FS Analysis

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

Outline: 1 - Prepare the workspace (scrub the environment, load required packages, set working directories, create helper functions). 2 - Library Counts. 3 - Barcode Cross-Contamination (Reference purity, Single-well controls). 4 - Power (indiv bc fitness change, treatment effects). 4.5: Treat Effects Stats (ALL) 5 - Treat Effects PANEL: 00 env, SALT 6 - Treat Effects PANEL: 00 env, COPR 7 - Treat Effects PANEL: 40 env, SALT 8 - Treat Effects PANEL: 40 env, COPR 9 - Treat Effects PANEL: 80 env, SALT 10 - Treat Effects PANEL: 80 env, COPR 11 - Treat Effects PANEL: 00-80 Geomean, SALT 12 - Treat Effects PANEL: 00-80 Geomean, COPR 13 - Treat Effects PANEL: 00-80 VarFit, SALT 14 - Treat Effects PANEL: 00-80 VarFit, COPR 15 - Treat Effects PANEL: 00-40 Geomean, SALT 16 - Treat Effects PANEL: 00-40 Geomean, COPR 17 - Treat Effects PANEL: 00-40 VarFit, SALT 18 - Treat Effects PANEL: 00-40 VarFit, COPR 19 - Kassen PANEL: 00-80, SALT 20 - Kassen PANEL: 00-80, COPR 21 - Kassen PANEL: 00-40, SALT 22 - Kassen PANEL: 00-40, COPR 23 - Vector Fitness PANEL: 00-80, SALT 24 - Vector Fitness PANEL: 00-80, COPR 25 - Vector Fitness PANEL: 00-40, SALT 26 - Vector Fitness PANEL: 00-40, COPR

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(grid)
```

```
## Loading required package: grid
```

```
require(gridExtra)
```

```
## Loading required package: gridExtra
```

```
# 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)
}

# %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)]
```

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
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)"
setwd(DIR_out)
pdf("SSS_CountsHists.pdf", height = 7, width = 7)
hist((unlist(temp)), breaks = 100, main = "Histogram of all barcode counts")
hist(log(unlist(temp)), breaks = 100, main = "Histogram of all (log-) barcode counts")

temp <- 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(temp)
##    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(temp)))
## [1] "Counts per barcode is extremely variable across barcodes due to treatment effects in fitness as
hist((temp), breaks = 25, main = "Histogram of barcode mean counts \nacross all fitness assay samples")
hist(log(temp), breaks = 25, main = "Histogram of (log-) barcode mean counts \nacross all fitness assay
dev.off()
## pdf
##   2

rm(myd, temp, lowcut)
```

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

```
# Reference purity.
# ------------------------------------------------------------
```

```
setwd(DIR_fit_formatted)
load("swref.rdata")
swref$ppcctsm <- swref$ppccts/112  # counts per contaminant
swref$ccm <- swref$cc/112  # cc rate per contaminant
temp <- swref[, c("ppects", "ppccts", "ppcctsm", "cc", "ccm")]
rm(swref)
setwd(DIR_out)
write.csv(temp, file = "SSS_ReferenceCC.csv")

print("cross-contamination rate data (MEAN, SD of 5 samples)")
## [1] "cross-contamination rate data (MEAN, SD of 5 samples)"
apply(temp, 2, mean, na.rm = T)
##               ppects               ppccts              ppcctsm
## 1328786.399999999907     802.600000000000       7.166071428571
##                   cc                  ccm
##       0.000596278601       0.000005323916
apply(temp, 2, sd, na.rm = T)
##              ppects              ppccts             ppcctsm
## 158208.519787336350    412.513393721949     3.683155301089
##                   cc                  ccm
##      0.000270146105      0.000002412019

setwd(DIR_out)
pdf("SSS_ReferenceCC.pdf", height = 7, width = 7)
plot(temp$cc * 100, pch = 19, ylim = c(0, 2), main = "Reference BC TOTAL CC%")
plot(temp$ccm * 100, pch = 19, ylim = c(0, 2/110), main = "Reference MEAN BC CONTAMINANT CC%")
dev.off()
## pdf
##   2

# single well controls
# ------------------------------------------------------------
setwd(DIR_evo_formatted)
load("swref.rdata")
temp <- swref[, c(1, 7, 8, 361:363)]
rm(swref)  # summarize replicates (where they exist)
tem <- temp[temp$replicate %in% c("", "R=1"), ]
tem[1, 4:6] <- apply(temp[1:3, 4:6], 2, mean)
tem[2, 4:6] <- apply(temp[4:6, 4:6], 2, mean)
tem[3, 4:6] <- apply(temp[7:9, 4:6], 2, mean)
tem[4, 4:6] <- apply(temp[10:12, 4:6], 2, mean)
tem[5, 4:6] <- apply(temp[13:15, 4:6], 2, mean)
tem[6, 4:6] <- apply(temp[16:18, 4:6], 2, mean)
temp <- tem[1:9, ]  # omit sulfer data for which no fitness assays were run.
temp$ppcctsm <- temp$ppccts/110  # counts per contaminant
temp$ccm <- temp$cc/110  # cc rate per contaminant
temp <- temp[, c("ppects", "ppccts", "ppcctsm", "cc", "ccm")]
setwd(DIR_out)
write.csv(temp, file = "SSS_SWControlCC.csv")

print("Ancestral CC rate data (MEAN, SD of 3 samples, each itself the mean of 3 triplicate measurements)
## [1] "Ancestral CC rate data (MEAN, SD of 3 samples, each itself the mean of 3 triplicate measurements
apply(temp[1:3, ], 2, mean)
```

```
##              ppects              ppccts             ppcctsm
## 282917.666666666628     30.888888888889      0.280808080808
##                  cc                 ccm
##      0.000111594788      0.000001014498
apply(temp[1:3, ], 2, sd)
##              ppects              ppccts             ppcctsm
## 30341.1955891289399    19.7802744990877     0.1798206772644
##                  cc                 ccm
##     0.0000726603037     0.0000006605482

print("SALT D50 CC rate data (MEAN, SD of 3 samples, each itself the mean of 3 triplicate measurements)")
## [1] "SALT D50 CC rate data (MEAN, SD of 3 samples, each itself the mean of 3 triplicate measurements)"
apply(temp[4:6, ], 2, mean)
##              ppects              ppccts             ppcctsm
## 220910.444444444438     71.666666666667      0.651515151515
##                  cc                 ccm
##      0.000302852194      0.000002753202
apply(temp[4:6, ], 2, sd)
##             ppects             ppccts            ppcctsm
## 66269.548164492007    46.279345044823     0.420721318589
##                  cc                 ccm
##     0.000121667560     0.000001106069

print("COPR D50 CC rate data (MEAN, SD of 3 samples, each sample only measured in singleton)")
## [1] "COPR D50 CC rate data (MEAN, SD of 3 samples, each sample only measured in singleton)"
apply(temp[7:9, ], 2, mean)
##            ppects            ppccts           ppcctsm                cc
## 338887.0000000000   1758.3333333333     15.9848484848      0.0076495212
##               ccm
##      0.0000695411
apply(temp[7:9, ], 2, sd)
##             ppects             ppccts            ppcctsm
## 135123.00683821389   1656.23981757876     15.05672561435
##                  cc                 ccm
##      0.00990196343      0.00009001785

print("ALL samples CC rate data (MEAN, SD of 9 samples, 1:6 each the mean of 3 triplicate measurments, 
## [1] "ALL samples CC rate data (MEAN, SD of 9 samples, 1:6 each the mean of 3 triplicate measurments,
apply(temp, 2, mean)
##             ppects             ppccts            ppcctsm
## 280905.03703703702    620.29629629630     5.63905723906
##                  cc                 ccm
##      0.00268798940      0.00002443627
apply(temp, 2, sd)
##            ppects            ppccts           ppcctsm                cc
## 92220.43281158894  1189.63746163420    10.81488601486     0.00619443686
##               ccm
##     0.00005631306

setwd(DIR_out)
pdf("SSS_SWControlCC.pdf", height = 7, width = 7)
plot(temp$cc * 100, pch = 19, ylim = c(0, 2), main = "SW Control BC TOTAL CC%")
abline(v = c(3.5, 6.5))
```

```
text(2, 1.75, "Ancestral")
text(5, 1.75, "SALT D50")
text(8, 1.75, "COPR D50")
plot(temp$ccm * 100, pch = 19, ylim = c(0, 2/110), main = "SW Control MEAN BC CONTAMINANT CC%")
abline(v = c(3.5, 6.5))
text(2, 1.75/110, "Ancestral")
text(5, 1.75/110, "SALT D50")
text(8, 1.75/110, "COPR D50")
dev.off()
## pdf
##   2

rm(temp, tem)
```

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

```
setwd(DIR_fit_formatted)
load("FitAssayData.rdata")

# 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.

# prep frame & calculate psd -----------
temp <- myrc
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)$power * 100,
    "% power to detect a 1% fitness change for an average BC"))
## [1] "25.8906869607391% power to detect a 1% fitness change for an average BC"
print(paste0("80% power to detect a ", pwr.t.test(power = 0.8, n = 4, sig.level = 0.05)$d *
    psd * 100, "% fitness change for an average BC"))
## [1] "80% power to detect a 2.16818600714925% 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)
```

```r
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
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 = "dashed", color = "gray") +
    geom_segment(x = 0, xend = pwr.t.test(power = 0.8, n = 4, sig.level = 0.05)$d *
        psd, y = 0.8, yend = 0.8, lty = "dashed", color = "gray") + geom_point(x = pwr.t.test(power = 0
    n = 4, sig.level = 0.05)$d * psd, y = 0.8, pch = 19, size = 3, color = "darkgray") +
    geom_text(x = 2.6/100, y = 0.74, label = paste0(round(pwr.t.test(power = 0.8,
        n = 4, sig.level = 0.05)$d * psd * 100, 2), "%", ",\n80%    "), color = "darkgray",
        size = 2.5) + geom_line() + scale_x_continuous(name = "Fitness Change\n (for Individual Strains
    breaks = seq(0, 0.04, by = 0.005), labels = paste0(seq(0, 0.04, by = 0.005) *
        100, "%"), limits = c(0, 0.04), expand = expand_scale(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 = expand_scale(mult = c(0,
        0.02))) + theme_classic() + labs(tag = "A") + theme(axis.text.x = element_text(angle = 45,
    hjust = 1))




# 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 u 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] 2.410952
myeffectsize <- 1/myrmse
myeffectsize
## [1] 0.4147739
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 = ((32 * 2) - 1 - 1), f2 = myeffectsize, sig.level = 0.05)$power *
    100, "% power to detect a 1% fitness change for an average BC (assuming no ext lineages)"))
## [1] "99.9067878783826% power to detect a 1% fitness change for an average BC (assuming no ext lineag
print(paste0("80% power to detect a ", pwr.f2.test(u = 1, v = ((32 * 2) - 1 -
    1), power = 0.8, sig.level = 0.05)$f2 * myrmse, "% fitness change for an average BC (assuming no ex
## [1] "80% power to detect a 0.305102007490767% fitness change for an average BC (assuming no ext linea
print(paste0(pwr.f2.test(u = 1, v = ((16 * 2) - 1 - 1), f2 = myeffectsize, sig.level = 0.05)$power *
    100, "% power to detect a 1% fitness change for an average BC (assuming 50% ext lineages)"))
## [1] "94.1126785092135% power to detect a 1% fitness change for an average BC (assuming 50% ext lineag
print(paste0("80% power to detect a ", pwr.f2.test(u = 1, v = ((16 * 2) - 1 -
    1), power = 0.8, sig.level = 0.05)$f2 * myrmse, "% fitness change for an average BC (assuming 50% e
## [1] "80% power to detect a 0.631557532773799% fitness change for an average BC (assuming 50% ext lin

# 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$v <- c(rep((32 * 2) - 1 - 1, times = 100), rep((16 * 2) - 1 - 1, times = 100))
powerTR$u <- 1
powerTR$n <- powerTR$v + 2
powerTR$n <- factor(powerTR$n, levels = c(64, 32))
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
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 = ((32 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, xend = pwr.f2.test(u = 1,
    v = ((32 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, y = 0,
    yend = 0.8, lty = "dashed", color = "gray") + geom_segment(x = pwr.f2.test(u = 1,
    v = ((16 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, xend = pwr.f2.test(u = 1,
    v = ((16 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, y = 0,
    yend = 0.8, lty = "dashed", color = "gray") + geom_segment(x = 0, xend = pwr.f2.test(u = 1,
    v = ((16 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, y = 0.8,
    yend = 0.8, lty = "dashed", color = "gray") + geom_point(x = pwr.f2.test(u = 1,
    v = ((32 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, y = 0.8,
    pch = 19, size = 3, color = "darkgray") + geom_point(x = pwr.f2.test(u = 1,
    v = ((16 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, y = 0.8,
    pch = 19, size = 3, color = "darkgray") + geom_text(x = 0.225, y = 0.875,
    label = paste0(round(pwr.f2.test(u = 1, v = ((32 * 2) - 1 - 1), power = 0.8,
        sig.level = 0.05)$f2 * myrmse, 2), "%,\n80%    "), color = "darkgray",
    size = 2) + geom_text(x = 0.75, y = 0.74, label = paste0(round(pwr.f2.test(u = 1,
    v = ((16 * 2) - 1 - 1), power = 0.8, sig.level = 0.05)$f2 * myrmse, 2),
    "%,\n80%    "), color = "darkgray", size = 2) + geom_line(aes(lty = n)) +
    scale_x_continuous(name = "Fitness Difference\n (Between Treatments) \n ",
        breaks = seq(0, 1.5, by = 0.25), labels = paste0(seq(0, 1.5, by = 0.25),
            "%"), limits = c(0, 1.5), expand = expand_scale(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 = expand_scale(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 panels A&B in a single figure ---
mywidth <- 3.345
myheight <- 6.69
pdf(file = "000_TEST_POWER_FIG.pdf", width = mywidth, height = myheight)
grid.arrange(pA, pB, layout_matrix = rbind(c(1, 1, 1, 1), c(1, 1, 1, 1), c(1,
    1, 1, 1), c(1, 1, 1, 1), c(2, 2, 2, 2), c(2, 2, 2, 2), c(2, 2, 2, 2), c(2,
    2, 2, 2)))
dev.off()  # one column wide, equal height
## pdf
##   2
rm(pA, pB, powerBC, powerTR, myeffectsize, myrmse, psd)
```