Annotated Figure List

Nate Olson

2017-04-11

Objective

Help metagenomic method (bioinformatic pipelines) users and developers better understand how their method is performing and what features they should or should not have confidence in.

These can include;

- types of features that are not well-behaved relative to our expectations in terms of quantitative and qualitative accuracy,

- defining a method limit of detection (based on number of PCR replicates with and without observed counts)
  - Approach - Multinomial sampling based approach where proportions are based on pooled replicates

Study Goal

Evaluate bioinformatic pipeline and feature performance. Need to evaluate in a manner that does not confound experimental artifacts with pipeline/ feature artifacts. Experimental artifacts include, low and no observed counts due to sampling and titrations not mixed according to expectations.

Open Questions

- Dealing with potentially uninformative features - qualitative analysis
- Bias and variance metrics
- What to do with NTC features (includes Escherichia)

Sample design

Titration Validation

ERCC spike-in qPCR and bacterial DNA qPCR See `mixing_and_validating_titrations.pdf` in artifacts

Bacterial DNA qPCR quantification …

Using mixtures - use E. coli as example

Include equations, relationship between expected

Diagram with scatter plot of observed counts and titrations, then observed and expected with colored titrations, shapes for pcr reps.
Residual lines???

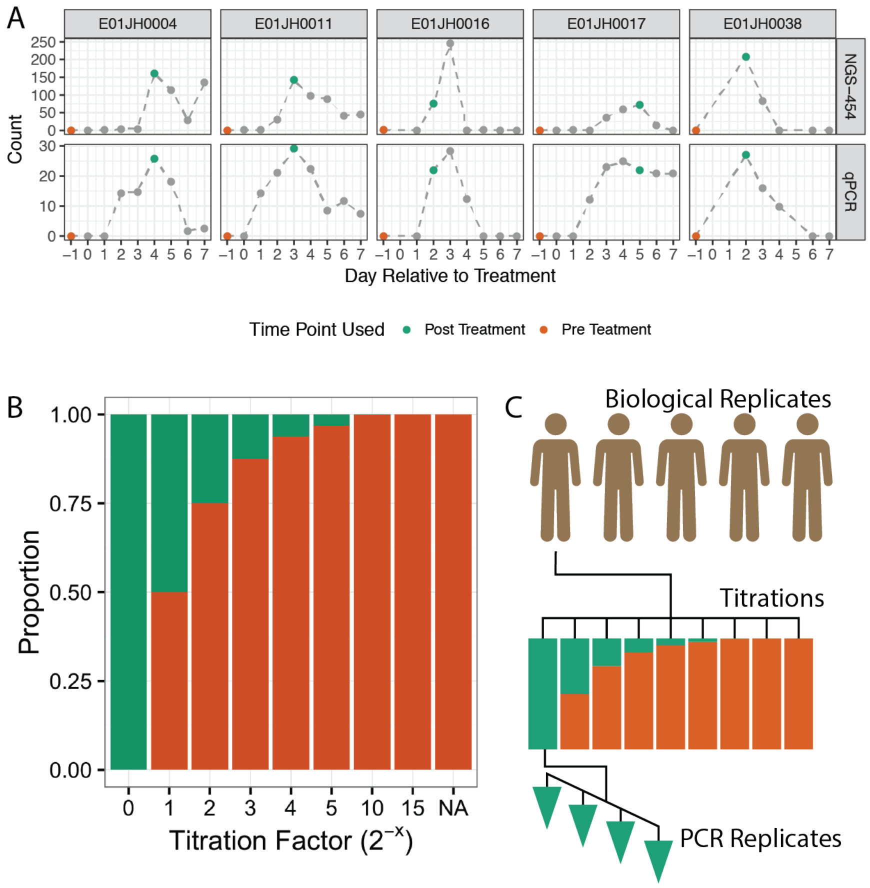


Figure 1: Sample selection and experimental design for two-sample titration 16S rRNA metagenomic sequencing assessment dataset. A) Pre- and post-treatment samples from five participants in a vaccine trial (Harro et al. 2011) were selected based on Escherichia coli abundance measured using qPCR and 454 16S rRNA metagenomics sequencing (454-NGS), data from Pop et al. 2016. Pre- and post-treatment samples are indicated with orange and green data points. Grey indicates other samples from the vaccine trial time series. B) The pre-treatment samples were titrated into post-treatment samples following a log_2 dilution series. The NA titration factor represents the unmixed pre-treatment sample. C) The five vaccine trial participants are biological replicates and independent sets of two-sample titrations were mixed for each. The result was a total of 45 samples, 7 titrations + 2 unmixed samples times 5 biological replicates. Four replicate PCRs were performed for each of the 45 samples resulting in 190 PCRs.

Seq QA

See `seq-qa.pdf` in artifacts.

Pipeline Characterization

see `pipeline_characterization.pdf` in artifacts

Feature categories

Objective of this categorization is to identify a set of features we would expect to be well behaved based on presence/ absence data alone. Informative features will be used for quantitative analysis and uninformative features qualitative analysis.
Uninformative features are potential indicators of an artifact of feature inference (clustering) are features where non-detected PCR replicates and samples cannot be explained by random sampling alone given the observed count values.

Excluding features observed in no template controls, unable to differentiate between count values due to reagent contaminants or thoes from the biological samples.

Informative Feature Categories

- Full - present in all samples and at least 3 of 4 PCR replicates
- Pre - present in all samples and all replicates, excluding post
- Post - present in all samples and all replicates, excluding pre

Uninformative Feature Categories

- Mix - not present in any unmixed pre- or post-treatment PCR replicates and at least 4 PCR replicates for one of the titrations

Potentially Informative Feature Categories

- None - Features not assigned to any of the other categories

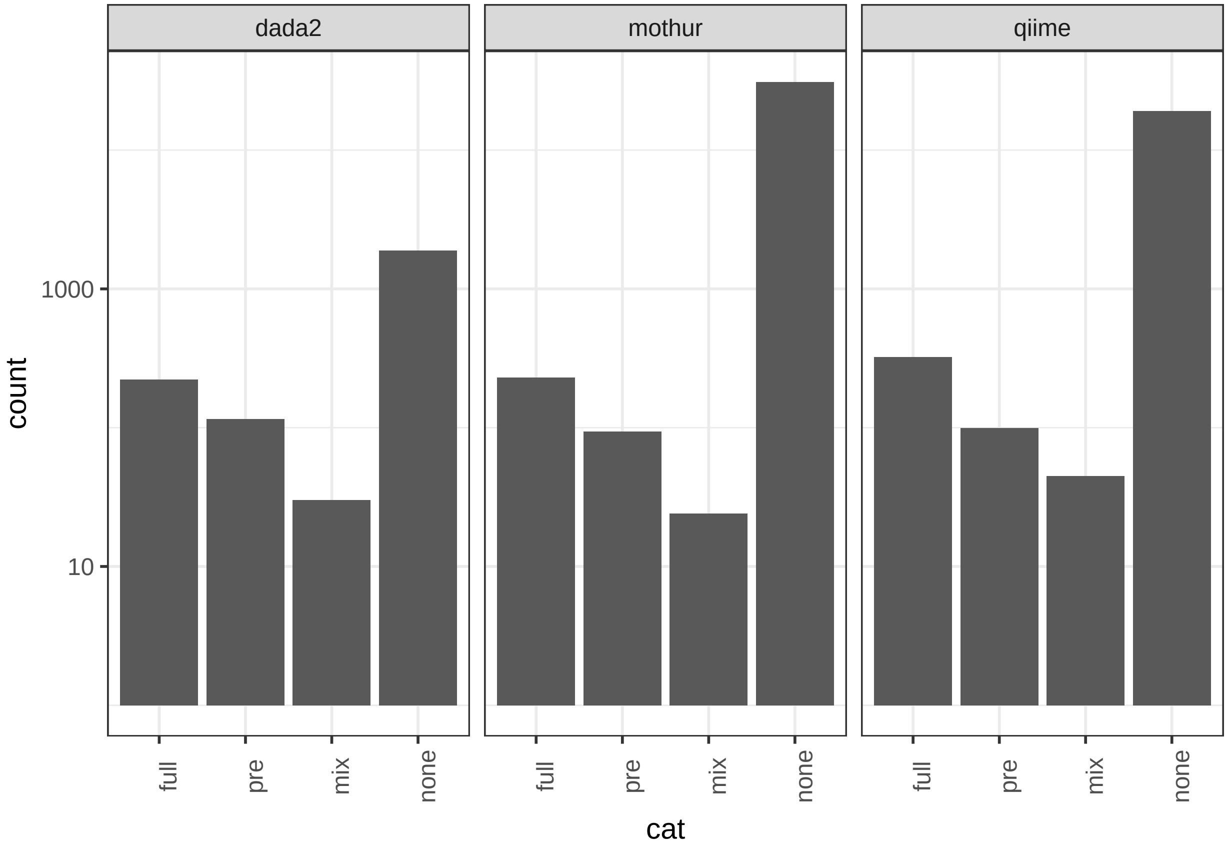
Notes
Most of the features are not assigned to a category. Of the unassigned features, #### are present in only one PCR replicate of one sample for a biological replicate, see `2017-04-10-Feature-Cat-Informative-Uninformative.Rmd` for breakdown of uncategorized features.
Most of the uncategorized features are present in at least one of the no template control samples.

Key Points

- Will use the informative features for quantitative analysis.
- Can look into the uncategorized features for additional features to include in the quantitative analysis if deemed necessary.
- Mix specific features that cannot be explained by random sampling are likely artifacts of feature inference.

```r
ntc_features <- readRDS("data/ntc_features.rds")
feature_cat <- readRDS("data/feature_categories_df.rds") %>%
      anti_join(ntc_features) %>% ungroup() %>%
      filter(biosample_id != "NTC", !is.na(biosample_id))
```

```
feature_cat %>% filter(cat != "cat_null") %>%
    mutate(cat = gsub(pattern = "cat_", replacement = "", cat),
           cat = if_else(cat == "near_full", "full",cat),
           cat = fct_relevel(cat, c("full","post","pre","mix","none"))) %>%
      ggplot() + geom_bar(aes(x = cat)) +
      facet_grid(.~pipe, scales = "free_y") +
      theme_bw() + theme(axis.text.x = element_text(angle = 90)) +
    scale_y_log10()
```



Non-informative features - relating biological and experimental

Dropout features - only pre or only post and present in all 4 PCR replicates Mix only features present in multiple titrations

Feature dropout as artifact of:

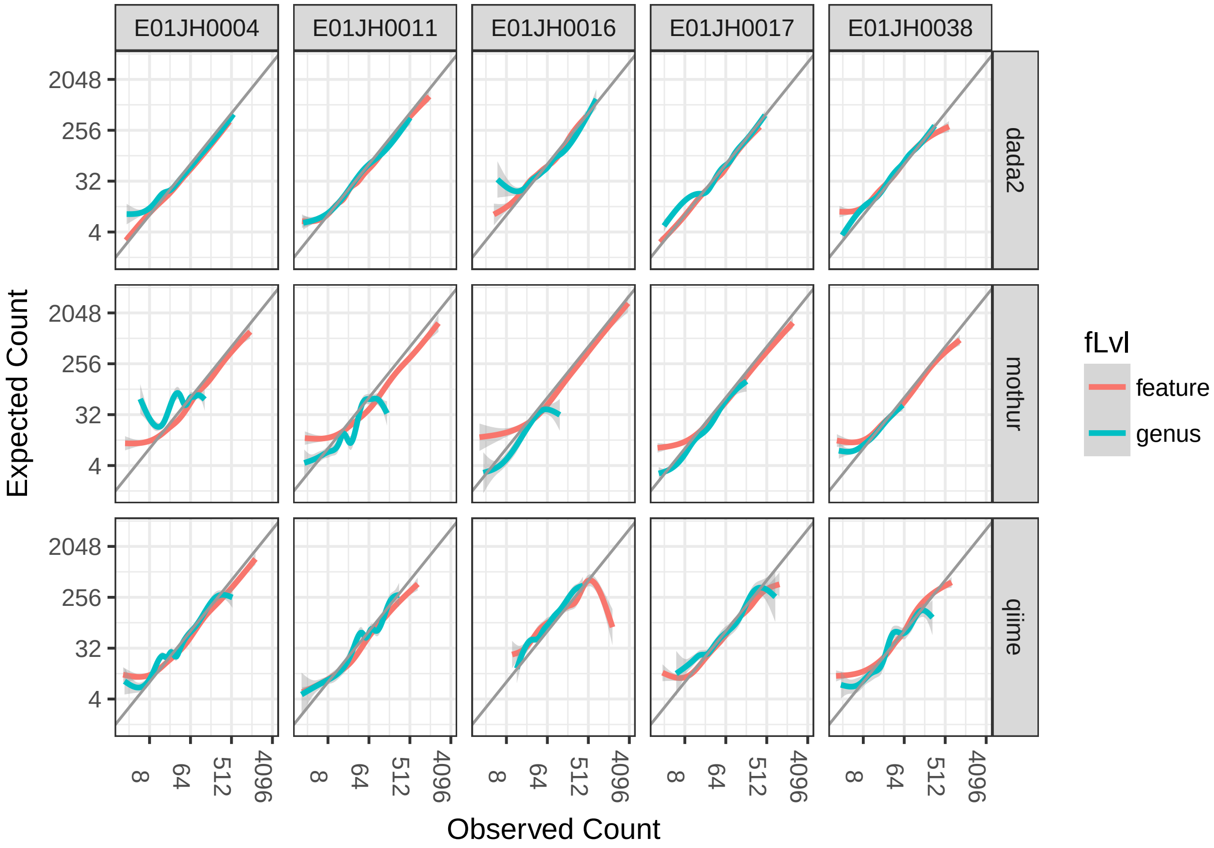
- clustering - distance to neighboring cluster center
- phylogenetic signal - proxy for sequence context and primer binding
- Biosample effect - same feature dropout for multiple samples

Biosample and pipeline general evaluation - bias and variance

```
count_exp_df <- readRDS("data/expected_count_values_feature_df.rds")
feature_cat <- readRDS("data/feature_categories_df.rds")
ntc_features <- readRDS("data/ntc_features.rds")
count_exp_df <- left_join(count_exp_df, feature_cat) %>%
      anti_join(ntc_features) %>%
      filter(cat %in% c("cat_full", "cat_pre", "cat_post"), id != "1-F9")

genus_exp_df <- readRDS("data/expected_count_values_genus_df.rds")
genus_cat <- readRDS("data/genus_categories_df.rds")
genus_ntc <- readRDS("data/ntc_genus.rds")
genus_exp_df <- left_join(genus_exp_df, genus_cat) %>%
      anti_join(genus_ntc) %>%
      filter(cat %in% c("cat_full", "cat_pre", "cat_post"), id != "1-F9")
exp_df <- bind_rows(feature = count_exp_df, genus = genus_exp_df, .id = "fLvl")
```

Overall relationship between the observed and expected values by pipeline and biological replicate. Red and teal fitted smoothing function (loess, local polynomial regression) to highlight the relationship between the observed and expected counts, for feature level and genus level count values. Excluding features observed in any no template control. These include some of the *Escherichia* features, may want to figure out a better filtering approach.

```
exp_df %>% ggplot() +
      geom_smooth(aes(x = obs_count + 1, y = exp_count + 1, color = fLvl)) +
      geom_abline(aes(intercept = 0, slope = 1), color = "grey60") +
      facet_grid(pipe~biosample_id)+ theme_bw() +
      labs(y = "Expected Count", x = "Observed Count", fill = "Abundance") +
      scale_y_continuous(trans = "log2") +
      scale_x_continuous(trans = "log2") +
      theme(axis.text.x = element_text(angle = 270))
```

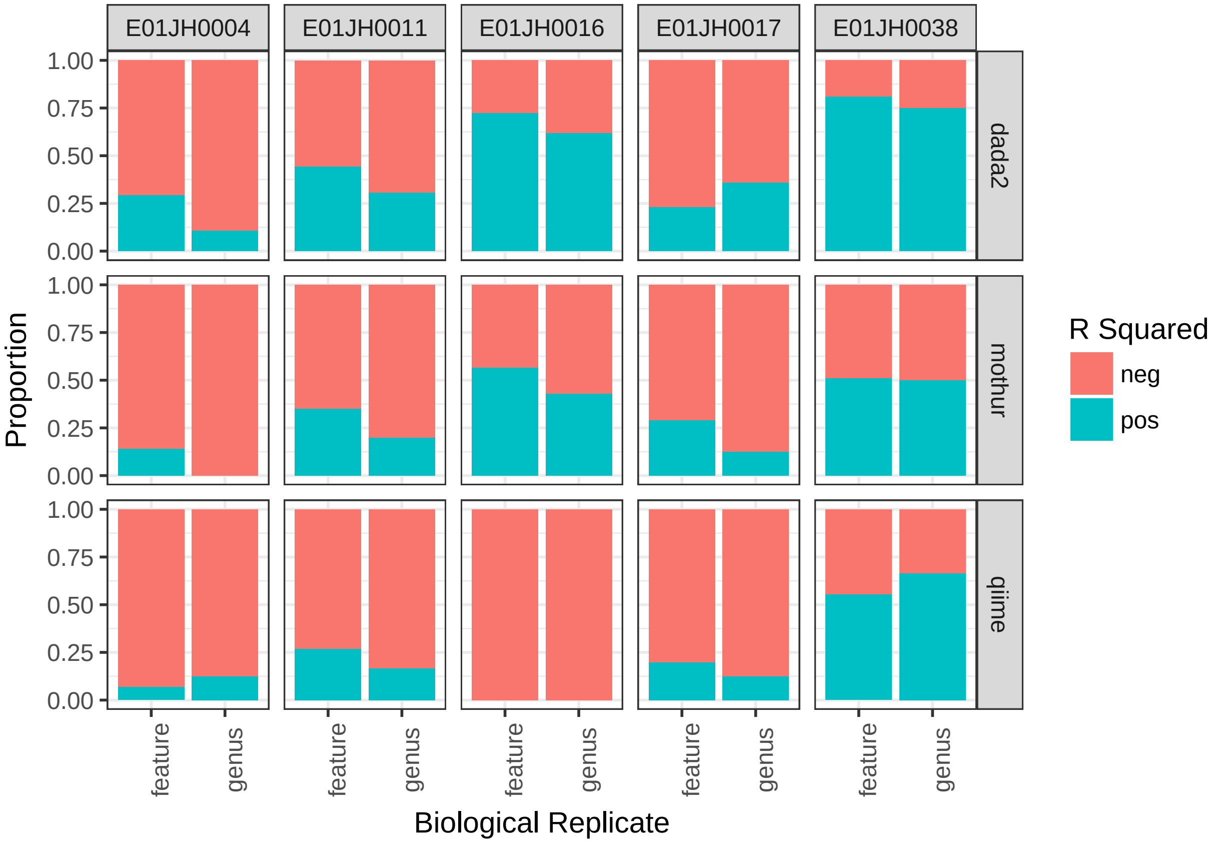


Bias - Biological Replicate

A number of features have negative R^2 values, indicating that the relationship between the observed and expected counts has greater variablitity than the variability in the observed counts alone.
Looking at individual features, the negative R^2 values are associated with features where the observed counts for the unmixed features is higher or lower than the observed counts for the mixed samples. Features with very negative R^2 values are likely artifacts of the feature inference process.

```
feature_r2_df <- count_exp_df %>%
      group_by(pipe, biosample_id, feature_id) %>%
      summarise(ss_total = sum((obs_count - mean(obs_count))^2),
                ss_res = sum(residual^2)) %>%
      mutate(r_squared = 1 - ss_res/ss_total)

genus_r2_df <- genus_exp_df %>%
      group_by(pipe, biosample_id, feature_id) %>%
      summarise(ss_total = sum((obs_count - mean(obs_count))^2),
                ss_res = sum(residual^2)) %>%
      mutate(r_squared = 1 - ss_res/ss_total)
r2_df <- bind_rows(feature = feature_r2_df, genus = genus_r2_df, .id = "fLvl")
```

```
r2_df %>% mutate(r2 = if_else(r_squared > 0, "pos","neg")) %>%
    ggplot() + geom_bar(aes(x = fLvl, fill = r2), position = "fill") +
    facet_grid(pipe~biosample_id) +
    labs(x = "Biological Replicate", y = "Proportion", fill = "R Squared") +
    theme_bw() + theme(axis.text.x = element_text(angle = 90))
```

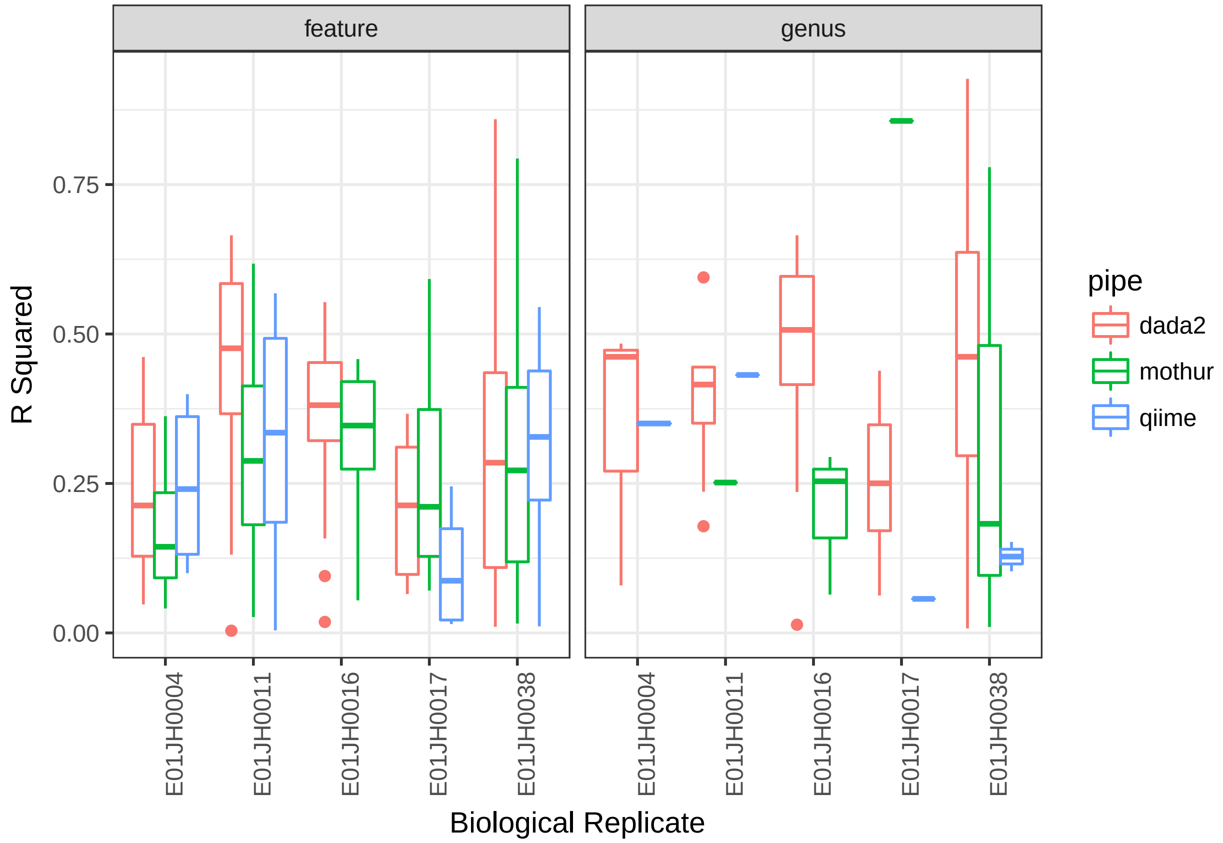


Features with non-negative R^2 values

Key Points

- Does aggregating at features to the genus level improve the quantitative accuracy of the data?
- Can test to see if aggregating the the genus level improves R^2, is this effect just due to higher counts as a result of aggregating counts.

```
r2_df %>% filter(r_squared > 0) %>%
    ggplot() +
    geom_boxplot(aes(x = biosample_id, y = r_squared, color = pipe)) +
    theme_bw() + facet_grid(~fLvl) +
    theme(axis.text.x = element_text(angle = 90)) +
    labs(x = "Biological Replicate", y = "R Squared")
```



```
fit <- lm(r_squared ~ fLvl + biosample_id*pipe, data = r2_df %>% filter(r_squared > 0))
```

Aggregating to the genus level increases the overall R^2 values. Potentially an artifact of the increases counts obtained when aggregating to the genus level. There is also a biological replicate affect which is potentially due to titrations not formulated as expected or due to interactions between sequences (DNA molecules) in the pre- and post-treatment unmixed samples.

```
aov(fit) %>% broom::tidy() %>% knitr::kable()
```

term	df	sumsq	meansq	statistic	p.value
fLvl	1	0.3849260	0.3849260	11.748449	0.0006952
biosample_id	4	0.7915862	0.1978966	6.040064	0.0001102
pipe	2	0.1733999	0.0867000	2.646198	0.0725979
biosample_id:pipe	7	0.3274427	0.0467775	1.427712	0.1936417
Residuals	296	9.6981385	0.0327640	NA	NA

```
aov(fit) %>% TukeyHSD() %>% broom::tidy() %>%
      filter(adj.p.value < 0.05) %>% knitr::kable()
```

term	comparison	estimate	conf.low	conf.high	adj.p.value
fLvl	genus-feature	0.0870223	0.0370571	0.1369874	0.0006952
biosample_id	E01JH0011-E01JH0004	0.1261688	0.0232334	0.2291043	0.0076927
biosample_id	E01JH0016-E01JH0004	0.1195598	0.0126638	0.2264557	0.0196811
biosample_id	E01JH0017-E01JH0011	-0.1299575	-0.2236419	-0.0362732	0.0015927
biosample_id	E01JH0017-E01JH0016	-0.1233485	-0.2213678	-0.0253291	0.0056777
biosample_id:pipe	E01JH0017:dada2-E01JH0011:dada2	-0.2049639	-0.3742074	-0.0357205	0.0039991

term	comparison	estimate	conf.low	conf.high	adj.p.value
biosample_id:pipe	E01JH0017:qiime-E01JH0011:dada2	-0.3215019	-0.6192235	-0.0237803	0.0206718
biosample_id:pipe	E01JH0017:dada2-E01JH0016:dada2	-0.1738287	-0.3409776	-0.0066798	0.0325220

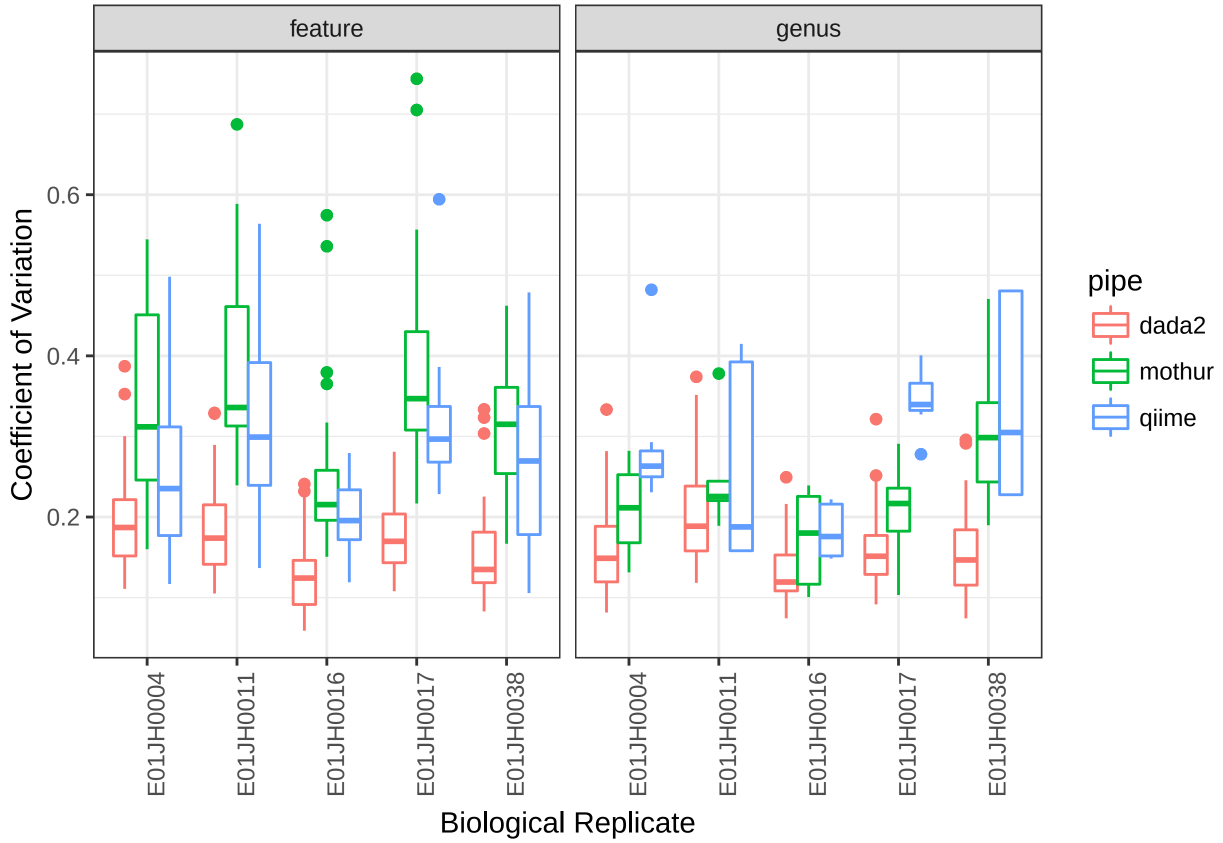
Count Variance - Biosample~Pipeline

Use the feature level coefficient of variation for the replicate counts as the variance metric.

```r
## will want to update to include unmixed counts as well
feature_var_metric <- count_exp_df %>%
    dplyr::rename(count = obs_count) %>%
    group_by(pipe, feature_id, biosample_id, t_fctr) %>%

    summarise(count_cov = sd(count)/mean(count),
              med_count = median(count)) %>%
    group_by(pipe, feature_id, biosample_id) %>%
    mutate(mean_count_cov = mean(count_cov, na.rm = TRUE)) %>%
    ungroup() %>%
    mutate(feature_id = fct_reorder(feature_id, mean_count_cov))

genus_var_metric <- genus_exp_df %>%
    dplyr::rename(count = obs_count) %>%
    group_by(pipe, feature_id, biosample_id, t_fctr) %>%

    summarise(count_cov = sd(count)/mean(count),
              med_count = median(count)) %>%
    group_by(pipe, feature_id, biosample_id) %>%
    mutate(mean_count_cov = mean(count_cov, na.rm = TRUE)) %>%
    ungroup() %>%
    mutate(feature_id = fct_reorder(feature_id, mean_count_cov))

var_metric <- bind_rows(feature = feature_var_metric,
                           genus = genus_var_metric, .id = "fLvl")
```

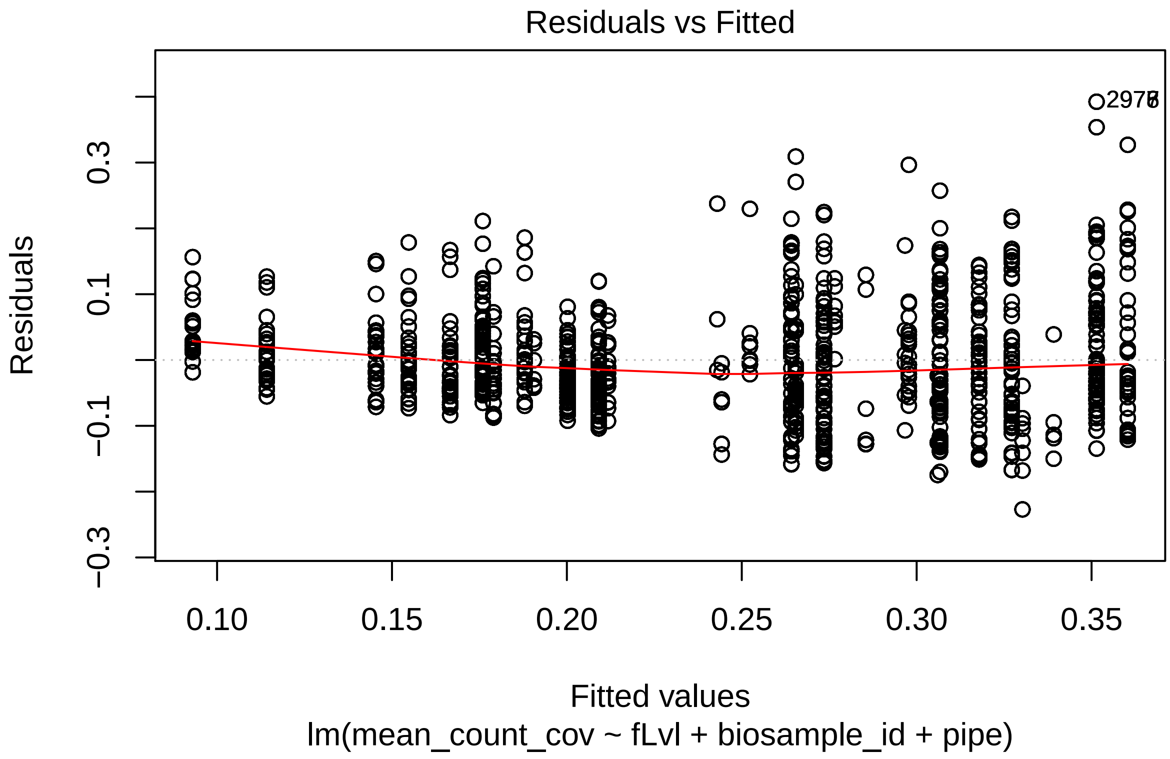
Count COV by biosample and pipeline. Unlike the bias metric, the bioinformatic pipeline used to process the sequence data contributes more to the total variability than the biological replicate, though most of the variability is between a set of features for a biological replicate and pipeline.

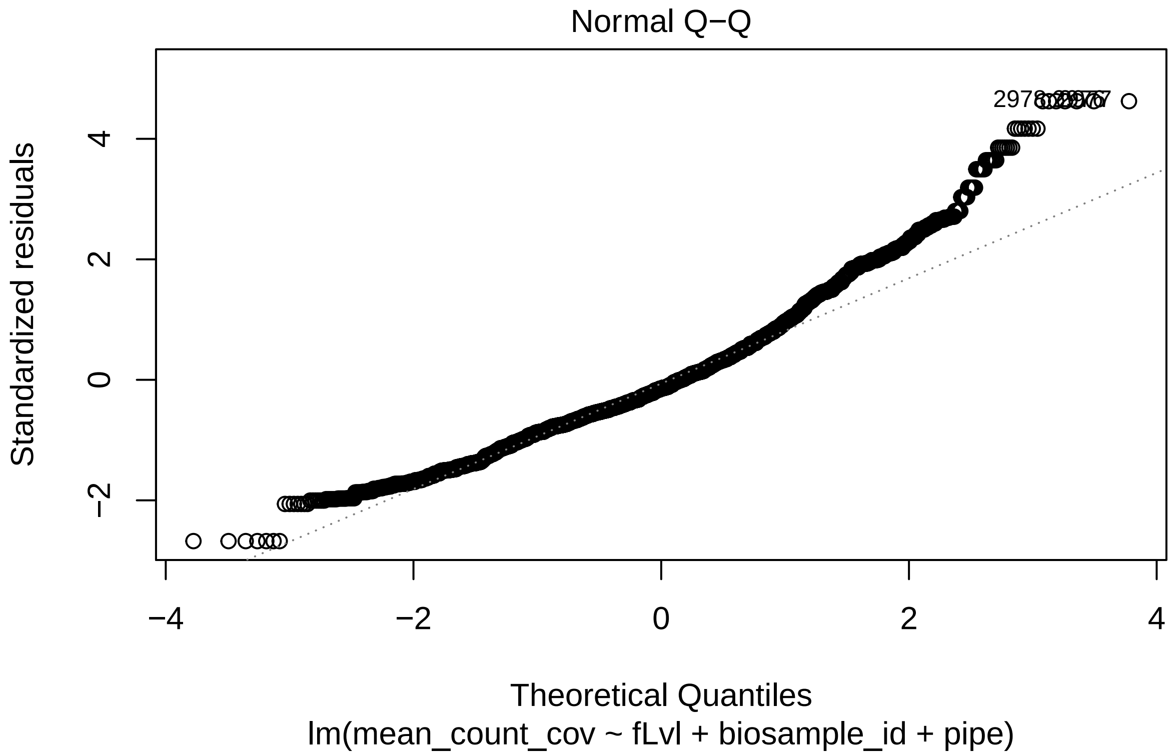
```r
var_metric %>% ggplot() +
    geom_boxplot(aes(x = biosample_id, y = mean_count_cov, color = pipe)) +
    theme_bw() + facet_grid(~fLvl) +
    theme(axis.text.x = element_text(angle = 90)) +
    labs(x = "Biological Replicate", y = "Coefficient of Variation")
```

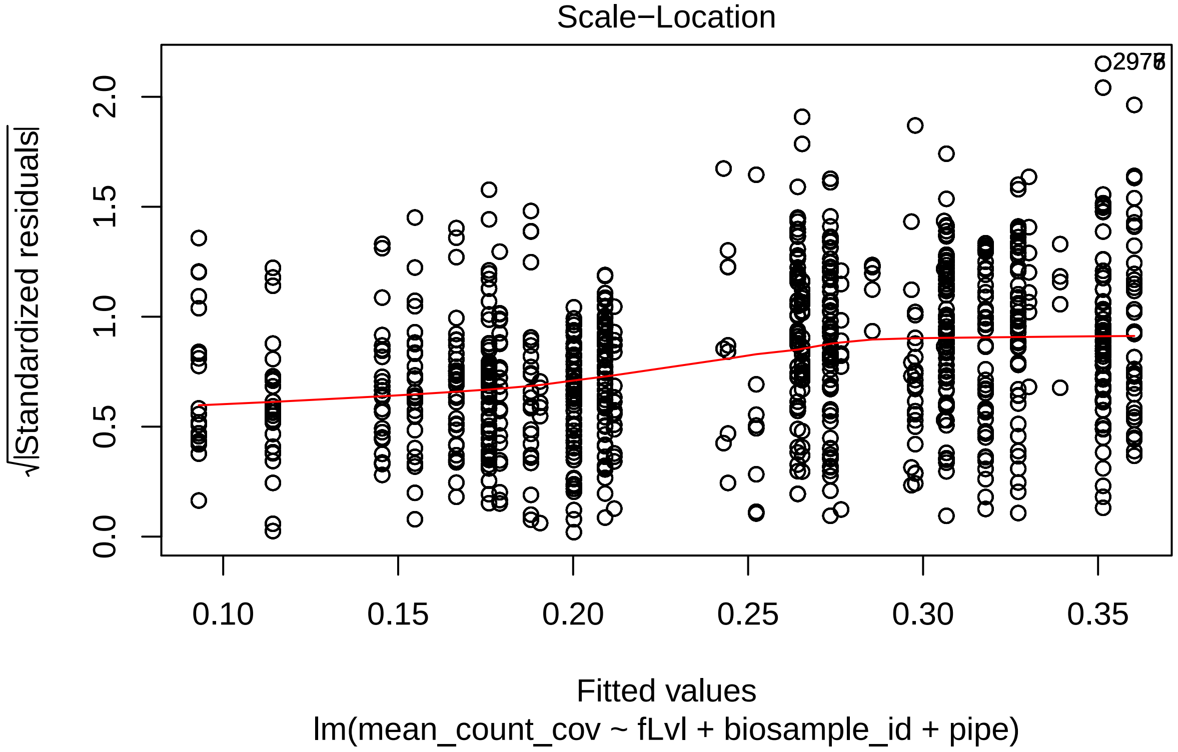


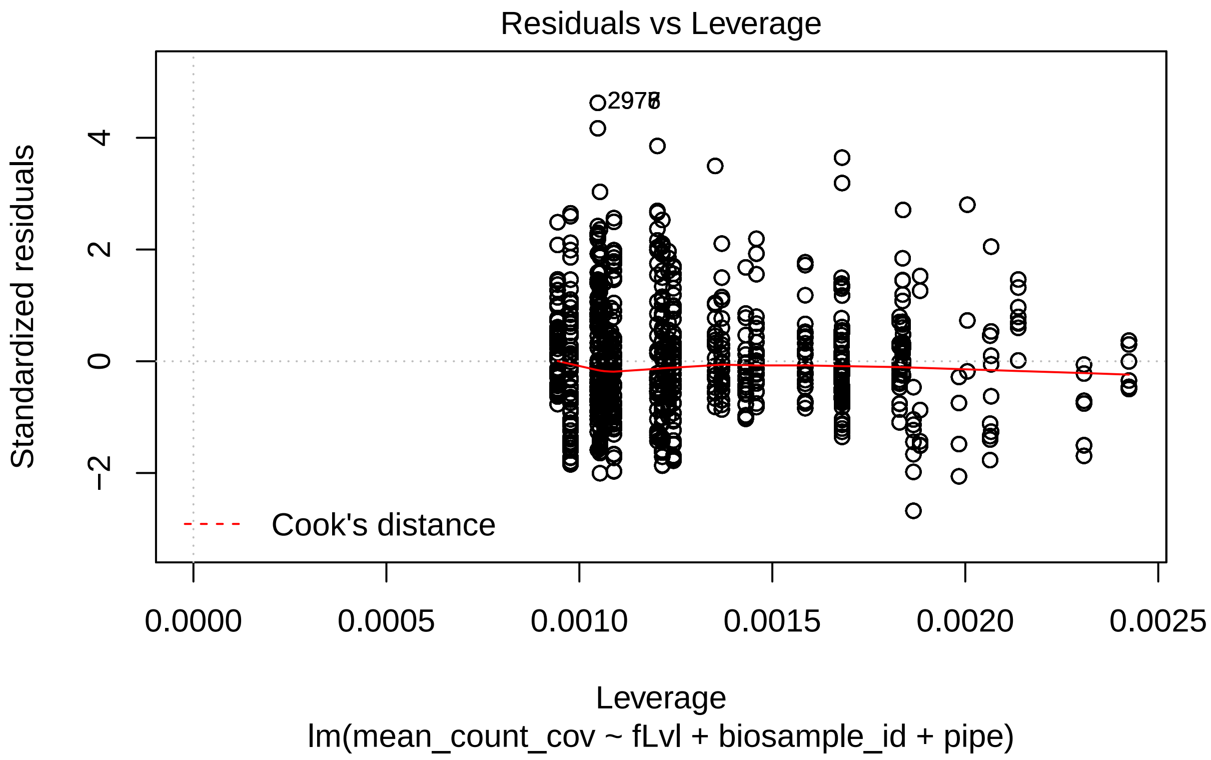
```
fit <- lm(mean_count_cov ~ fLvl + biosample_id + pipe, data = var_metric)
```

```
plot(fit)
```




Normal Q−Q
2978
Standardized residuals
4
2
0
−2
−4
−2
0
2
4
Theoretical Quantiles
lm(mean_count_cov ~ fLvl + biosample_id + pipe)


Scale−Location
2976
√|Standardized residuals|
2.0
1.5
1.0
0.5
0.0
0.10
0.15
0.20
0.25
0.30
0.35
Fitted values
lm(mean_count_cov ~ fLvl + biosample_id + pipe)



```
aov(fit) %>% broom::tidy() %>% knitr::kable()
```

term	df	sumsq	meansq	statistic	p.value
fLvl	1	4.555901	4.5559006	632.2509	0
biosample_id	4	4.895422	1.2238554	169.8421	0
pipe	2	24.558550	12.2792750	1704.0719	0
Residuals	6278	45.238283	0.0072058	NA	NA
DADA2 has the l	owest c	oefficient o	f variation		

```
aov(fit) %>% TukeyHSD() %>% broom::tidy() %>%
      filter(adj.p.value < 0.05) %>% knitr::kable()
```

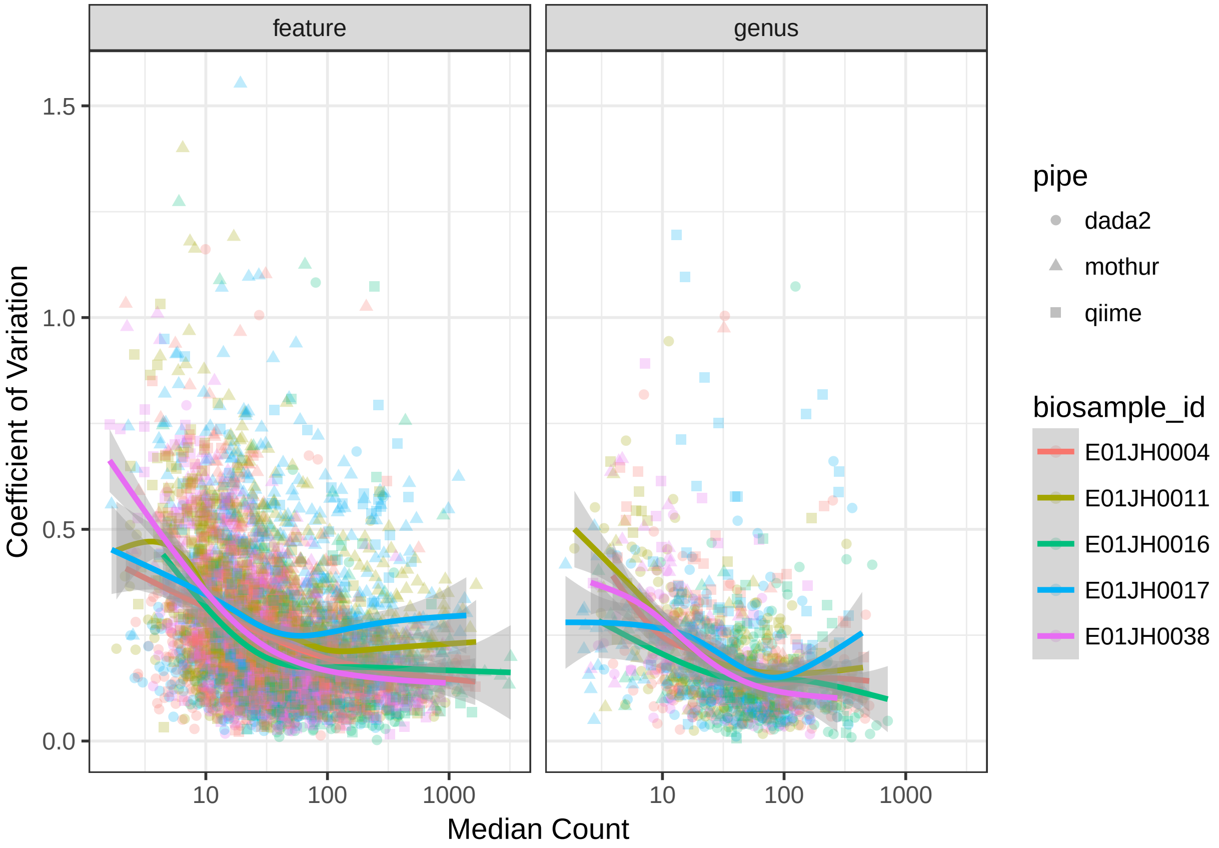
term	comparison	estimate	conf.low	conf.high	adj.p.value
fLvl	genus-feature	-0.0665650	-0.0717545	-0.0613754	0
biosample_id	E01JH0011-E01JH0004	0.0347358	0.0262402	0.0432313	0
biosample_id	E01JH0016-E01JH0004	-0.0535975	-0.0637224	-0.0434727	0
biosample_id	E01JH0017-E01JH0004	0.0320135	0.0234349	0.0405921	0
biosample_id	E01JH0016-E01JH0011	-0.0883333	-0.0987136	-0.0779529	0
biosample_id	E01JH0038-E01JH0011	-0.0339133	-0.0430815	-0.0247451	0
biosample_id	E01JH0017-E01JH0016	0.0856110	0.0751625	0.0960595	0
biosample_id	E01JH0038-E01JH0016	0.0544199	0.0437244	0.0651155	0
biosample_id	E01JH0038-E01JH0017	-0.0311911	-0.0404363	-0.0219458	0
pipe	mothur-dada2	0.1415753	0.1355259	0.1476247	0
pipe	qiime-dada2	0.0894466	0.0833769	0.0955163	0
pipe	qiime-mothur	-0.0521287	-0.0586940	-0.0455633	0

Feature level analysis - bias and variance

Relationship between count COV and median count

The feature level COV is higher for low abundance features then flattens out to ~0.25.

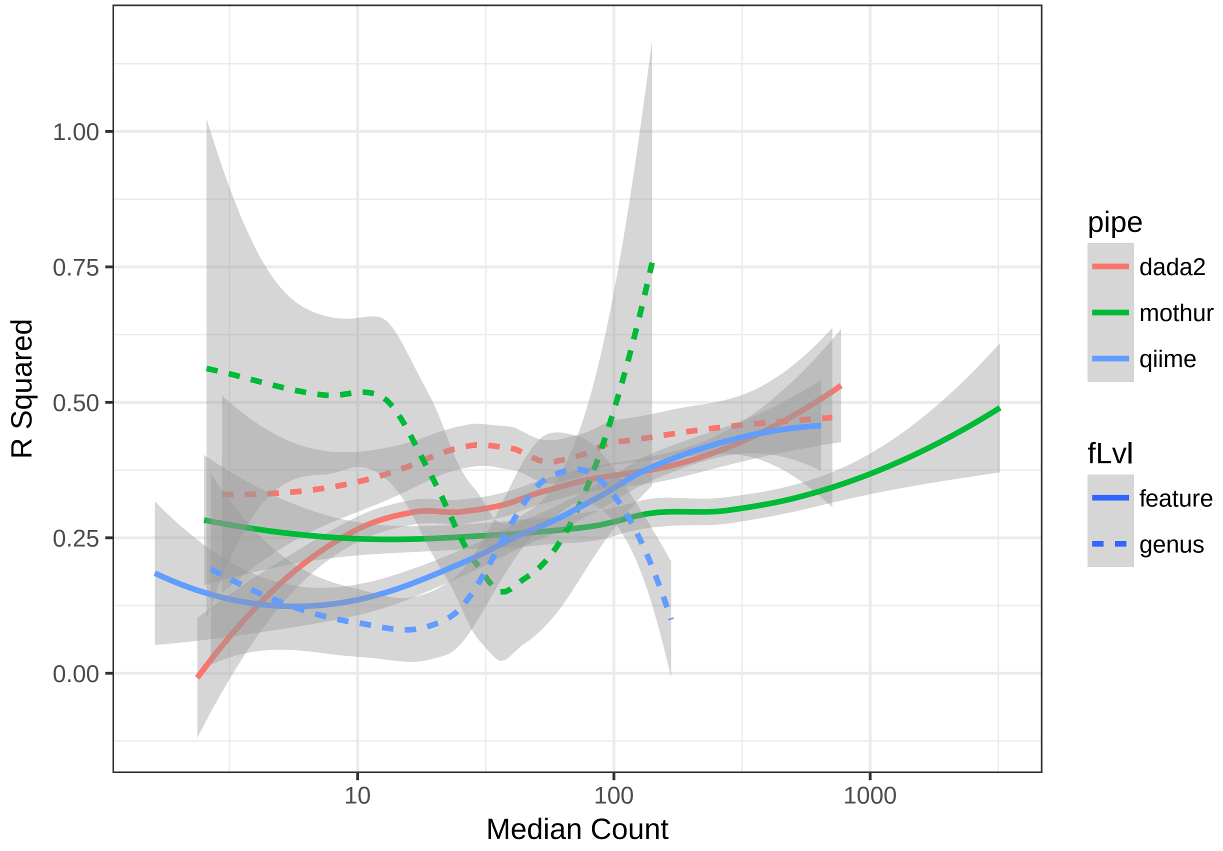
```
var_metric %>% ggplot() +
    geom_point(aes(x = med_count, y = count_cov,
                   color = biosample_id, shape = pipe),
               alpha = 0.25) +
    geom_smooth(aes(x = med_count, y = count_cov, color = biosample_id)) +
    scale_x_log10() +
    theme_bw() +
      facet_wrap(~fLvl) +
    labs(x = "Median Count", y = "Coefficient of Variation")
```



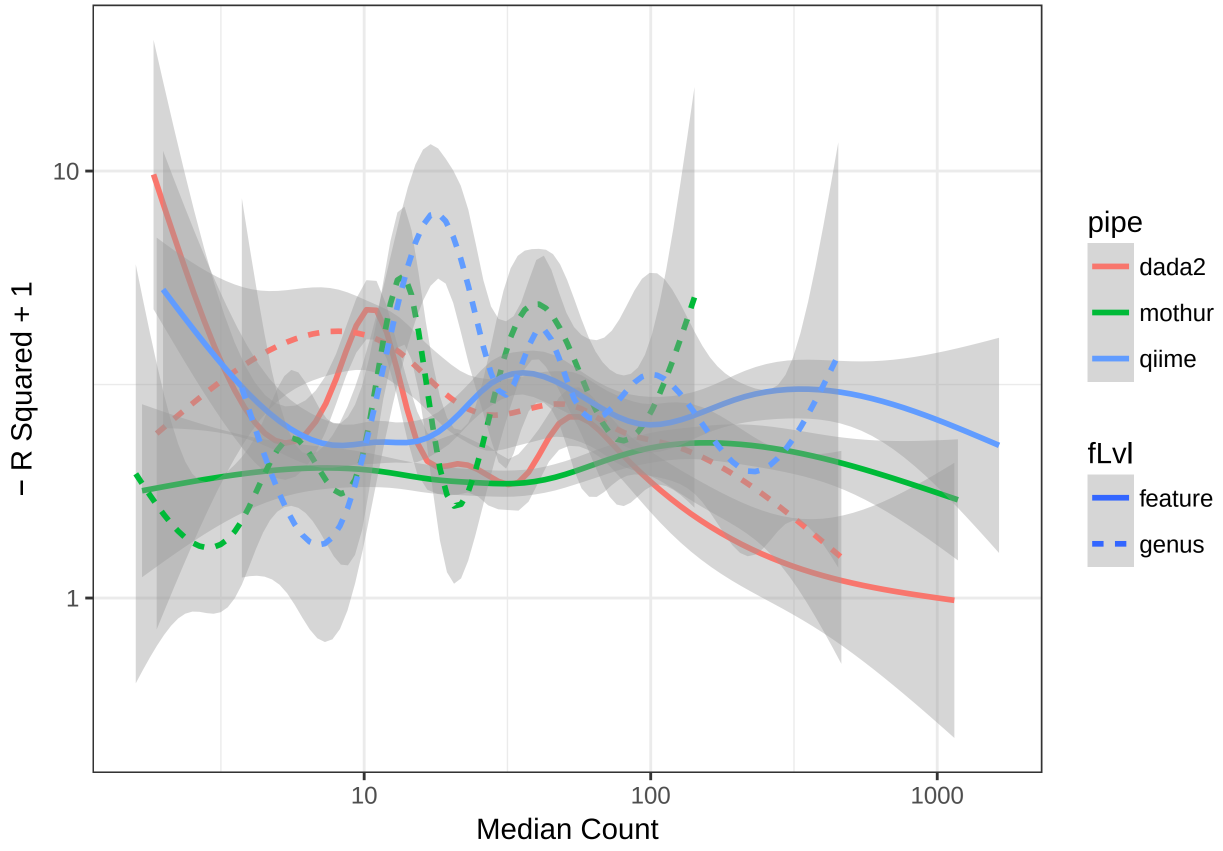
Relationship between R^2 and median count

Only looking at features with positive R^2 values

```
r2_df %>% left_join(var_metric) %>% filter(r_squared > 0) %>%
    ggplot() +
    geom_smooth(aes(x = med_count, y = r_squared, color = pipe, linetype = fLvl)) +
    scale_x_log10() +
    theme_bw() +
    labs(x = "Median Count", y = "R Squared")
```



```
r2_df %>% left_join(var_metric) %>% filter(r_squared < 0) %>%
    ggplot() +
    geom_smooth(aes(x = med_count, y = -r_squared + 1, color = pipe, linetype = fLvl)) +
    scale_x_log10() + scale_y_log10() +
    theme_bw() +
    labs(x = "Median Count", y = "- R Squared + 1")
```



- Types of outlier features - **TODO** Identification of outlier features

Relating types of outlier features with biological and experimental factors

- biological taxonomy/ phylogenetic signal
- experimental - sampling based approach

Session information

Git repo commit information

```
repo <- repository(path = ".")
last_commit <- commits(repo)[[1]]
```

The current git commit of this file is fc5d126380c480dc4343061f567df9b1507f7dd7, which is on the master branch and was made by nate-d-olson on 2017-04-11 17:49:55. The current commit message is added quant analysis and genus level NTC id. The repository is online at https://github.com/nate-d-olson/mgtst-pub

Platform Information

```
s_info <- devtools::session_info()
print(s_info$platform)
```

```
##  setting  value
##  version  R version 3.3.3 (2017-03-06)
##  system   x86_64, darwin15.6.0
##  ui       unknown
##  language (EN)
##  collate  en_US.UTF-8
##  tz       America/New_York
##  date     2017-04-11
```

Package Versions

```
s_info$packages %>% filter(`*` == "*") %>% select(-`*`) %>%
      knitr::kable()
```

package	version	date	source
ape	4.1	2017-02-14	CRAN (R 3.3.2)
dplyr	0.5.0	2016-06-24	CRAN (R 3.3.2)
forcats	0.2.0	2017-01-23	CRAN (R 3.3.2)
ggplot2	2.2.1	2016-12-30	CRAN (R 3.3.2)
git2r	0.18.0	2017-01-01	CRAN (R 3.3.2)
nlme	3.1-131	2017-02-06	CRAN (R 3.3.3)
purrr	0.2.2	2016-06-18	CRAN (R 3.3.1)
readr	1.1.0	2017-03-22	CRAN (R 3.3.2)
tibble	1.3.0	2017-04-01	CRAN (R 3.3.3)
tidyr	0.6.1	2017-01-10	CRAN (R 3.3.2)
tidyverse	1.1.1	2017-01-27	CRAN (R 3.3.2)