,
    face = "bold"), axis.text = element_text(size = 10, face = "bold"), plot.tag = element_text(face =
    size = 20), legend.position = c(0.15, 0.815), legend.background = element_rect(colour = "black",
    fill = "grey95", size = 1, linetype = "solid"), legend.key.size = unit(10,
    "pt")) + guides(color = guide_legend(reverse = TRUE)) +
annotate("text", x = 25, y = 1, label = "NaCl", fontface = "bold", size = 8)
pAA
```




```
pBB <- ggplot(data = m2, aes(x = dw_00_80 * 100, y = spread_dw_00_80 * 100,
    group = treat, color = treat)) + geom_point(alpha = 0.66, pch = 19, cex = 2) +
    scale_x_continuous(breaks = seq(fitaxdatlimMN[1], fitaxdatlimMN[2], by = 0.1 *
        100), limits = c(fitaxdatlimMN[1], fitaxdatlimMN[2]), labels = scales::number_format(accuracy =
        100)) + scale_y_continuous(breaks = seq(fitaxdatlimSPREAD[1], fitaxdatlimSPREAD[2],
    by = 0.1 * 100), limits = c(fitaxdatlimSPREAD[1], fitaxdatlimSPREAD[2]),
    labels = scales::number_format(accuracy = 0.01 * 100)) + scale_color_manual(values = mypalette,
    name = "Treatment") + ylab("Absolute Difference Fitness Change (%)") + xlab("Geometric Mean Fitness
    labs(tag = "B") + theme_classic() + theme(axis.title = element_text(size = 12,
    face = "bold"), axis.text = element_text(size = 10, face = "bold"), plot.tag = element_text(face =
    size = 20), legend.position = "none") + annotate("text", x = 25, y = 1,
    label = expression(bold(CuSO["4"])), fontface = "bold", size = 8)
pBB
## Warning in is.na(x): is.na() applied to non-(list or vector) of type
## 'expression'
```




```
setwd(DIR_out)
pdf(file = "SSS_geomean_vs_spread.pdf", width = 5, height = 9)
cowplot::plot_grid(pAA, pBB, nrow = 2)
## Warning in is.na(x): is.na() applied to non-(list or vector) of type
## 'expression'
dev.off()  # one column wide, equal height
## pdf
##   2

rm(pAA, pBB, m1, m2, fitaxdatlimMN, fitaxdatlimSPREAD)
```

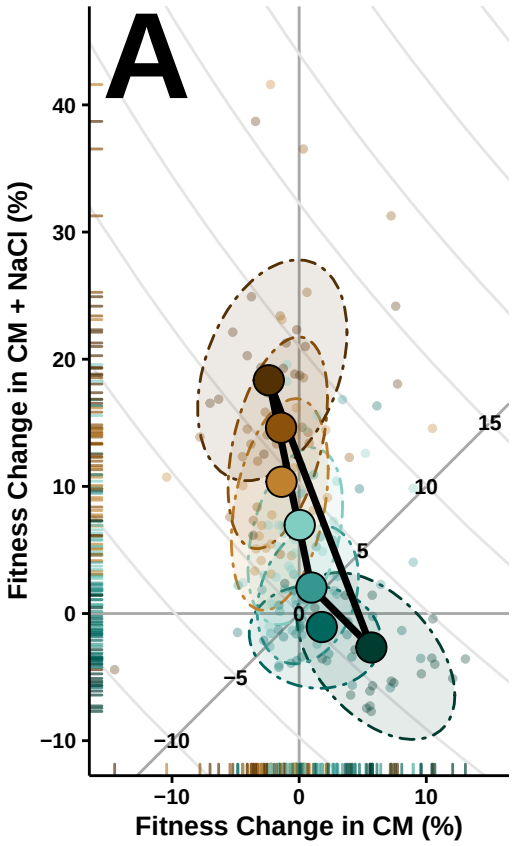
Kassen Plotting!

```
# create a matrix of the geometric fitness surface (used for plotting) -----
ll <- 1000
geomfit <- matrix(nrow = ll, ncol = ll)
x <- seq(from = 0.05, 1.6, length.out = ll)
y <- seq(from = 0.05, 1.6, length.out = ll)
colnames(geomfit) <- x
rownames(geomfit) <- y
for (i in 1:nrow(geomfit)) {
    for (j in 1:ncol(geomfit)) {
        geomfit[i, j] <- (x[j] * y[i])^(0.5)
    }
}
geomfit <- round(geomfit, digits = 4)
```

```
geomfit <- reshape::melt(geomfit)
colnames(geomfit) <- c("geomfitx", "geomfity", "geomfitfit")
geomfit <- (geomfit - 1) * 100
rm(i, j, ll, x, y)
```

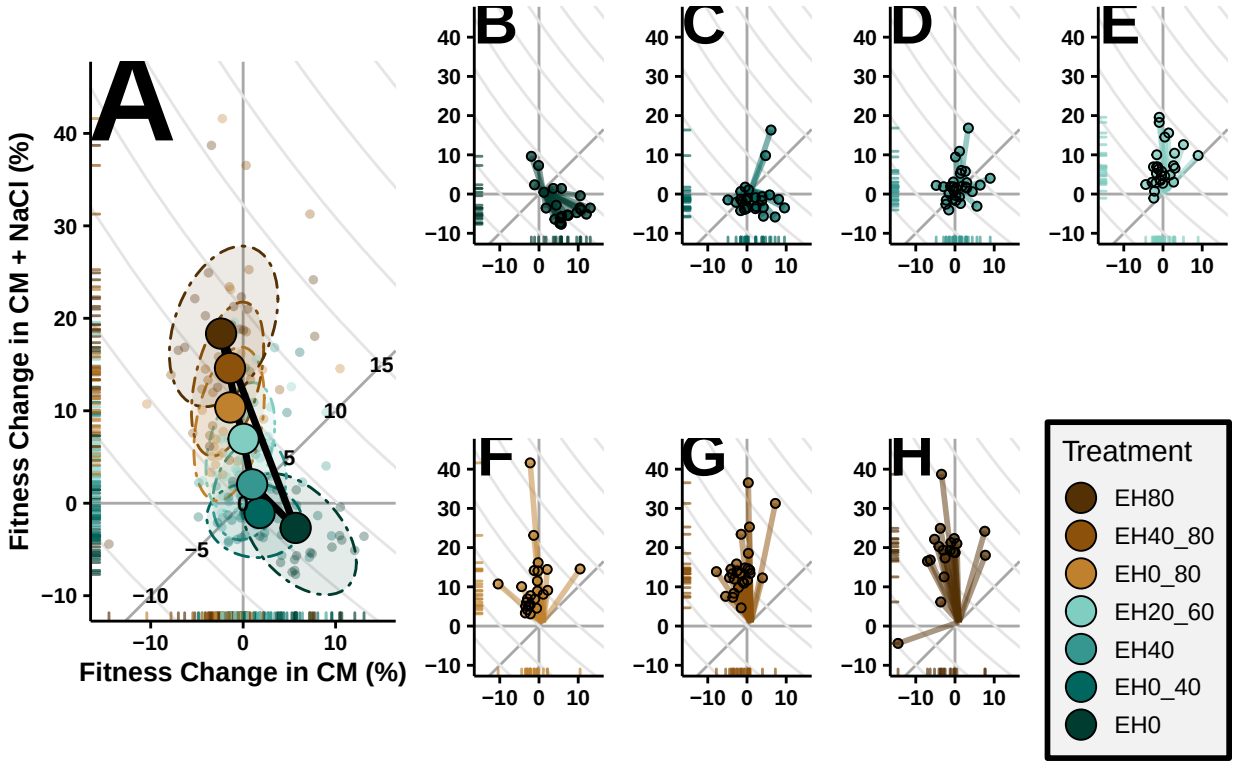
```
# get the data ------------------------------------------------
m <- myrcw[myrcw$epo %in% c("SALT_A", "SALT_B") & !is.na(myrcw$dw_00) & !is.na(myrcw$dw_80),
    ]
m$Fitness_in_envX <- m$dw_00
m$Fitness_in_envY <- m$dw_80
m$Treatment <- m$treat

# create dataframe for Treatmentment geomtric mean fitness values.
geomeans <- aggregate(cbind(m$Fitness_in_envX + 1, m$Fitness_in_envY + 1) ~
    m$treat, m, psych::geometric.mean)
# geomeans <- aggregate(cbind(m$Fitness_in_envX+1, m$Fitness_in_envY+1) ~
# m$treat, m, mean) # no qualitative difference if just 'mean' is used.
# geomeans <- aggregate(cbind(m$Fitness_in_envX+1, m$Fitness_in_envY+1) ~
# m$treat, m, median) # no qualitative difference if just 'median' is used.
colnames(geomeans) <- c("Treatment", "Fitness_in_envX", "Fitness_in_envY")
geomeans[, c(2, 3)] <- geomeans[, c(2, 3)] - 1


m$Fitness_in_envX <- m$Fitness_in_envX * 100
m$Fitness_in_envY <- m$Fitness_in_envY * 100
geomeans[, c(2, 3)] <- geomeans[, c(2, 3)] * 100


# Create the Kassen Plot --------------------------------
p1 <- ggplot() + geom_vline(xintercept = 0, color = "darkgray") + geom_hline(yintercept = 0,
    color = "darkgray") + geom_abline(slope = 1, color = "darkgray") + xlab("Fitness Change in CM (%)") +
    ylab("Fitness Change in CM + NaCl (%)") + coord_equal(xlim = c(-15, 15),
    ylim = c(-10, 45)) + theme_classic() + scale_color_manual(values = mypalette) +
    scale_fill_manual(values = mypalette)
for (i in c(0.9, 0.95, 1, 1.05, 1.1, 1.15, 1.2, 1.25, 1.3)) {
    p1 <- p1 + geom_line(data = geomfit[geomfit$geomfitfit == (i - 1) * 100,
        ], aes(x = geomfitx, y = geomfity), lty = "solid", color = "gray90")
}
p1 <- p1 + stat_ellipse(data = m, aes(x = Fitness_in_envX, y = Fitness_in_envY,
    fill = Treatment, color = Treatment), alpha = 0.1, geom = "polygon", linetype = 6,
    show.legend = F, level = 0.75) + geom_point(data = m, aes(x = Fitness_in_envX,
    y = Fitness_in_envY, color = Treatment), size = 1, alpha = 0.33, show.legend = F) +
    geom_segment(mapping = aes(x = geomeans[1, 2], y = geomeans[1, 3], xend = geomeans[7,
        2], yend = geomeans[7, 3]), linetype = 1, size = 1.25, inherit.aes = F) +
    geom_segment(mapping = aes(x = geomeans[3, 2], y = geomeans[3, 3], xend = geomeans[7,
        2], yend = geomeans[7, 3]), linetype = 1, size = 1.25, inherit.aes = F) +
    geom_segment(mapping = aes(x = geomeans[1, 2], y = geomeans[1, 3], xend = geomeans[3,
        2], yend = geomeans[3, 3]), linetype = 1, size = 1.25, inherit.aes = F) +
    geom_point(data = geomeans, aes(x = Fitness_in_envX, y = Fitness_in_envY,
        fill = Treatment), color = "black", cex = 5, shape = 21, show.legend = T) +
    annotate("text", x = c((0.9 - 1) * 100, (0.95 - 1) * 100, (1 - 1) * 100,
        (1.05 - 1) * 100, (1.1 - 1) * 100, (1.15 - 1) * 100), y = c((0.9 - 1) *
        100, (0.95 - 1) * 100, (1 - 1) * 100, (1.05 - 1) * 100, (1.1 - 1) *
        100, (1.15 - 1) * 100), label = c("-10", "-5", "0", "5", "10", "15"),
```

```
        size = 2.91, fontface = "bold") + geom_rug(data = m, aes(x = Fitness_in_envX,
    color = treat), length = unit(0.015, "npc"), alpha = 0.66, show.legend = F) +
    geom_rug(data = m, aes(y = Fitness_in_envY, color = treat), length = unit(0.03,
        "npc"), alpha = 0.66, show.legend = F) + theme(axis.title = element_text(size = 10,
    face = "bold", color = "black"), axis.ticks = element_line(color = "black"),
    axis.text = element_text(size = 8, face = "bold", color = "black"), legend.key.size = unit(10,
        "pt"), legend.background = element_rect(colour = "black", fill = "grey95",
        size = 1, linetype = "solid")) + guides(fill = guide_legend(reverse = TRUE)) +
    guides(color = guide_legend(reverse = TRUE)) + annotate("text", x = -12,
    y = 44, label = "A", fontface = "bold", size = 16)
p1leg <- cowplot::get_legend(p1)
p1 <- p1 + theme(legend.position = "none")
p1
```




```
# Create the Vector Fitness Panels -------------------------
vectorpanel <- function(ptreat, palettecolorpos, ptitle) {
    pZZ <- ggplot() + geom_vline(xintercept = 0, color = "darkgray") + geom_hline(yintercept = 0,
        color = "darkgray") + geom_abline(slope = 1, color = "darkgray") + xlab("") +
        ylab(NULL) + coord_equal(xlim = c(-15, 15), ylim = c(-10, 45)) + theme_classic()
    for (i in c(0.9, 0.95, 1, 1.05, 1.1, 1.15, 1.2, 1.25, 1.3)) {
        pZZ <- pZZ + geom_line(data = geomfit[geomfit$geomfitfit == (i - 1) *
            100, ], aes(x = geomfitx, y = geomfity), lty = "solid", color = "gray90")
    }
    pZZ <- pZZ + geom_segment(data = m[m$treat %in% c(ptreat), ], aes(x = 1,
        y = 1, xend = m$Fitness_in_envX[m$treat %in% c(ptreat)], yend = m$Fitness_in_envY[m$treat %in%
            c(ptreat)]), color = mypalette[palettecolorpos], size = 2/2, inherit.aes = F,
        alpha = 0.5) + geom_point(data = m[m$treat %in% c(ptreat), ], aes(x = m$Fitness_in_envX[m$treat
```

```
        c(ptreat)], y = m$Fitness_in_envY[m$treat %in% c(ptreat)]), color = mypalette[palettecolorpos],
        fill = mypalette[palettecolorpos], size = 2.5/2, alpha = 0.75) + geom_point(data = m[m$treat %i
        c(ptreat), ], aes(x = m$Fitness_in_envX[m$treat %in% c(ptreat)], y = m$Fitness_in_envY[m$treat
        c(ptreat)]), shape = 1, colour = "black", size = 2.5/2) + geom_rug(data = m[m$treat %in%
        c(ptreat), ], aes(x = Fitness_in_envX, color = treat), length = unit(0.03,
        "npc"), alpha = 0.66, color = mypalette[palettecolorpos]) + geom_rug(data = m[m$treat %in%
        c(ptreat), ], aes(y = Fitness_in_envY, color = treat), length = unit(0.06,
        "npc"), alpha = 0.66, color = mypalette[palettecolorpos]) + theme(axis.title = element_text(siz
        face = "bold", color = "black"), axis.ticks = element_line(color = "black"),
        axis.text = element_text(size = 8, face = "bold", color = "black"),
        legend.position = "none") + annotate("text", x = -11, y = 44, label = ptitle,
        fontface = "bold", size = 8)
    pZZ
}

p2 <- vectorpanel("EH0", 1, "B")
p3 <- vectorpanel("EH0_40", 2, "C")
p4 <- vectorpanel("EH40", 3, "D")
p5 <- vectorpanel("EH20_60", 4, "E")
p6 <- vectorpanel("EH0_80", 5, "F")
p7 <- vectorpanel("EH40_80", 6, "G")
p8 <- vectorpanel("EH80", 7, "H")

require(grid)
## Loading required package: grid
require(gridExtra)
## Loading required package: gridExtra
lay <- as.matrix(rbind(cbind(1, 1, 2, 3, 4, 5), cbind(1, 1, 6, 7, 8, 9)))
x <- grid.arrange(p1, p2, p3, p4, p5, p6, p7, p8, p1leg, layout_matrix = lay)
```



```r
setwd(DIR_out)
pdf(file = paste("SSS_SALT_KASSEN.pdf"), height = 6, width = 11)
grid::grid.draw(x)
dev.off()
## pdf
##   2
```