Evaluate How `song_count` changes with trial at 40C

Michael Gilchrist

date: 2022-10-20

Goal

- Evaluate trends in song_count under a (near) constant temperature.

Set up

Load libraries

```
## load libraries
library(stats)
require(MASS) # provides negative binomial fitting:  glm.nb
```

```
## Loading required package: MASS
```

```
library(RSQLite)  # Don't think we need this.
library(rTPC)  ##
library(nls.multstart)
library(broom)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------- tidyverse
## 1.3.2 --
```

```
## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::select() masks MASS::select()
```

```
library(ggplot2)
library(ggpubr)
library(grid) ## provides textGrob
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(viridisLite)

#options(ggplot2.continuous.colour="viridis",
#        ggplot2.discrete.colour="viridis",
#        ggplot2.scale_fill_discrete = scale_fill_viridis_d,
#        ggplot2.scale_fill_continuous = scale_fill_viridis_c)

library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
##
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(lme4)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
library(nlme)
```

```
##
## Attaching package: 'nlme'
##
## The following object is masked from 'package:lme4':
##
##     lmList
##
## The following object is masked from 'package:dplyr':
##
##     collapse
```

```
library(gnm)
library(rsample) ## provides bootstraps()

library(RVAideMemoire) # provides overdisp.glmer()
```

```
## *** Package RVAideMemoire v 0.9-81-2 ***
##
## Attaching package: 'RVAideMemoire'
##
## The following object is masked from 'package:gnm':
##
##     se
##
## The following object is masked from 'package:lme4':
##
##     dummy
##
## The following object is masked from 'package:broom':
##
##     bootstrap
```

```
library(humidity) ## provides VPD
library(weathermetrics)
library(latex2exp)
```

Local Functions

```
kprint <- function(input, ...) {
    print(knitr::kable(input, ...))
cat('\n\n<!-- -->\n\n')
}
```

Plotting settings

```
## From: https://data-se.netlify.app/2018/12/12/changing-the-default-color-scheme-in-ggplot2/

theme_set(theme_minimal(base_size = 9))
theme_update(
    plot.title = element_text(size = rel(1.1)),
    plot.subtitle = element_text(size = rel(1)))

if(!exists("old_opts")) old_opts <- options()  # save old options

options(ggplot2.continuous.colour="viridis")
options(ggplot2.continuous.fill = "viridis")
options(ggplot2.discrete.colour="viridis")
options(ggplot2.discrete.fill = "viridis")
```

Load Data

```
## Read in ZEFI Data sets
## Treat 'repeatability' as round = 0
## Add round info

## Repeatability was done between round 1 and 2, female was present, but only one temp. so treating as

output_dir <- "output"

if(!dir.exists(output_dir)) dir.create(output_dir)
git_root <- system("git rev-parse --show-toplevel", intern = TRUE)

data_raw = list()

data_raw[[1]] <- read.csv(file.path(git_root, "data", "raw_data", "HSPi-Round-1-Heat-Trials.csv")) %>% r
    ## Note T237 and T230 are missing numbers in the song_count column
    ## so we are filtering these observations out until they are found
    filter(!is.na(song_count))

data_raw[[2]] <- read.csv(file.path(git_root, "data", "raw_data", "HSPi-Repeatability-Song-Count.csv"))
    mutate(round = 2) %>%
    ungroup()

data_raw[[3]] <-read.csv(file.path(git_root, "data", "raw_data", "HSPi-Round-2-Heat-Trials.csv")) %>%
    mutate(round = 3) %>%
    ## Deal with missing temp_mean and humidity_mean values
    ## in round == 3
    ## 2022/10/19 - code no longer needed
    ## group_by(temp_target) %>%
    ##mutate(temp = if_else((round == 3 & is.na(temp_mean)),
    ##                      mean(temp_mean, na.rm = TRUE),
    ##                      temp_mean)) %>%
    ##mutate(humidity = if_else((round == 3 & is.na(humidity_mean)),
    ##                          mean(humidity_mean, na.rm = TRUE),
    ##                          humidity_mean)) %>%
    ungroup()


## Join data and discard empty columns
data_full <- full_join(data_raw[[1]], data_raw[[2]]) %>%
    full_join(data_raw[[3]]) %>%
    discard(~all(is.na(.) | . =="")) %>% ## get rid of columns of only NA
    mutate(trial_completed = !(is.na(song_count)) ) %>%
    mutate(song_count = ifelse(is.na(song_count), 0, song_count)) %>%
    mutate(song_count = song_count*1.0) %>% ## convert to a double so it's not treated as an integer
    mutate(chamber = as.factor(chamber), male = as.factor(male)) %>%
    ## create a male specific round and global trial index `trial`
    group_by(male, round) %>%
    mutate(trial_round = rank(date)) %>%
    ungroup(round) %>%
    mutate(trial = rank(date)) %>%
    ungroup() %>%
```

```
    mutate(song_count_plus_1 = (song_count + 1)) %>%
    mutate(log_song_count_plus_1 = log(song_count + 1)) %>%
    mutate(temp_target = as.numeric(temp_target)) %>%
    ## Create generic 'temp' column which is either
    ## temp_mean, if it exists, or temp_target, if it doesn't
    mutate(temp = if_else(is.na(temp_mean),
                          temp_target,
                          temp_mean)) %>%
    ## Add column with total song_count for a given round
    group_by(male, round) %>%
    mutate(count_total_round = sum(song_count)) %>%
    ungroup() %>%
    mutate(song_prop = song_count/count_total_round) %>%
    ## assuming poisson error
    ## From glm man page
    ## > Non-'NULL' 'weights' can be used to indicate that different
    ## >  observations have different dispersions (with the values in
    ## >  'weights' being inversely proportional to the dispersions);
    ## add +1 to deal with single 0
    ## Interpret dispersion as ~sd() or se() not var()
    mutate(count_wt = sqrt(1/(song_count + 1))) %>%
    ## need to rescale wts for song_prop data
    mutate(prop_wt = count_wt * count_total_round) %>%
    ## Add vpd
    mutate(svp = SVP(t = temp_mean + 273.15, isK = TRUE), vpd = svp*(1-humidity_mean/100) ) %>%
    group_by(round) %>%
    mutate(vpd_offset = vpd - mean(vpd)) %>%
    ungroup() %>%
    relocate(song_count, song_prop, vpd, temp_mean, humidity_mean, .after = male) %>%
    mutate() ## Dummy function so we can comment out lines above it w/o any issues
```

```
## Joining, by = c("male", "chamber", "date", "song_count", "counter", "temp_target",
## "round")
## Joining, by = c("male", "chamber", "date", "song_count", "counter", "test_order",
## "temp_target", "round")
```

Examine Data

Create Working Dataset

```
data_ind <- data_full %>%
#    filter(round %in% c(2,3)) %>%
#    filter(count_total_round >= 1) %>%
    mutate()

## copy data frame and assign `male =  "combined")
data_comb <- data_ind %>% mutate(male = "combined")

data <- bind_rows(data_ind, data_comb)
```

Examine How Var varies with Mean `temp_target = 40`

WARNING: `warning()` are off

```
data_40  <- data_ind %>%
    filter(temp_target == 40) %>%
    unique()

dim(data_40)
```

```
## [1] 53 30
```

```
stats_40 <- data_40 %>%
    group_by(male) %>%
    summarize(count = length(song_count),
              mean = mean(song_count),
              var = var(song_count, na.rm = TRUE),
              cv = sqrt(var)/mean,
              dispersion = var/mean) %>% mutate()

print("We have 15 males, 5 of which we only have 1 observation at 40C")
```

```
## [1] "We have 15 males, 5 of which we only have 1 observation at 40C"
```

```
stats_40
```

```
## # A tibble: 15 x 6
##    male  count   mean    var     cv dispersion
##    <fct> <int>  <dbl>  <dbl>  <dbl>      <dbl>
##  1 T229      5 159.   8820.   0.592      55.6
##  2 T230      4   2.25   20.2  2           9
##  3 T231      5 106.   9770.   0.929      91.8
##  4 T234      5  53.4   688.   0.491      12.9
##  5 T235      5  78.4   890.   0.381      11.4
##  6 T236      5 184.   3233.   0.310      17.6
##  7 T237      4 145.    693.   0.181       4.77
##  8 T243      5 229    4218.   0.284      18.4
##  9 T244      5  78.6  1477.   0.489      18.8
## 10 T246      1   0      NA   NA          NA
## 11 T247      5 118.   3223.   0.483      27.4
## 12 T257      1 253      NA   NA          NA
## 13 T258      1  45      NA   NA          NA
## 14 T259      1   0      NA   NA          NA
## 15 T260      1  43      NA   NA          NA
```

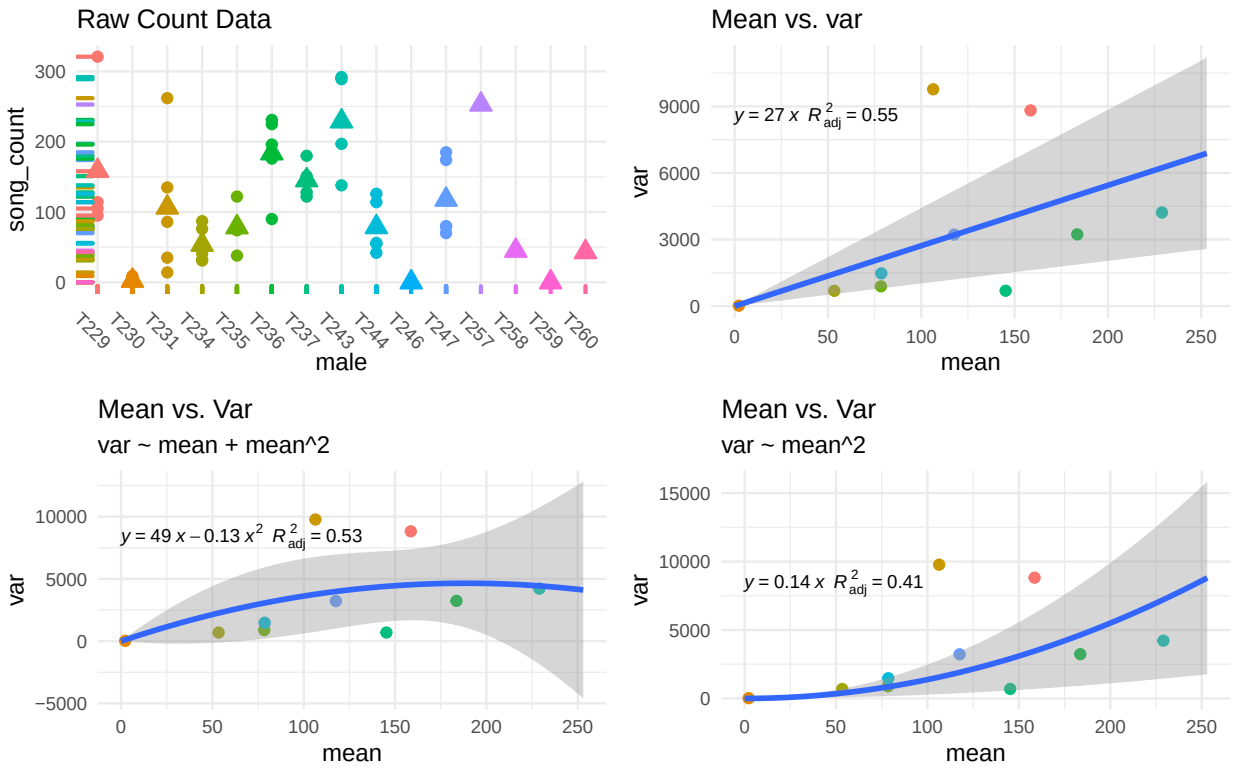
```
song_stats <- stats_40

print("Flag birds birds with a coefficient of variation `cv` > 0.5 & `mean` < 50")
```

```
## [1] "Flag birds birds with a coefficient of variation ‘cv‘ > 0.5 & ‘mean‘ < 50"
```

```r
cv_threshold <- 0.5
mean_threshold <- 50

male_high_cv <- stats_40 %>% filter(cv > cv_threshold) %>% pull(male)
data_male_high_cv <- data_40 %>% filter(male %in% male_high_cv)

male_low_mean <- stats_40 %>% filter(mean < mean_threshold) %>% pull(male)
data_male_low_mean <- data_40 %>% filter(male %in% male_low_mean)


g0 <- ggplot(data_male_high_cv) +
    aes(x=song_count,
        color = male,
        fill = male) +
    geom_histogram() +
    labs(title = paste0("Data for males with exceptionally high cviances", paste(male_high_cv, collapse
         )


for(filter_high_cv in c(FALSE, TRUE)){
    for(filter_low_mean in c(FALSE, TRUE)){

        data_tmp <- data_40
        song_stats_tmp <- song_stats

        if(filter_high_cv){
            data_tmp <- filter(data_tmp, !(male %in% male_high_cv) )
            song_stats_tmp <- filter(song_stats_tmp, !(male %in% male_high_cv) )
        }

        if(filter_low_mean){
            data_tmp <- filter(data_tmp, !(male %in% male_low_mean) )
            song_stats_tmp <- filter(song_stats_tmp, !(male %in% male_low_mean) )
        }

        g0 <- ggplot(data_tmp) +
            aes(x=male,
                y = song_count,
                color = male,
                fill = male) +
            geom_point() +
            geom_rug(size=0.75) +
            ## theme(axis.ticks.x=element_line(size=30))+
            geom_point(
                mapping = aes(x = male,
                              y = mean,
                              color = male),
                size = 2.5,
                shape = 24,
                data = song_stats_tmp) +
            theme(legend.position = "none", axis.text.x = element_text(angle = -45)) +
            labs(title = "Raw Count Data")
```

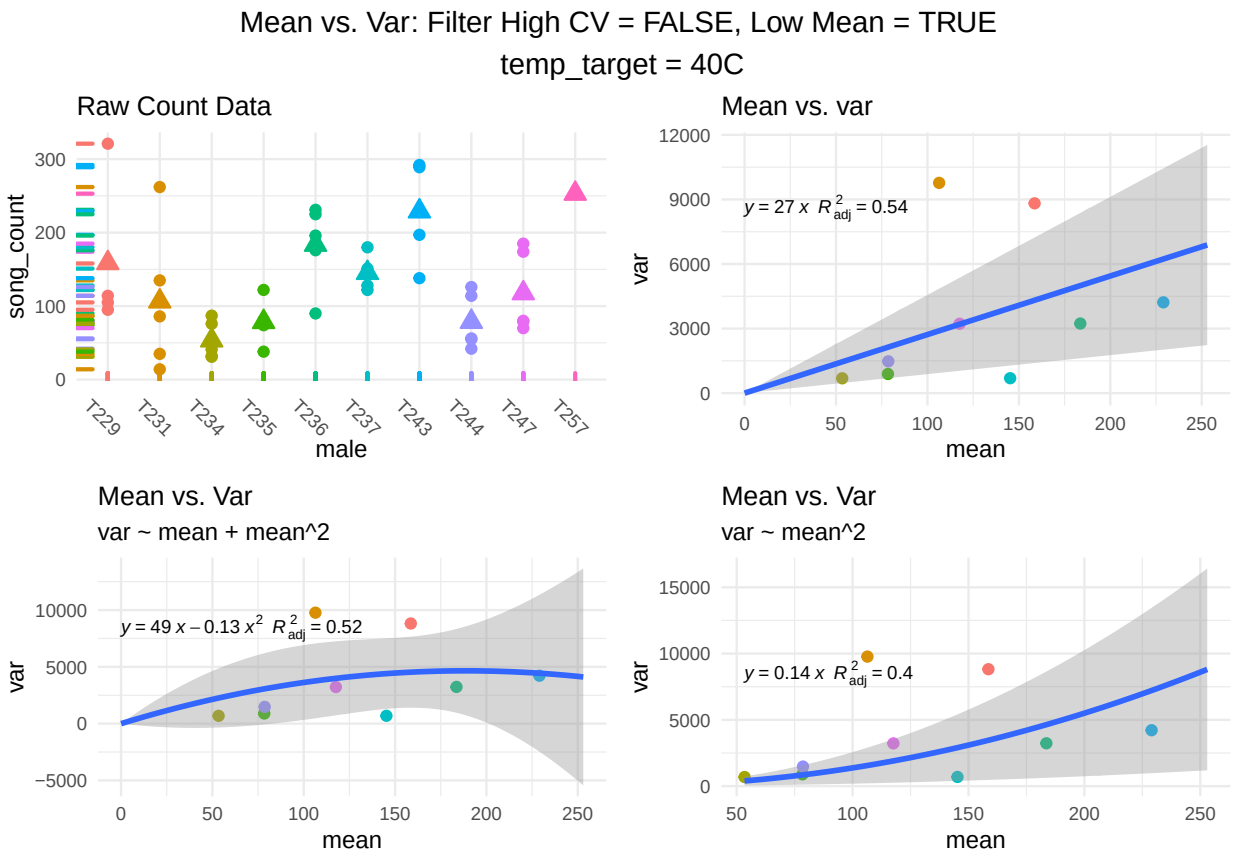
```
g1 <- ggplot(data_tmp) +
    aes(x=song_count,
        color = male,
        fill = male) +
    geom_histogram() +
    xlim(0, NA) +
    labs(title = "Raw Count Data")

formula = y ~ -1 + x
g2 <- ggplot(song_stats_tmp) +
    aes(x=mean, y = var) +
    geom_point(aes(color = male)) +
    geom_smooth(method='lm', formula = formula, fullrange = TRUE) +
    ##stat_cor(label.y.npc = 0.95) +
    stat_regline_equation(label.y.npc = 0.9, aes(label =  paste(..eq.label.., ..adj.rr.label..,
    theme(legend.position = "none") +
    xlim(0, NA) +
    labs(title = "Mean vs. var")

fit_g2 <- lm(var ~ -1 + mean, data = song_stats_tmp)

g3 <- ggplot(song_stats_tmp) +
    aes(x=mean, y = var) +
    geom_point(aes(color = male)) +
    geom_smooth(method='lm', formula= formula, fullrange = TRUE) +
    stat_regline_equation(label.y.npc = 0.9, aes(label =  paste(..eq.label.., ..adj.rr.label..,
    theme(legend.position = "none") +
    labs(title = "Mean vs. Var",
         subtitle = "log(var) ~ log(mean)") +
    scale_x_log10() +
    scale_y_log10()

formula <- y ~ -1 + x + I(x^2)
g4 <- ggplot(song_stats_tmp) +
    aes(x=mean, y = var) +
    geom_point(aes(color = male)) +
    geom_smooth(method='lm', formula = formula, fullrange = TRUE) +
    stat_regline_equation(label.y.npc = 0.9, aes(label =  paste(..eq.label.., ..adj.rr.label..,
    theme(legend.position = "none") +
    xlim(0, NA) +
    labs(title = "Mean vs. Var",
         subtitle = "var ~ mean + mean^2")
                                #    scale_x_log10() +
                                #    scale_y_log10()
fit_g4 <- lm(var ~ -1 + mean +  I(mean^2), data = song_stats_tmp)

formula <- y ~ -1 + I(x^2)
g5 <- ggplot(song_stats_tmp) +
    aes(x=mean, y = var) +
    geom_point(aes(color = male)) +
    geom_smooth(method='lm', formula = formula, fullrange = TRUE) +
    stat_regline_equation(label.y.npc = 0.9,
                          aes(label = paste(
```

```
                                        ..eq.label..,
                                        ..adj.rr.label.., sep = "~~")),
                                    formula = formula, size = 2.5) +
            theme(legend.position = "none") +
            labs(title = "Mean vs. Var",
                 subtitle = "var ~ mean^2")

        fit_g5 <- lm(var ~ -1 + I(mean^2), data = song_stats_tmp)
        #ifelse(length(dev.list()) < 4, dev.new(), dev.next())

        ga <- grid.arrange(g0, g2, g4, g5,
                     ncol=2,
                     top=textGrob(
                         paste0("Mean vs. Var: Filter High CV = ", filter_high_cv,
                                ", Low Mean = ", filter_low_mean, "\n temp_target = 40C"),
                         gp=gpar(fontsize = 11))
                     )
        ##cat("\n\n\\pagebreak\n")
        ##print("<P style='page-break-before: always'>")    #forced new-page happens here.

        print(paste0("Mean vs Var; filter high cv: ", filter_high_cv, "; filter low mean: ", filter_low
        print(summary(fit_g2))
        print(summary(fit_g4))
        print(summary(fit_g5))

ga

        dev.print(device = pdf, file = file.path(output_dir, paste0("mean.vs.var_filter.high-", filter_l

    }
}
```



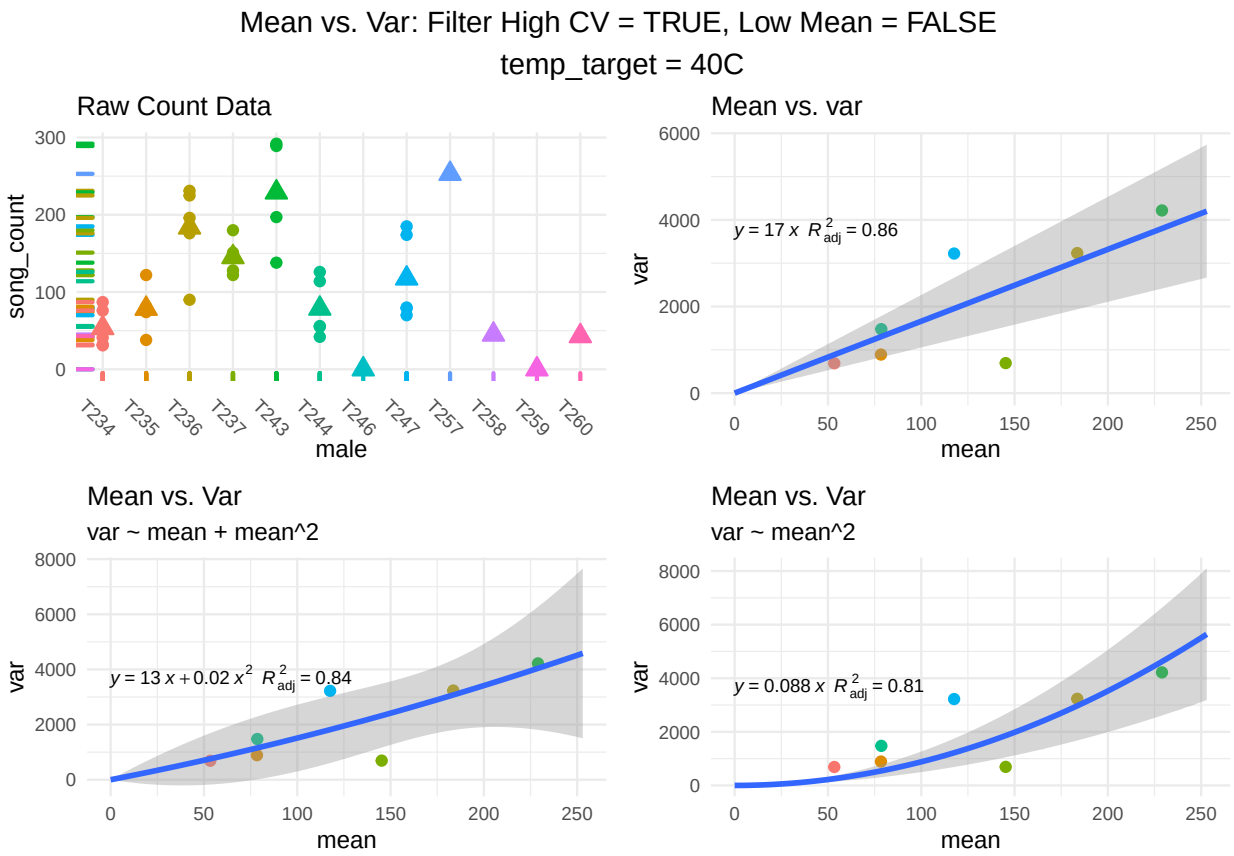

```
## [1] "Mean vs Var; filter high cv: FALSE; filter low mean: FALSE"
##
## Call:
## lm(formula = var ~ -1 + mean, data = song_stats_tmp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3259.7 -1633.0  -713.5     7.1  6874.9
##
## Coefficients:
##      Estimate Std. Error t value Pr(>|t|)
## mean    27.21       7.52   3.619  0.00558 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3125 on 9 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.5927, Adjusted R-squared:  0.5474
## F-statistic:  13.1 on 1 and 9 DF,  p-value: 0.005583
##
##
## Call:
## lm(formula = var ~ -1 + mean + I(mean^2), data = song_stats_tmp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3711.3 -1585.6 -1090.7  -122.8  6004.4
```

```
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## mean       49.2854    26.9567   1.828    0.105
## I(mean^2)  -0.1306     0.1529  -0.854    0.418
##
## Residual standard error: 3174 on 8 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.6267, Adjusted R-squared:  0.5334
## F-statistic: 6.715 on 2 and 8 DF,  p-value: 0.01942
##
##
## Call:
## lm(formula = var ~ -1 + I(mean^2), data = song_stats_tmp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2995.9 -1048.2   170.4  1147.3  8212.9
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## I(mean^2)  0.13757    0.04863   2.829   0.0198 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3563 on 9 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.4707, Adjusted R-squared:  0.4119
## F-statistic: 8.004 on 1 and 9 DF,  p-value: 0.01975
```



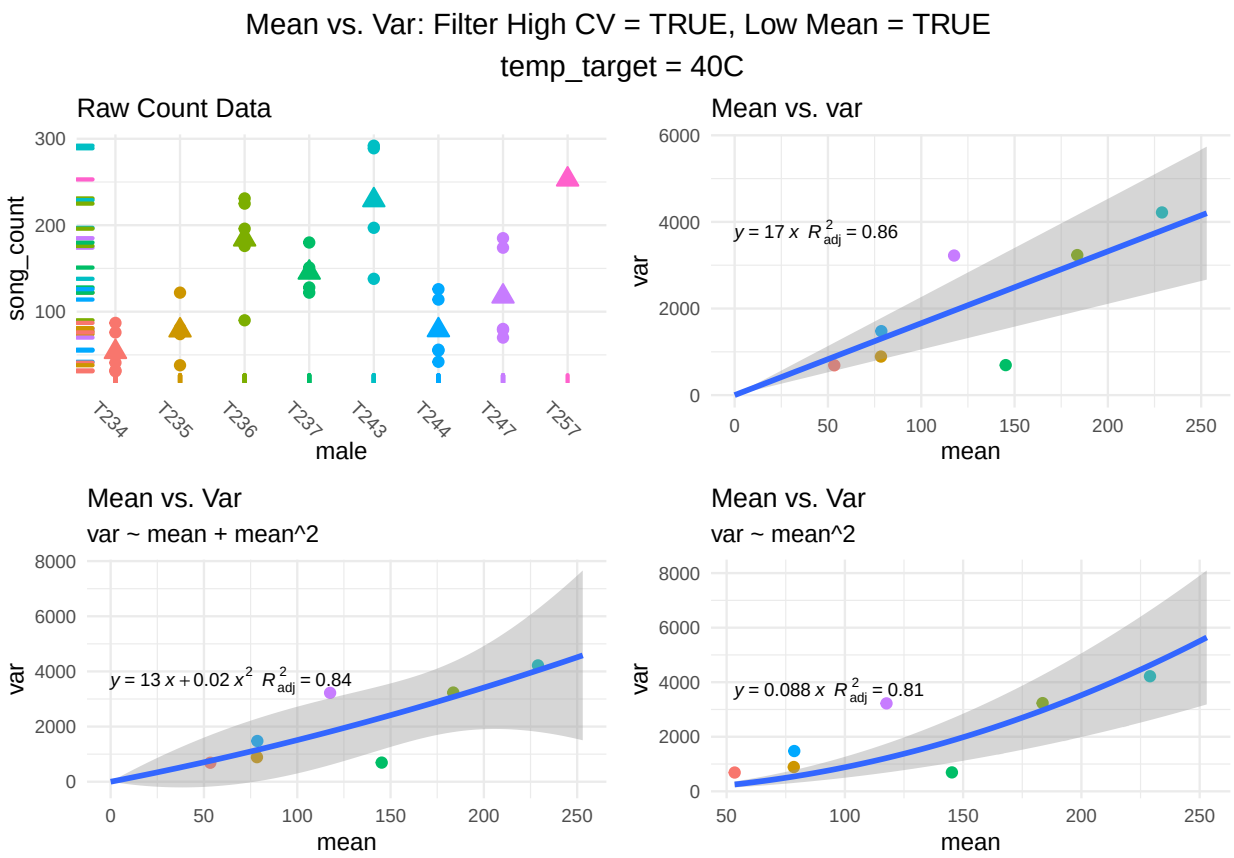

```
## [1] "Mean vs Var; filter high cv: FALSE; filter low mean: TRUE"
##
## Call:
## lm(formula = var ~ -1 + mean, data = song_stats_tmp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3259.8 -1763.0  -764.9    23.0  6874.8
##
## Coefficients:
##      Estimate Std. Error t value Pr(>|t|)
## mean   27.213      7.976   3.412   0.0092 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3315 on 8 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.5927, Adjusted R-squared:  0.5418
## F-statistic: 11.64 on 1 and 8 DF,  p-value: 0.009198
##
##
## Call:
## lm(formula = var ~ -1 + mean + I(mean^2), data = song_stats_tmp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3711.7 -1591.1 -1414.4  -220.3  6003.8
```

```
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## mean       49.2998    28.8216   1.711    0.131
## I(mean^2)  -0.1306     0.1635  -0.799    0.451
##
## Residual standard error: 3392 on 7 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.6267, Adjusted R-squared:  0.5201
## F-statistic: 5.876 on 2 and 7 DF,  p-value: 0.03178
##
##
## Call:
## lm(formula = var ~ -1 + I(mean^2), data = song_stats_tmp)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -2996  -1404    296   1321   8213
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## I(mean^2)  0.13757    0.05158   2.667   0.0285 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3779 on 8 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.4707, Adjusted R-squared:  0.4045
## F-statistic: 7.114 on 1 and 8 DF,  p-value: 0.02848
```




```
## [1] "Mean vs Var; filter high cv: TRUE; filter low mean: FALSE"
##
## Call:
## lm(formula = var ~ -1 + mean, data = song_stats_tmp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1718.5  -304.7   171.9   301.0  1271.0
##
## Coefficients:
##      Estimate Std. Error t value Pr(>|t|)
## mean   16.601      2.477   6.703 0.000535 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 914.2 on 6 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.8822, Adjusted R-squared:  0.8626
## F-statistic: 44.93 on 1 and 6 DF,  p-value: 0.0005351
##
##
## Call:
## lm(formula = var ~ -1 + mean + I(mean^2), data = song_stats_tmp)
##
## Residuals:
##        1        2        3        4        5        6        8
##   -69.43  -260.67   158.60 -1630.15   179.33   322.59  1406.31
```

```
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## mean      13.14106    9.35311   1.405    0.219
## I(mean^2)  0.01964    0.05087   0.386    0.715
##
## Residual standard error: 986.8 on 5 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.8856, Adjusted R-squared:  0.8399
## F-statistic: 19.36 on 2 and 5 DF,  p-value: 0.004426
##
##
## Call:
## lm(formula = var ~ -1 + I(mean^2), data = song_stats_tmp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1166.28   -70.03   348.64   684.69  2004.57
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## I(mean^2)  0.08812    0.01568   5.622  0.00135 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1064 on 6 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.8404, Adjusted R-squared:  0.8139
## F-statistic: 31.61 on 1 and 6 DF,  p-value: 0.001353
```




```
## [1] "Mean vs Var; filter high cv: TRUE; filter low mean: TRUE"
##
## Call:
## lm(formula = var ~ -1 + mean, data = song_stats_tmp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1718.5  -304.7   171.9   301.0  1271.0
##
## Coefficients:
##      Estimate Std. Error t value Pr(>|t|)
## mean   16.601      2.477   6.703 0.000535 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 914.2 on 6 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.8822, Adjusted R-squared:  0.8626
## F-statistic: 44.93 on 1 and 6 DF,  p-value: 0.0005351
##
##
## Call:
## lm(formula = var ~ -1 + mean + I(mean^2), data = song_stats_tmp)
##
## Residuals:
##        1        2        3        4        5        6        7
##   -69.43  -260.67   158.60 -1630.15   179.33   322.59  1406.31
```

```
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## mean      13.14106    9.35311   1.405    0.219
## I(mean^2)  0.01964    0.05087   0.386    0.715
##
## Residual standard error: 986.8 on 5 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.8856, Adjusted R-squared:  0.8399
## F-statistic: 19.36 on 2 and 5 DF,  p-value: 0.004426
##
##
## Call:
## lm(formula = var ~ -1 + I(mean^2), data = song_stats_tmp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1166.28   -70.03   348.64   684.69  2004.57
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## I(mean^2)  0.08812    0.01568   5.622  0.00135 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1064 on 6 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.8404, Adjusted R-squared:  0.8139
## F-statistic: 31.61 on 1 and 6 DF,  p-value: 0.001353
```

Conclusion

- Excluding males with very high variances indicates that for the remaining males `var ~ mean`, but overdispersed relative to the poisson.
- Using multiple trials at one temp could be an effective way to identify 'reliable' males.

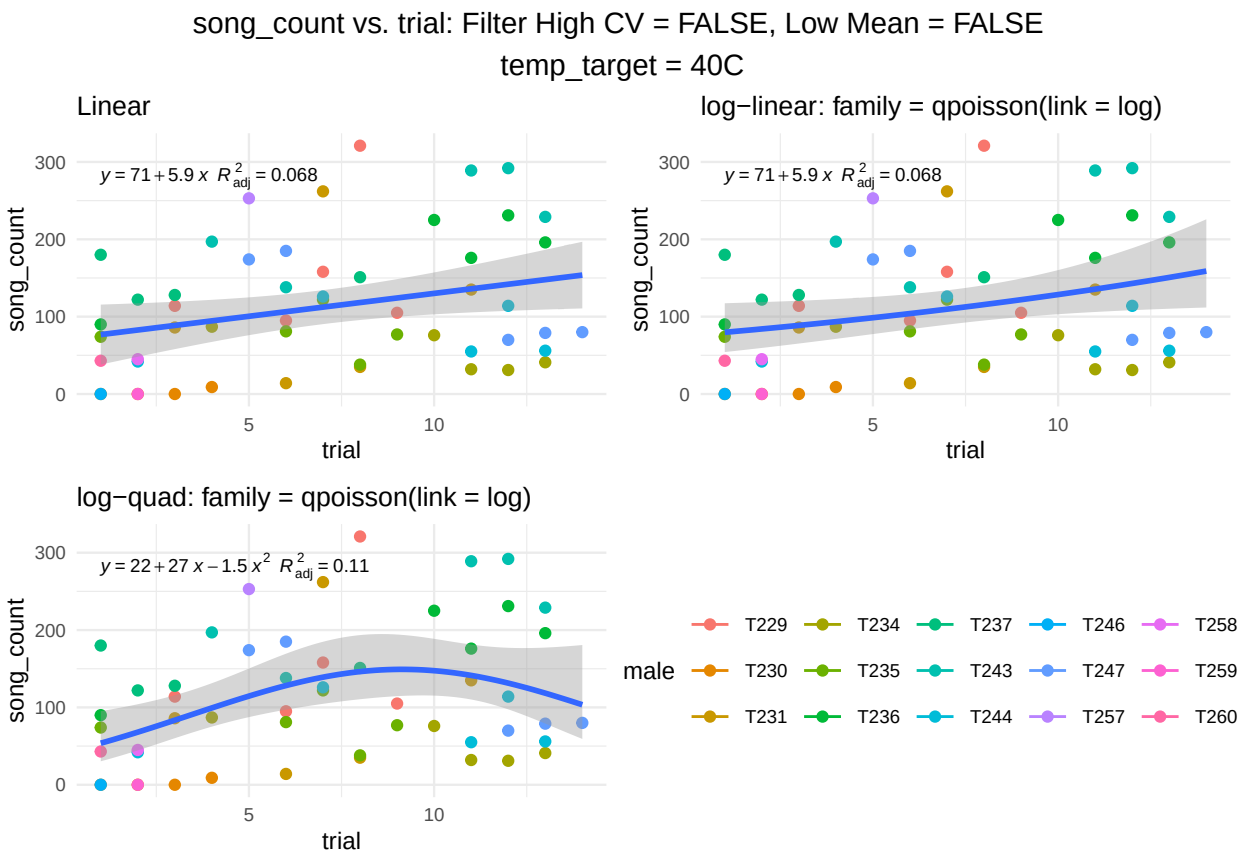
Compare `song_count` vs `trial` at 40C

WARNING: `warning()` are off

```
for(filter_high_cv in c(FALSE, TRUE)){
    for(filter_low_mean in c(FALSE, TRUE)){


        data_tmp <- data_40 %>% group_by(male) %>%
            arrange(trial, .by_group = TRUE) %>%
            mutate(cummean = cummean(song_count)) %>%
            select(male, trial, song_count, cummean)

        if(filter_high_cv){
            data_tmp <- filter(data_tmp, !(male %in% male_high_cv) )
        }
```

```
if(filter_low_mean){
    data_tmp <- filter(data_tmp, !(male %in% male_low_mean) )
}



g1 <- ggplot(data_tmp) +
    aes(x = trial, y = song_count) +
    geom_point(aes(color = male), position = "jitter") +
    geom_line(aes(x = trial, y = cummean, color = male)) +
    theme(legend.position="bottom")

legend <- get_legend(g1)

## Apply a linear regression

formula <- y ~ 1 + x
g0 <- ggplot(data_tmp, aes(x = trial, y = song_count)) +
    geom_point(aes(color = male))


g2 <- g0 +
    geom_smooth(method = 'glm', formula = formula, fullrange = TRUE) +
    stat_regline_equation(label.y.npc = 0.9, aes(label =  paste(..eq.label.., ..adj.rr.label..,
    theme(legend.position = "none") +
    xlim(1, NA) +
    labs(title = "Linear")
##g2

fit_g2 <- glm( song_count ~ 1 + trial, data = data_tmp)

g3 <- g0 + geom_smooth(method = 'glm',
                       formula= formula,
                       ## Can't use link = identity because it leads to negative expectations
                       method.args = list(family = quasipoisson(link = "log"))) +
    stat_regline_equation(label.y.npc = 0.9, aes(label =  paste(..eq.label.., ..adj.rr.label..,
    theme(legend.position = "none") +
    xlim(1, NA) +
    labs(title = "log-linear: family = qpoisson(link = log)")
##g3

fit_g3 <- glm( song_count ~ 1 + trial, data = data_tmp,
               family = quasipoisson(link = "log"))
formula <- y ~ 1 + x + I(x^2)
g4 <- g0 + geom_smooth(method='glm',
                       formula= formula,
                       ## Can't use link = identity because it leads to negative expectations
                       method.args = list(family = quasipoisson(link = "log"))) +
    stat_regline_equation(label.y.npc = 0.9, aes(label =  paste(..eq.label.., ..adj.rr.label..,
    theme(legend.position = "none") +
    xlim(1, NA) +
    labs(title = "log-quad: family = qpoisson(link = log)")
##g4
```

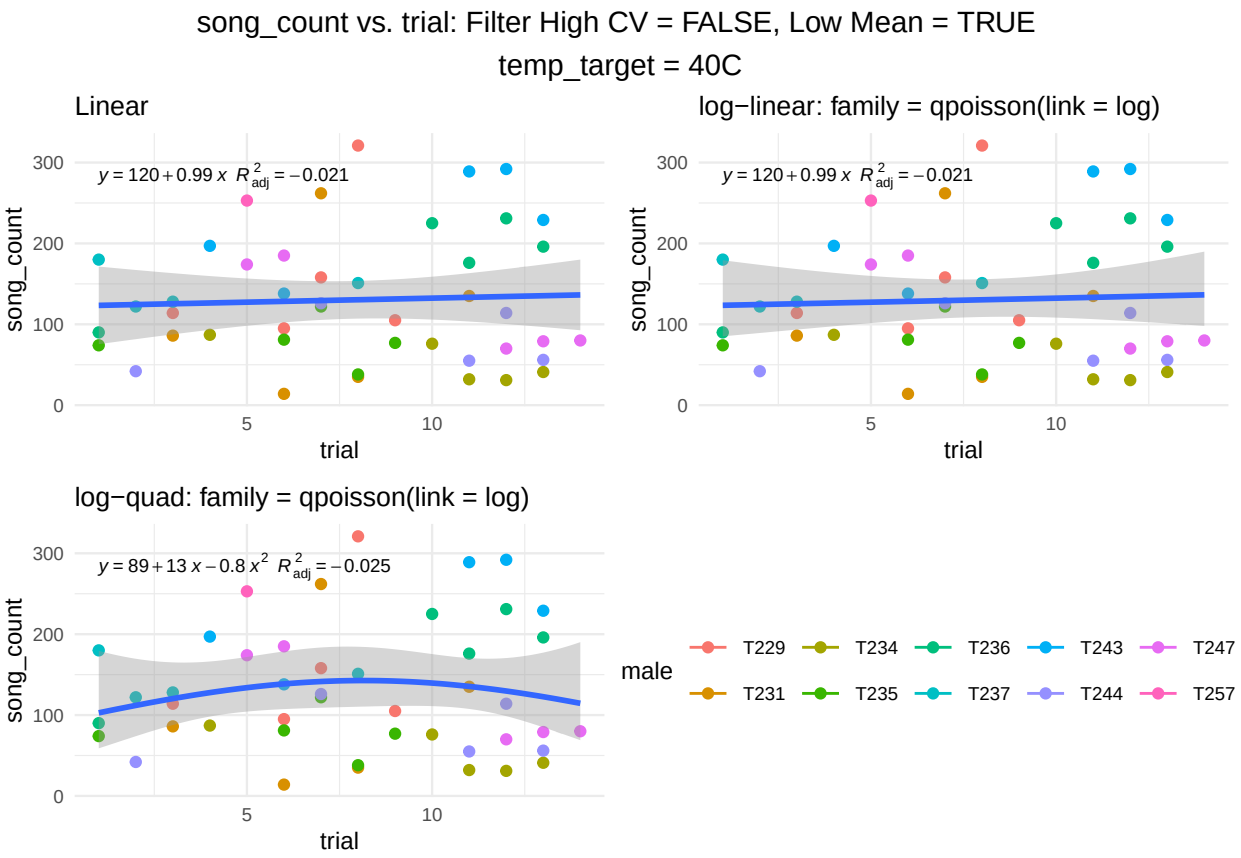
```
        fit_g4 <- glm(song_count ~ 1 + trial + I(trial^2), data = data_tmp,
                      family = quasipoisson(link = "log"))

        ga <- grid.arrange(g2, g3, g4, as_ggplot(legend),
                           ncol=2,
                           top=textGrob(
                               paste0("song_count vs. trial: Filter High CV = ", filter_high_cv, ", Low
                               gp=gpar(fontsize = 11))
                           )
        ##cat("\n\n\\pagebreak\n")
        ##print("<P style='page-break-before: always'>")    #forced new-page happens here.

        print(paste0("song_count vs. trial; filter.high: ", filter_high_cv, "; filter.low.mean: ", filt

        print(summary(fit_g2))
        print(summary(fit_g3))
        print(summary(fit_g4))

        ga

        dev.print(device = pdf, file = file.path(output_dir, paste0("song_count.vs.trial_filter.high-",
    }
}
```




```
## [1] "song_count vs. trial; filter.high: FALSE; filter.low.mean: FALSE"
```

```
##
## Call:
## glm(formula = song_count ~ 1 + trial, data = data_tmp)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -110.908   -76.819    -7.571    45.678   202.761
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   70.902     22.072   3.212  0.00228 **
## trial          5.917      2.708   2.185  0.03353 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 6667.673)
##
##     Null deviance: 371877  on 52  degrees of freedom
## Residual deviance: 340051  on 51  degrees of freedom
## AIC: 621.04
##
## Number of Fisher Scoring iterations: 2
##
##
## Call:
## glm(formula = song_count ~ 1 + trial, family = quasipoisson(link = "log"),
##     data = data_tmp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -13.319   -7.867   -0.683    3.911   15.646
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.32614    0.21825  19.822   <2e-16 ***
## trial        0.05303    0.02448   2.166    0.035 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 59.69275)
##
##     Null deviance: 3611.6  on 52  degrees of freedom
## Residual deviance: 3327.6  on 51  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
##
##
## Call:
## glm(formula = song_count ~ 1 + trial + I(trial^2), family = quasipoisson(link = "log"),
##     data = data_tmp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

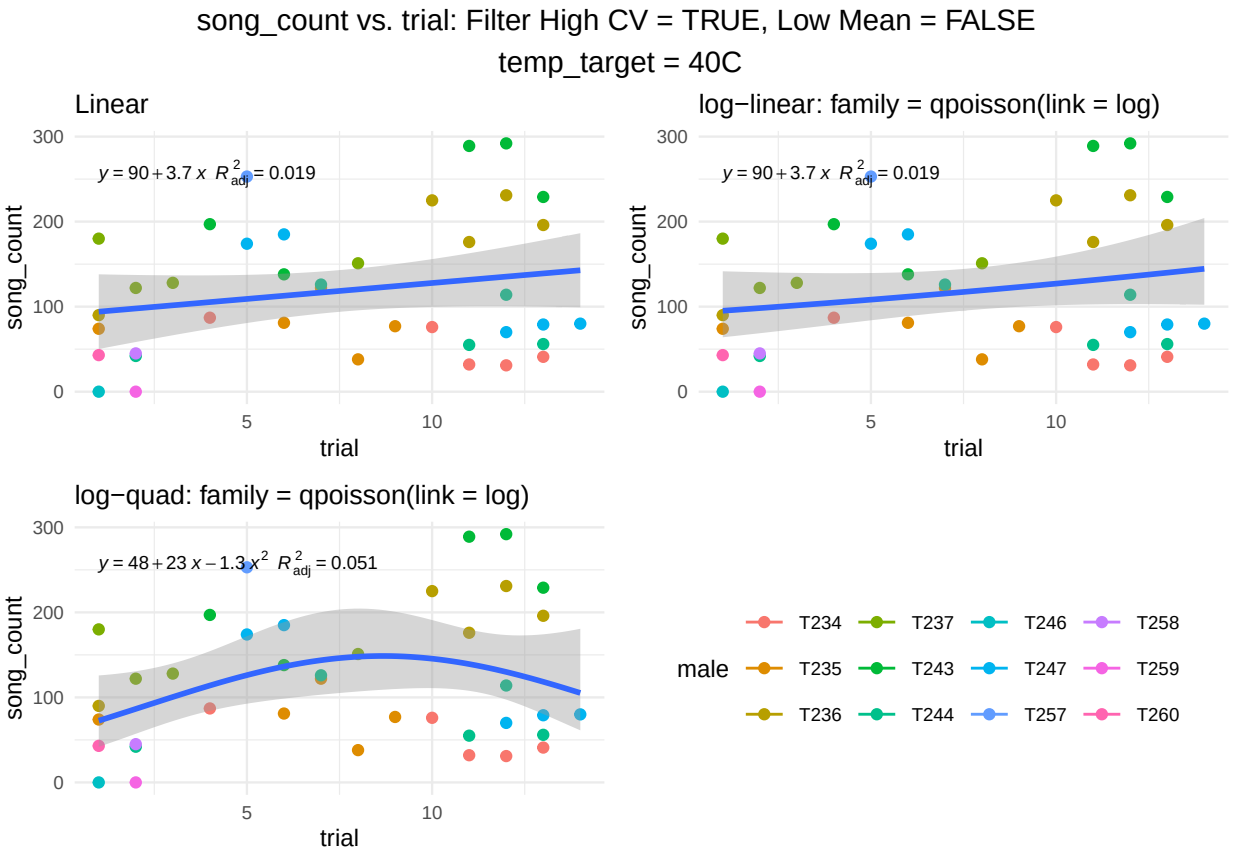
```
## -12.921   -6.519   -1.481    4.698   13.516
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.717425   0.386246   9.624 5.73e-13 ***
## trial        0.282076   0.115036   2.452   0.0177 *
## I(trial^2)  -0.015445   0.007511  -2.056   0.0450 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 56.64162)
##
##     Null deviance: 3611.6  on 52  degrees of freedom
## Residual deviance: 3078.6  on 50  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```




```
## [1] "song_count vs. trial; filter.high: FALSE; filter.low.mean: TRUE"
##
## Call:
## glm(formula = song_count ~ 1 + trial, data = data_tmp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -114.37   -56.33   -11.38    56.61   190.65
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 122.3982    27.2313   4.495 5.19e-05 ***
## trial         0.9946     3.0966   0.321     0.75
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 6363.635)
##
##     Null deviance: 274293  on 44  degrees of freedom
## Residual deviance: 273636  on 43  degrees of freedom
## AIC: 525.78
##
## Number of Fisher Scoring iterations: 2
##
##
## Call:
## glm(formula = song_count ~ 1 + trial, family = quasipoisson(link = "log"),
##     data = data_tmp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -12.908   -5.259   -1.034    4.690   14.051
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.808539   0.211229  22.765   <2e-16 ***
## trial       0.007662   0.023762   0.322    0.749
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 48.46583)
##
##     Null deviance: 2096.1  on 44  degrees of freedom
## Residual deviance: 2091.1  on 43  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
##
##
## Call:
## glm(formula = song_count ~ 1 + trial + I(trial^2), family = quasipoisson(link = "log"),
##     data = data_tmp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -13.596   -5.750   -1.325    3.756   12.806
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.534157   0.375166  12.086 2.93e-15 ***
## trial        0.104658   0.108919   0.961    0.342
## I(trial^2)  -0.006422   0.007001  -0.917    0.364
```

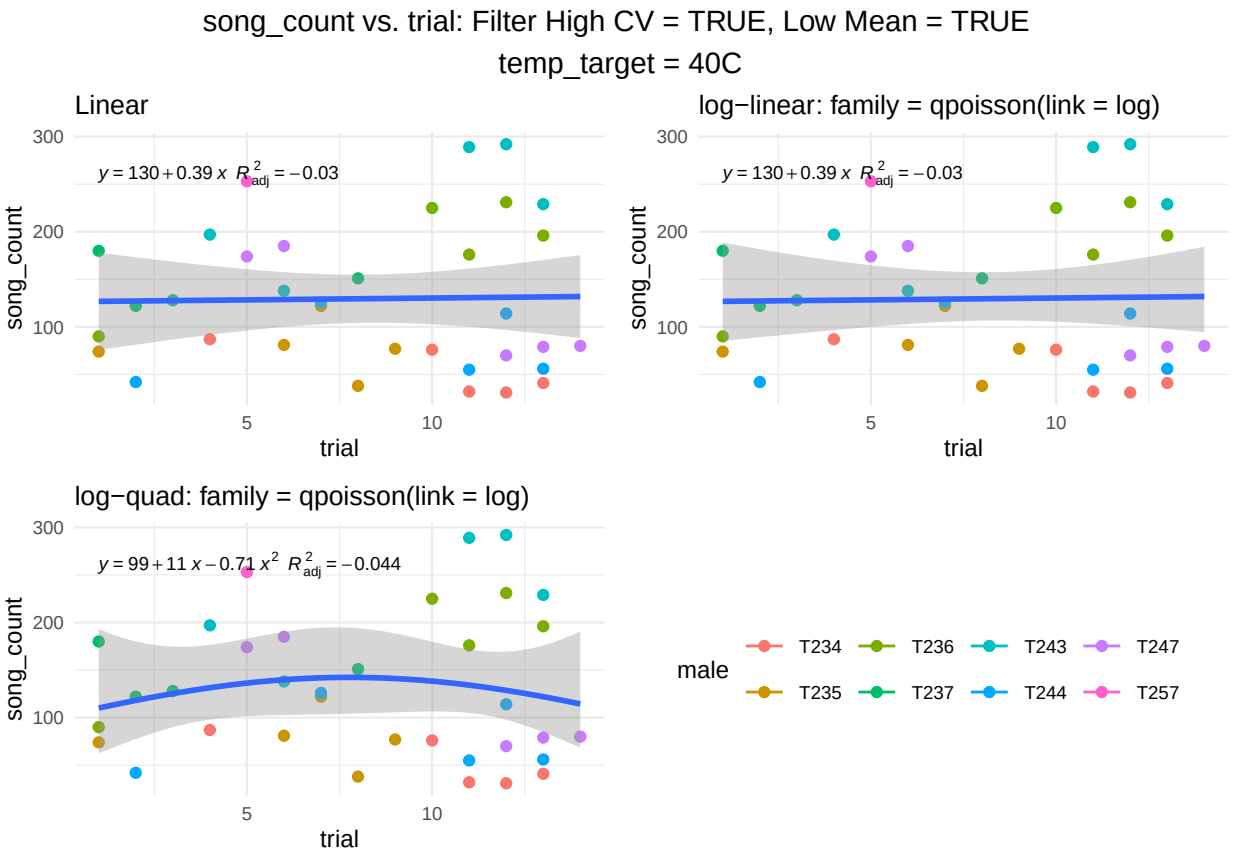
```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 48.09502)
##
##     Null deviance: 2096.1  on 44  degrees of freedom
## Residual deviance: 2049.8  on 42  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```




```
## [1] "song_count vs. trial; filter.high: TRUE; filter.low.mean: FALSE"
##
## Call:
## glm(formula = song_count ~ 1 + trial, data = data_tmp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -104.30   -61.42   -18.34    60.94   157.44
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   90.353     24.817   3.641 0.000826 ***
## trial          3.746      2.843   1.318 0.195741
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for gaussian family taken to be 6107.371)
##
##     Null deviance: 236575  on 38  degrees of freedom
## Residual deviance: 225973  on 37  degrees of freedom
## AIC: 454.6
##
## Number of Fisher Scoring iterations: 2
##
##
## Call:
## glm(formula = song_count ~ 1 + trial, family = quasipoisson(link = "log"),
##     data = data_tmp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -14.020   -6.115   -1.792    5.135   11.864
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.52356    0.22382  20.210   <2e-16 ***
## trial        0.03213    0.02427   1.324    0.194
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 50.75283)
##
##     Null deviance: 2119.6  on 38  degrees of freedom
## Residual deviance: 2029.3  on 37  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
##
##
## Call:
## glm(formula = song_count ~ 1 + trial + I(trial^2), family = quasipoisson(link = "log"),
##     data = data_tmp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -13.151   -6.059   -1.510    3.988   12.197
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.086186   0.373482  10.941 5.33e-13 ***
## trial        0.211196   0.117265   1.801   0.0801 .
## I(trial^2)  -0.012178   0.007741  -1.573   0.1245
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 49.90484)
##
##     Null deviance: 2119.6  on 38  degrees of freedom
## Residual deviance: 1903.7  on 36  degrees of freedom
```

```
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```




```
## [1] "song_count vs. trial; filter.high: TRUE; filter.low.mean: TRUE"
##
## Call:
## glm(formula = song_count ~ 1 + trial, data = data_tmp)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -100.093   -53.622    -7.151    54.708   160.907
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 126.4330    28.9079   4.374 0.000115 ***
## trial         0.3884     3.1424   0.124 0.902393
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for gaussian family taken to be 5847.42)
##
##     Null deviance: 193054  on 34  degrees of freedom
## Residual deviance: 192965  on 33  degrees of freedom
## AIC: 406.85
##
```

```
## Number of Fisher Scoring iterations: 2
##
##
## Call:
## glm(formula = song_count ~ 1 + trial, family = quasipoisson(link = "log"),
##     data = data_tmp)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -10.5257   -5.1277   -0.6343    4.5435   12.0779
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.839887   0.224035  21.603   <2e-16 ***
## trial       0.003003   0.024245   0.124    0.902
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 44.90935)
##
##     Null deviance: 1486.5  on 34  degrees of freedom
## Residual deviance: 1485.8  on 33  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
##
##
## Call:
## glm(formula = song_count ~ 1 + trial + I(trial^2), family = quasipoisson(link = "log"),
##     data = data_tmp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -10.612   -5.734   -1.356    4.480   12.327
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.619510   0.381334  12.114 1.71e-13 ***
## trial        0.087189   0.116270   0.750    0.459
## I(trial^2)  -0.005615   0.007547  -0.744    0.462
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 45.64702)
##
##     Null deviance: 1486.5  on 34  degrees of freedom
## Residual deviance: 1460.2  on 32  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

Conclusion

When filtering out low mean (which have few trials and, thus, bias the analysis towards a small intercept) results in no evidence that `song_count` increases with `trial`

End

```
knitr::knit_exit()
```