RNA-seq data cleaning

voom to DEG

Kim Dill-McFarland, kadm@uw.edu

version October 21, 2021

Contents

Overview

This document covers the Hawn lab recommended analysis pipeline to determine differentially expressed genes (DEG). This pipeline includes simple linear modeling and linear mixed effects modeling with main effects, interaction terms, co-variates, and random effects. There is discussion of how to choose a 'best fit' model using fit, significant genes, and biological knowledge. The example data are human, bulk, paired-end RNA-seq, but this pipeline can be applied to other organisms or single-read libraries.

0. Setup

Software

This pipeline should be completed in R and RStudio. You should also install the following packages.

```
#CRAN packages
install.packages(c("tidyverse"))

#Bioconductor packages
install.packages("BiocManager")
BiocManager::install(c("edgeR", "limma", "patchwork"))

#GitHub packages
install.packages("devtools")
devtools::install_github("BIGslu/BIGverse", force=TRUE)
```

And load them into your current R session.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(BIGverse)
```

```
## -- Attaching packages ---------------------------------------- BIGverse 0.2.0 --
```

```
## v kimma      1.0.0     v BIGpicture 0.0.1
## v RNAetc     0.1.0
```

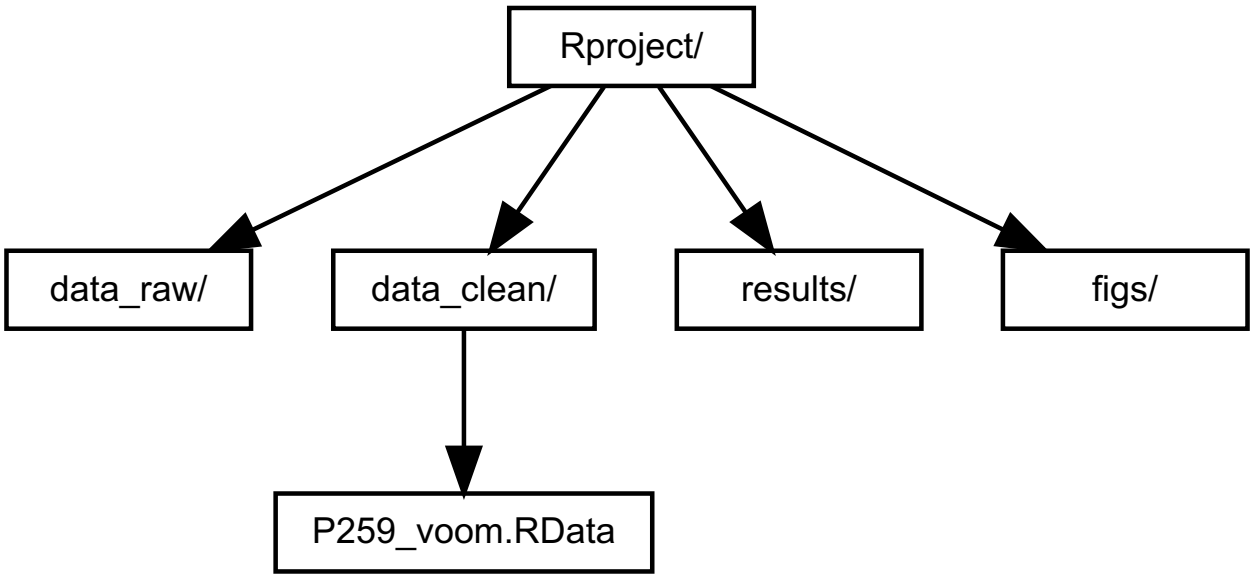
```
## -- Conflicts ------------------------------------------ BIGverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(limma)
library(edgeR)
library(patchwork)

set.seed(651)
```

Directory structure

To directly use the code in this pipeline, you must organize your files as follows.



All of the data needed are contained in a single `Rdata` object in `data_clean/`.

Example data

Example data were obtained from virus-stimulated human plasmacytoid dendritic cells (pDC). For full methods, see Dill-McFarland *et al.* 2021. Eosinophil-mediated suppression and Anti-IL-5 enhancement of plasmacytoid dendritic cell interferon responses in asthma. J Allergy Clin Immunol. In revision GitHub. Some patient identifiers and metadata have been altered for this tutorial. Of note, these data were obtained from the same data set as the `kimma` package.

Specifically, this tutorial uses data processed using our fastq to counts and counts to voom pipelines, resulting in voom-normalized, log2 counts per million (CPM) expression and associated sample metadata in a `limma EList` object in the `data_clean/` directory.

1. Load data

All counts, gene, and sample metadata are contained in a single object.

```
#Attach data
attach("data_clean/P259_voom.RData")
#Extract and rename data object
dat <- dat.abund.norm.voom
```

We access each data frame within this `Elist` using `$`. The normalized log2 CPM expression data are contained in `E`.

```
dat$E[1:3,1:7]
```

```
##                     lib1    lib10     lib2     lib3     lib4     lib5     lib6
## ENSG00000186827 7.610795 5.917194 6.549302 7.706893 6.423839 6.702732 6.192989
## ENSG00000186891 7.479251 4.542296 5.543352 6.837607 3.876029 5.438202 4.108455
## ENSG00000160072 5.454945 5.723903 4.690578 4.592769 5.048669 5.787408 5.590324
```

Library and donor metadata are in `targets`.

```
dat$targets[1:3,1:10]
```

```
##       group lib.size norm.factors libID   ptID virus batch  asthma age sex
## lib1      1  1174099    1.0716309  lib1 donor1  none     A healthy  35   F
```

```
## lib10     1  2639415    0.6808503 lib10 donor5   HRV     B  asthma  26   M
## lib2      1  1297298    1.0053665  lib2 donor1   HRV     A healthy  35   F
```

Gene metadata are in `genes`.

```
dat$genes[1:3,1:4]
```

```
##                 ensembl_gene_id entrezgene_id   gene_biotype chromosome_name
## ENSG00000186891 ENSG00000000419          8813 protein_coding              20
## ENSG00000160072 ENSG00000000457         57147 protein_coding               1
## ENSG00000041988 ENSG00000000460         55732 protein_coding               1
```

There are a couple other data frames in our `dat` object, but they are not relevant to this tutorial.

Additionally, models can take a couple minutes to run. You can load all model results for this tutorial.

```
load("results/model_results.RData")
```

2. Simple linear model

2.1: A single main effect (virus)

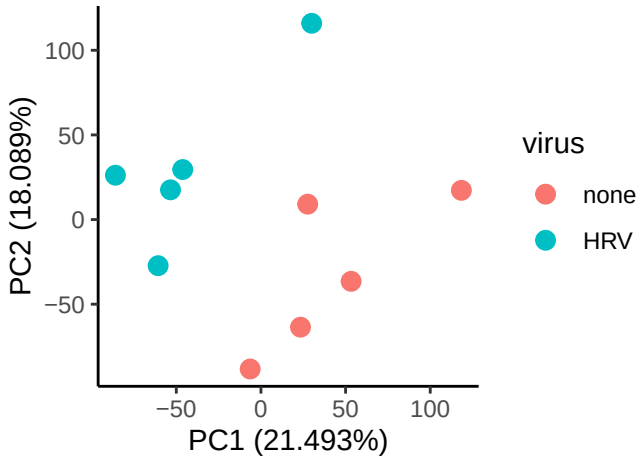
First, we consider our research question and what variable(s) we need to answer that question. In these data, the first variable of interest is `virus` to determine how viral infection impacts gene expression. In R, this model is written as:

```
~ virus
```

On a coarse scale such as PCA, we can see that `virus` impacts gene expression with uninfected controls (none) grouping together away from HRV-infected samples.

```
plot_pca(dat, vars = "virus")
```

```
## Joining, by = "libID"
## $virus
```



We run this linear model in `kimma` using `kmFit`.

```
virus <- kmFit(dat = dat, model = "~ virus", run.lm = TRUE)
```

```
Running models on 5 individuals. No kinship provided.
lm model: expression~ virus
12830 genes complete
```

We see that numerous genes are significant for `virus`. However, this is not the best model for these data since we have other factors that may impact expression.

```
summarise_kmFit(fdr = virus$lm)
```

```
## # A tibble: 2 x 8
##   model variable             fdr_0.05 fdr_0.1 fdr_0.2 fdr_0.3 fdr_0.4 fdr_0.5
##   <chr> <fct>                   <int>   <int>   <int>   <int>   <int>   <int>
## 1 lm    virusHRV                  169     547    1218    1866    2800    3690
## 2 lm    total (nonredundant)      169     547    1218    1866    2800    3690
```

2.2: Multiple main effects (virus & asthma)

We are actually interested in how individuals with and without asthma respond to virus. We can add variables to our model with `+` such as

`~ virus + asthma`

However, this model only captures the main effects of each variable in isolation. Specifically, this model tells you how virus impacts genes expression and how asthma impacts gene expression. It does not address how viral impacts *differ* between those with and without asthma.

Interaction terms

One way to assess this is with an interaction term written as:

`~ virus + asthma + virus:asthma`

or short-handed with `*`. Note, these two models are equivalent in R.

`~ virus * asthma`

This model now tests both the main effects and their interaction.

```
virus_interact <- kmFit(dat = dat, model = "~ virus*asthma", run.lm = TRUE)
```

```
Running models on 5 individuals. No kinship provided.
lm model: expression~ virus*asthma
12830 genes complete
```

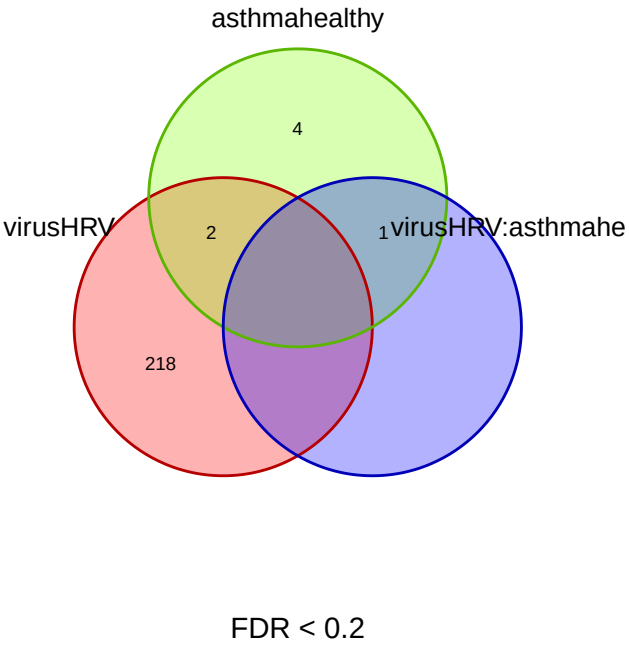
We now see 3 variables in our results equivalent to the variables in the long form of our model equation. Notice that we've lost significance for `virus`, and the lowest FDR cutoff for any significance is 0.2. Because this data set is small, an interaction model may be too complex, and we may not have the power to see interaction effects.

```
summarise_kmFit(fdr = virus_interact$lm)
```

```
## # A tibble: 4 x 8
##   variable               model fdr_0.05 fdr_0.1 fdr_0.2 fdr_0.3 fdr_0.4 fdr_0.5
##   <fct>                  <chr>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 asthmahealthy          lm           0       0       7       8     105     383
## 2 virusHRV               lm           0       0     220     415     712    1099
## 3 virusHRV:asthmahealthy lm           0       0       1       1       1       1
## 4 total (nonredundant)   lm           0       0     225     421     769    1303
```

Importantly, a gene with a significant interaction term cannot be assessed for the main effects. For example, there is 1 gene here that is significant for the interaction (blue) and the asthma main term (green). However, we *cannot* use this asthma result, because it is comparing all healthy to all asthma samples without taking virus into account. Since we know there is an interaction effect for this gene, the asthma comparison alone is incorrectly averaging across samples we know to be different (none vs HRV).

```
plot_venn_genes(model_result = virus_interact, model = "lm", fdr.cutoff = 0.2)
```



If this were our final model, our DEG list would be all interaction genes (blue) as well as the intersect of virus and asthma main terms (red-green). This second group encompasses genes that change with virus similarly in healthy and asthma donors but are always higher in one asthma group.

Pairwise contrasts

Another way to model interactions is with pairwise contrasts. This means that instead of including the `virus:asthma` term, you create a new term that includes both variables like `virus_asthma`. You then model that term and select the pairwise comparisons that are meaningful between your groups.

`~ virus_asthma`

First, we create our new combined variables in the metadata.

```
dat$targets <- dat$targets %>%
  rowwise() %>%
  mutate(virus_asthma = paste(virus, asthma, sep="_")) %>%
  #set reference level for new variable
  mutate(virus_asthma = fct_reorder(factor(virus_asthma), "none_healthy"))
```

Then we run the model and specify that we also want to run contrasts.

```
virus_contrast <- kmFit(dat = dat, model = "~ virus_asthma",
                        run.lm = TRUE, run.contrast = TRUE)
```

```
Running models on 5 individuals. No kinship provided.
lm model: expression~ virus_asthma
12830 genes complete
```

We now see 3 different variables in our model results. These represent each of the groups compared to the reference level we set, 'none_healthy'.

```
summarise_kmFit(fdr = virus_contrast$lm)
```

```
## # A tibble: 4 x 8
```

```
##   model variable                fdr_0.05 fdr_0.1 fdr_0.2 fdr_0.3 fdr_0.4 fdr_0.5
##   <chr> <fct>                      <int>   <int>   <int>   <int>   <int>   <int>
## 1 lm    virus_asthmaHRV_asthma        NA      NA     118     490    1026    1722
## 2 lm    virus_asthmaHRV_healthy        3       3     249     676    1201    1955
## 3 lm    virus_asthmanone_asthma       NA      NA       7       8     105     383
## 4 lm    total (nonredundant)           3       3     312     951    1797    2997
```

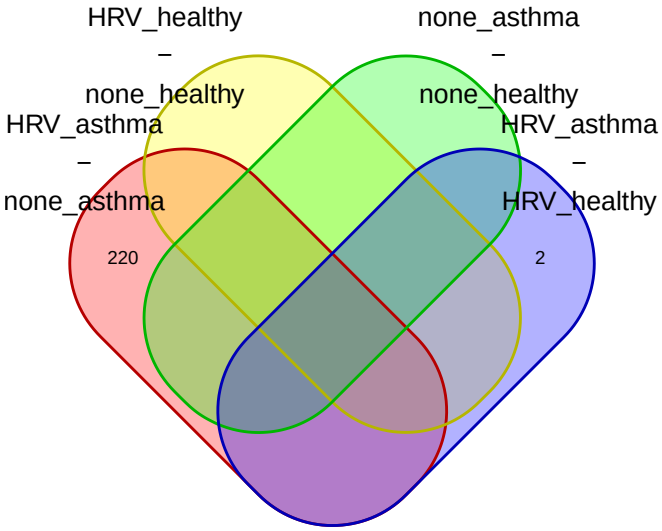
We also get all pairwise contrasts between the 4 groups. The names are cutoff but our contrasts are none_healthy - HRV_asthma, none_healthy - HRV_healthy, none_healthy - none_asthma, HRV_asthma - HRV_healthy, HRV_asthma - none_asthma, HRV_healthy - none_asthma‘

```
summarise_kmFit(fdr = virus_contrast$lm.contrast, contrast = TRUE)
```

```
## # A tibble: 7 x 9
##   model       variable contrast fdr_0.05 fdr_0.1 fdr_0.2 fdr_0.3 fdr_0.4 fdr_0.5
##   <chr>       <fct>    <chr>       <int>   <int>   <int>   <int>   <int>   <int>
## 1 lm.contrast virus_a~ HRV_hea~       96     413    1177    2154    3394    4702
## 2 lm.contrast virus_a~ none_he~        3       3     249     676    1201    1955
## 3 lm.contrast virus_a~ HRV_ast~       NA       2       2      11      16      75
## 4 lm.contrast virus_a~ HRV_ast~       NA      NA     220     415     712    1099
## 5 lm.contrast virus_a~ none_he~       NA      NA     118     490    1026    1722
## 6 lm.contrast virus_a~ none_he~       NA      NA       7       8     105     383
## 7 lm.contrast total (~ <NA>           96     413    1280    2500    3995    5644
```

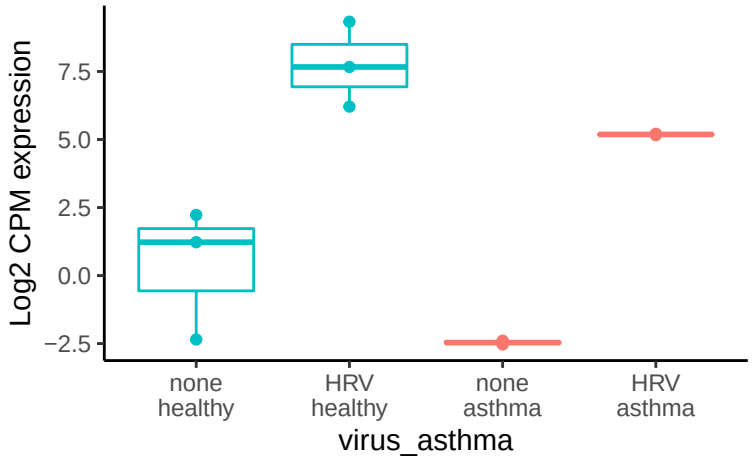
Not all of these contrasts are of interest, so we select just the effects of virus (red, yellow) and asthma (green, blue). We see that many genes change with virus in the asthma group (red) and a couple are different between health and asthma donors in the presence of virus (blue). The lack of significant genes in healthy donors (yellow) or uninfected samples (green) is an artefact of this small data set with only 2 - 3 donors per group. In a real analysis, you may be interested in genes that only change with virus in one group (red and yellow no overlap) or those that change with virus (red or yellow) and are different with asthma (green or blue).

```
plot_venn_genes(model_result = virus_contrast, model = "lm.contrast",
                fdr.cutoff = 0.2,
                contrasts = c("HRV_asthma - none_asthma",
                              "HRV_healthy - none_healthy",
                              "none_asthma - none_healthy",
                              "HRV_asthma - HRV_healthy"))
```

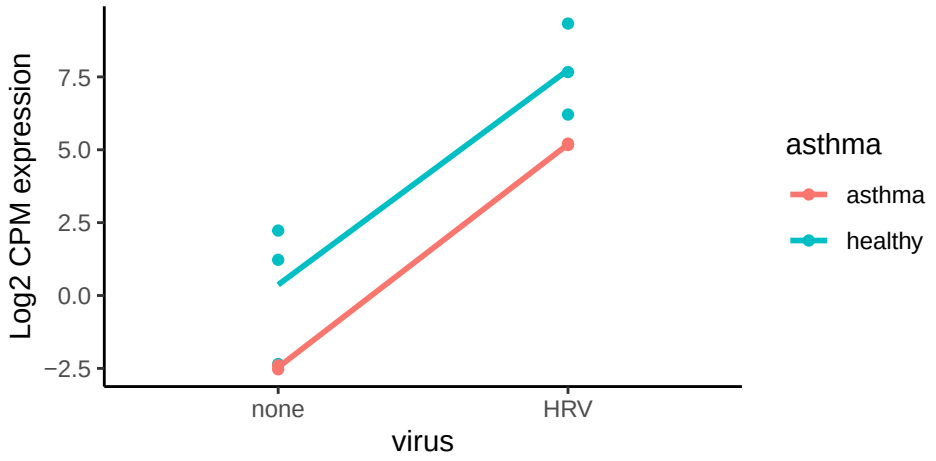


When to use interaction vs contrasts

At their heart, interaction and contrast models are trying to answer the same question. However, statistically, they are very different. Contrasts compare means between groups (see below) and you must select which pairwise comparisons are meaningful.



An interaction term tests if two slopes differ. In these data, this is comparing the change (slope) in response to virus in healthy vs asthmatic donors like below In most cases, it is more difficult to achieve significance when comparing slopes as opposed to means.



In general, we recommend using the interaction term to define differentially expressed genes (DEG). Then, as a post-hoc test, run contrasts only on significant DEGs to further probe the results. This is demonstrated later in this tutorial. It is like running an ANOVA to compare groups A,B,C and then TukeyHSD to determine which groups differ from which (A vs B, B vs C, A vs C).

Contrasts may be employed as your main model instead in cases such as:

- Small data sets where you are under-powered and would benefit from the reduced complexity of contrasts (1 variable) vs an interaction (3 variables)
- When there is no significance for the interaction term
- When you are only interested in a subset of the pairwise comparisons encompassed by the interaction term. For example, if you wanted to test a longitudinal study as time point 1 vs 2, 2 vs 3, 3 vs 4, etc

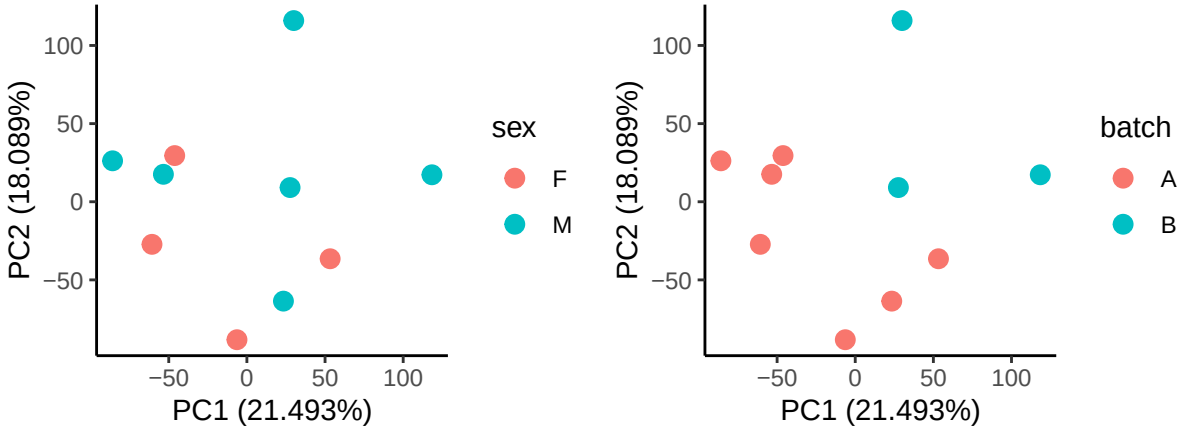
2.3: Co-variates (batch & sex)

We also want to consider the effects of variables that may impact or prevent us from seeing our main effects. In human studies, this often includes age and sex, though we only have sex for these data. We also want to consider batch since these data were sequenced in two batches.

To determine which co-variates to include, let's see if they have a large impact on the data in PCA.

```
plot_pca(dat, vars = c("sex","batch")) %>%
  wrap_plots(ncol=2)
```

```
## Joining, by = "libID"
```



We see some grouping by batch; thus, we have evidence to include batch in the model despite the fact that these data we batch-corrected using `ComBat-Seq`. This is not uncommon with batch correction as we err on

the side of under-correcting by batch to retain true biological variation. In contrast, sex does not show any groupings and may not be needed. PCA alone, however, is not enough to determine what co-variates need to be in our models.

Next, we run the models with and without co-variates for comparison. To include co-variates, we add them to the models with +.

```
virus_batch <- kmFit(dat = dat, model = "~ virus*asthma + batch", run.lm = TRUE)
```

```
Running models on 5 individuals. No kinship provided.
lm model: expression~ virus*asthma + batch
12830 genes complete
```

```
virus_batch_sex <- kmFit(dat = dat, model = "~ virus*asthma + batch + sex",
                         run.lm = TRUE)
```

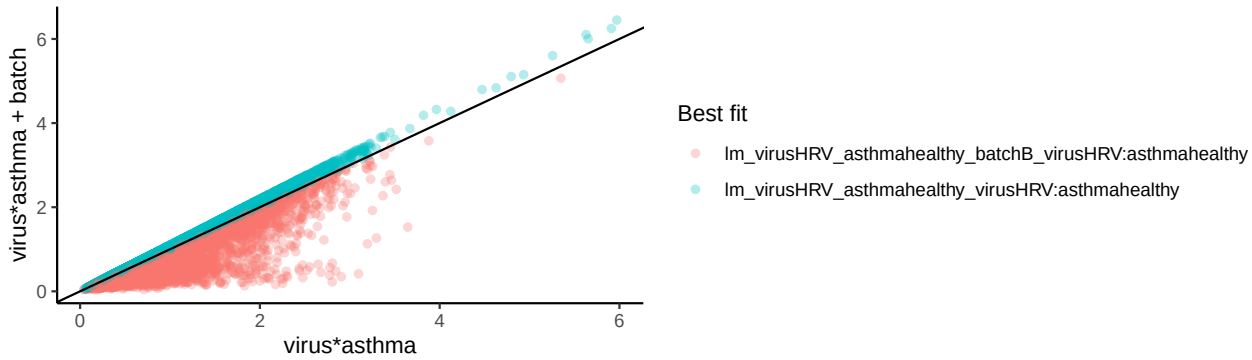
```
Running models on 5 individuals. No kinship provided.
lm model: expression~ virus*asthma + batch + sex
12830 genes complete
```

We compare model fits with sigma, the mean of residuals for each gene. Smaller sigmas indicate a better fitting model so genes below the 1:1 line are better fit by the y-axis model (red). Those above the line are better fit by the x-axis model (teal). Throughout this tutorial, we put the more complex model as y. Our plotting function also outputs a message with how many genes are best fit by each model.

```
plot_sigma(model_result = virus_interact, model_result_y = virus_batch,
           x="lm", y="lm") +
  labs(x = "virus*asthma", y = "virus*asthma + batch")
```

```
## Total genes best fit by
```

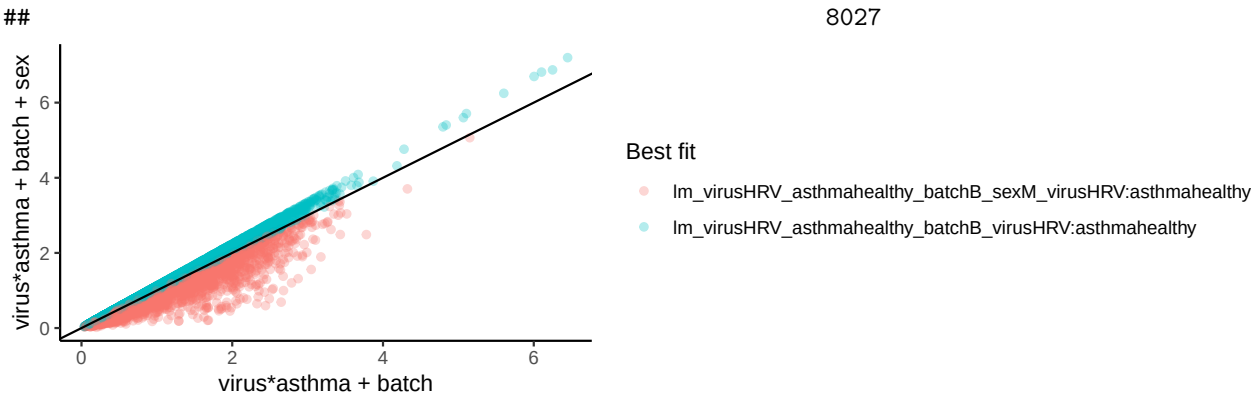
```
##
## lm_virusHRV_asthmahealthy_batchB_virusHRV:asthmahealthy
##                                                    6956
##        lm_virusHRV_asthmahealthy_virusHRV:asthmahealthy
##                                                    5874
```




```
plot_sigma(model_result = virus_batch, model_result_y = virus_batch_sex,
           x="lm", y="lm") +
  labs(x = "virus*asthma + batch", y = "virus*asthma + batch + sex")
```

```
## Total genes best fit by
```

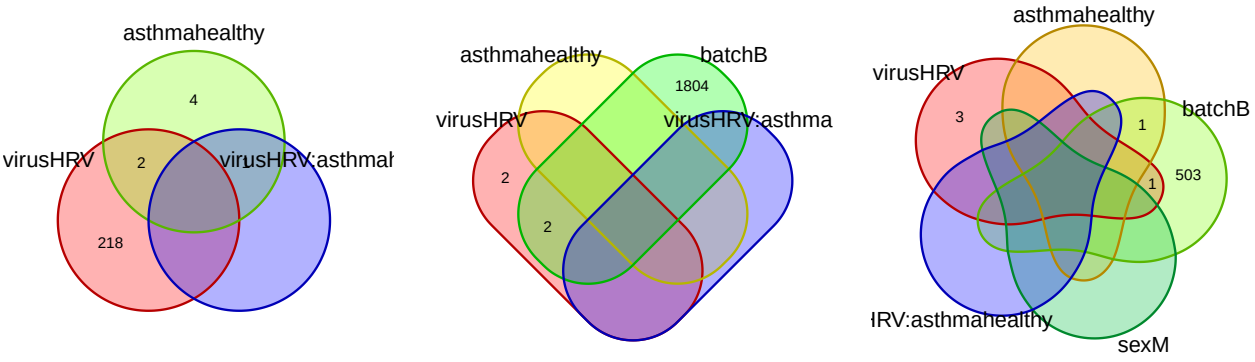
```
##
## lm_virusHRV_asthmahealthy_batchB_sexM_virusHRV:asthmahealthy
##                                                         4803
##      lm_virusHRV_asthmahealthy_batchB_virusHRV:asthmahealthy
```



We see that each model is the best fit for a subset of genes. In the first plot, batch improves the model fit for just over half of genes, and this improvement is substantial for some genes (*e.g.* red dots far from the 1:1 line). In the second plot, sex further improves the fit for roughly a third of genes.

Next, we compare how many genes are significant with and without co-variates. We arbitrarily select a high FDR cutoff of 0.2 though you may wish to look at a couple.

```
plot_venn_genes(model_result = virus_interact, model = "lm",
                fdr.cutoff = 0.2) +
plot_venn_genes(model_result = virus_batch, model = "lm",
                fdr.cutoff = 0.2) +
plot_venn_genes(model_result = virus_batch_sex, model = "lm",
                fdr.cutoff = 0.2)
```



FDR < 0.2 FDR < 0.2 FDR < 0.2

First, consider how many genes are significant for the co-variates themselves. Batch is significant for many genes while sex is not significant for any. Next, compare the number of virus-significant genes. Adding batch and/or sex to the model decreases the number of virus (red) as well as interaction significant genes (blue). The dramatic decrease in virus genes is likely evidence that we do not have enough power to support the co-variate model complexity (*e.g.* too few samples relative to the number of variables). Given the size of this data set (N = 5), this is not surprising.

To summarize:

Batch

- Strongly impacts overall gene expression in PCA
- Improves model fit for a small majority of genes
- Significant for many genes
- Decreases the number of virus-significant genes

Sex

- Does not impact overall gene expression in PCA
- Improves model fit for a minority of genes
- Not significant for any genes
- Decreases the number of virus-significant genes

If it weren't for the impacts on virus genes, we would have clear evidence to include batch and exclude sex from the model. However, given the size of this data set and evidence that even 1 co-variate cannot be supported, we will not include any at this time.

`~ virus*asthma`

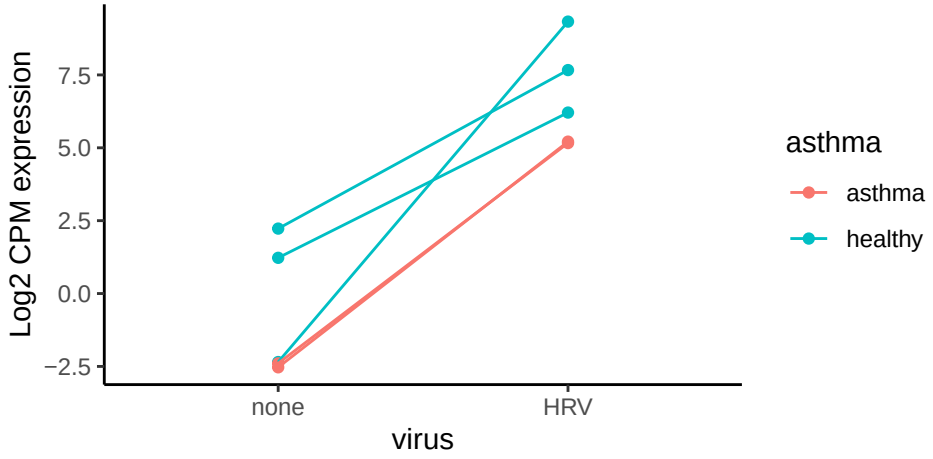
In your data, you may have co-variates that are not as clear as these. In that case, it's important to prioritize model fit and sample size over significant gene counts. You should not include or exclude a co-variate just because it gets you more significant genes for your main terms. This is especially true if the co-variate negatively impacts model fit.

You may also have non-statistical evidence for including a co-variate. If you have established biological evidence that a co-variate is important in your system or it's standard in your field, you may wish to include it even if it does not improve fit and is not significant. If the co-variate, however, greatly worsens fit, you should not include it even if it's standard. Instead, present the sigma plots to support its exclusion.

3. Linear mixed effects model

3.1: Random effects (ptID)

These data come from a paired study design with uninfected (none) and infected (HRV) samples from the same donor's cells. We take this into account in our model by using donor as a random effect. This allows the model to match samples from the same donor. It greatly improves our statistical power as we now have 1 slope per donor instead of 1 mean slope across all donors. So, we can see if all individual donors change similarly or not.



Random effects are added to the model with `+ (1|block)` where the block is the variable you want to pair samples by. Thus, we now run a mixed effects model `lme` with

`~ virus*asthma + (1|ptID)`

In `kimma`, this is run similarly to a simple model except we add our random term and ask it to `run.lme`.

```
virus_lme <- kmFit(dat = dat, model = "~ virus*asthma + (1|ptID)", run.lme = TRUE)
```

```
Running models on 5 individuals. No kinship provided.
lme/lmekin model: expression~virus*asthma+(1|ptID)
```

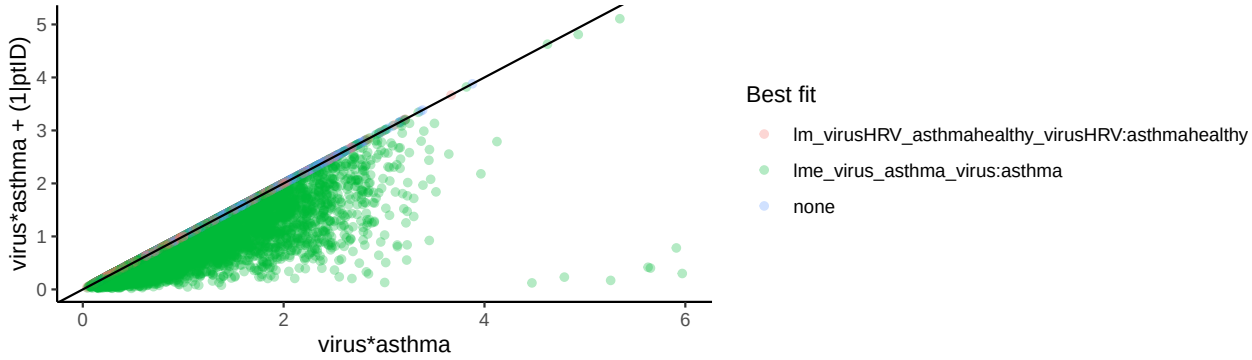
```
12830 genes complete
```

Note, we could run both models at once in `kmFit` if we set `run.lm=TRUE, run.lme=TRUE`. Since we already ran the simple linear model, we'll skip that here.

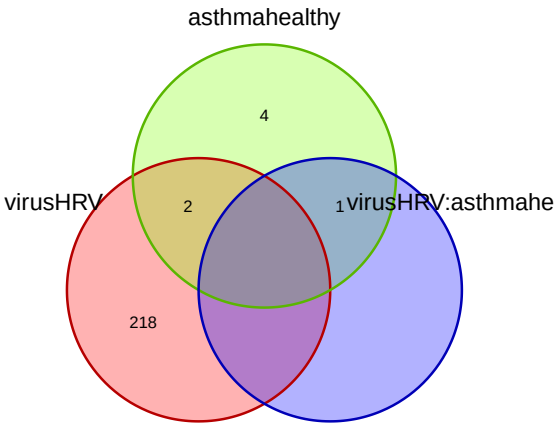
Comparing models as we did with co-variates, we see that adding the random effect improved the model fit for the majority of genes ($> 80\%$) and increased the number of virus-significant and interaction DEG. Note there is also a 'none' category where both models had the same fit (sigma). This illustrates the much improved power of a mixed effects model when you have paired samples.

```
plot_sigma(model_result = virus_interact, model_result_y = virus_lme,
           x="lm", y="lme") +
  labs(x = "virus*asthma", y = "virus*asthma + (1|ptID)")
```

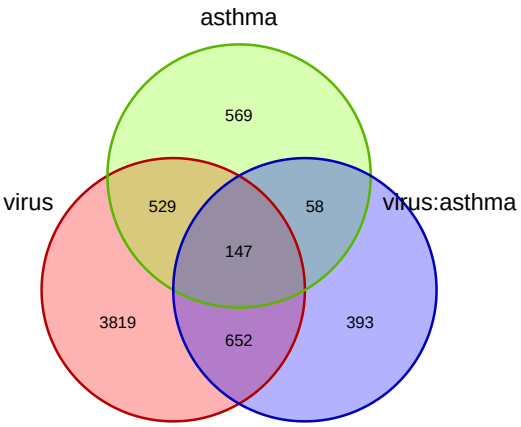
```
## Total genes best fit by
##
## lm_virusHRV_asthmahealthy_virusHRV:asthmahealthy
##                                            1357
##                     lme_virus_asthma_virus:asthma
##                                           10493
##                                            none
##                                             980
```




```
plot_venn_genes(model_result = virus_interact, model = "lm",
                fdr.cutoff = 0.2) +
plot_venn_genes(model_result = virus_lme, model = "lme",
                fdr.cutoff = 0.2)
```



FDR < 0.2



FDR < 0.2

Comparison to `limma`

`limma` offers a sudo random effect in linear modeling with `duplicateCorrelation`. This estimates the random effect of donor across all genes and uses one value for all models. The full mixed effect model in `kimma` estimates the random effect for each gene, thus fitting a different value for each gene's model.

In our analyses, we've found `limma` and `kimma` results to be similar when the sample size or variable effects are large. In the case of small data sets or small effects, however, there can be a dramatic difference.

For example, we run a `limma` model for these data.

```
mm_limma <- model.matrix(~ virus*asthma, data=dat$targets)

#Block by donor
consensus.corr <- duplicateCorrelation(dat$E, mm_limma,
                      block=dat$targets$ptID)$consensus.correlation
consensus.corr
```

```
## [1] 0.3155376
```

```
# Fit model to transformed count data. Calculate eBayes
fit_limma <- eBayes(lmFit(dat$E, mm_limma,
                                  block=dat$targets$ptID,
                                  correlation=consensus.corr))

#Extract pval
virus_limma <- extract_lmFit(design = mm_limma, fit = fit_limma)
```

We see that not taking the paired sample design into account results in few DEG. Using duplicate correlation in `limma` gains some but `kimma`'s full mixed effects model best detects signal across all variables.

```
# ~ virus*asthma in kimma
summarise_kmFit(fdr = virus_interact$lm)
```

```
## # A tibble: 4 x 8
##   variable              model fdr_0.05 fdr_0.1 fdr_0.2 fdr_0.3 fdr_0.4 fdr_0.5
##   <fct>                 <chr>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 asthmahealthy         lm           0       0       7       8     105     383
## 2 virusHRV              lm           0       0     220     415     712    1099
```

```
## 3 virusHRV:asthmahealthy lm           0       0       1       1       1       1
## 4 total (nonredundant)   lm           0       0     225     421     769    1303
```

```
# ~ virus*asthma + duplicate correlation in limma
summarise_lmFit(fdr = virus_limma)
```

```
## # A tibble: 4 x 7
##   variable               fdr_0.05 fdr_0.1 fdr_0.2 fdr_0.3 fdr_0.4 fdr_0.5
##   <fct>                     <int>   <int>   <int>   <int>   <int>   <int>
## 1 asthmahealthy                 0       0      27      48     118     247
## 2 virusHRV                    170     370     868    1425    2090    2837
## 3 virusHRV:asthmahealthy        4       6      28      50      98     155
## 4 total (nonredundant)        170     371     884    1447    2139    2935
```

```
# ~ virus*asthma + (1|ptID) in kimma
summarise_kmFit(fdr = virus_lme$lme)
```

```
## # A tibble: 4 x 8
##   model variable             fdr_0.05 fdr_0.1 fdr_0.2 fdr_0.3 fdr_0.4 fdr_0.5
##   <chr> <fct>                   <int>   <int>   <int>   <int>   <int>   <int>
## 1 lme   asthma                    625     870    1303    1728    2138    2757
## 2 lme   virus                    3380    4096    5147    6035    6880    7719
## 3 lme   virus:asthma              753     946    1250    1614    2069    2562
## 4 lme   total (nonredundant)     4035    4906    6167    7247    8200    9201
```

3.2 Co-variates (batch & sex)

Given the large changes with a mixed effect model, let's re-consider batch and sex co-variates.

```
virus_batch_lme <- kmFit(dat = dat, model = "~ virus*asthma + batch + (1|ptID)",
                         run.lme = TRUE)
```

```
Running models on 5 individuals. No kinship provided.
lme/lmekin model: expression~virus*asthma+batch+(1|ptID)
12830 genes complete
```

```
virus_batch_sex_lme <- kmFit(dat = dat,
                             model = "~ virus*asthma + batch + sex + (1|ptID)",
                             run.lme = TRUE)
```

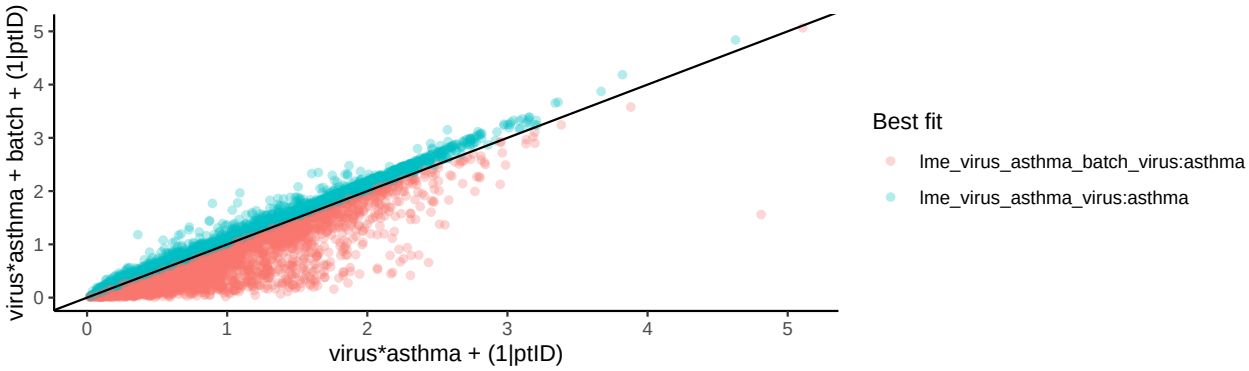
```
Running models on 5 individuals. No kinship provided.
lme/lmekin model: expression~virus*asthma+batch+sex+(1|ptID)
12830 genes complete
```

We see no clear evidence to keep co-variates based on model fit since genes are split for best fit model.

```
plot_sigma(model_result = virus_lme, model_result_y = virus_batch_lme,
           x="lme", y="lme") +
  labs(x = "virus*asthma + (1|ptID)", y = "virus*asthma + batch + (1|ptID)")
```

```
## Total genes best fit by
```

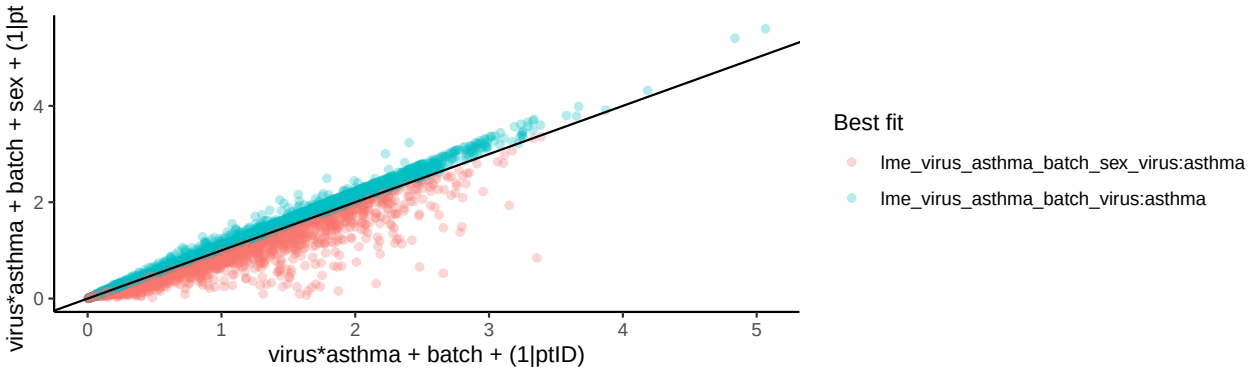
```
##
## lme_virus_asthma_batch_virus:asthma       lme_virus_asthma_virus:asthma
##                                6472                                6358
```




```
plot_sigma(model_result = virus_batch_lme, model_result_y = virus_batch_sex_lme,
           x="lme", y="lme") +
  labs(x = "virus*asthma + batch + (1|ptID)",
       y = "virus*asthma + batch + sex + (1|ptID)")
```

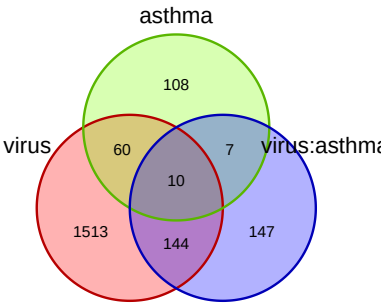
```
## Total genes best fit by
```

```
##
## lme_virus_asthma_batch_sex_virus:asthma     lme_virus_asthma_batch_virus:asthma
##                                    4833                                    7997
```

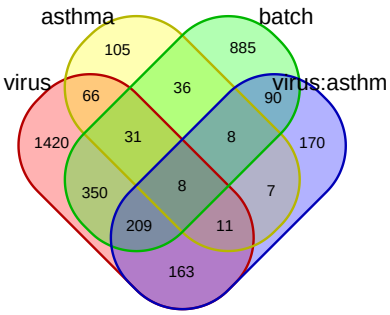


Both co-variates increase the number of virus-significant genes and are themselves significant for a number of genes. Here, we use a much lower FDR cutoff so the values are easier to see.

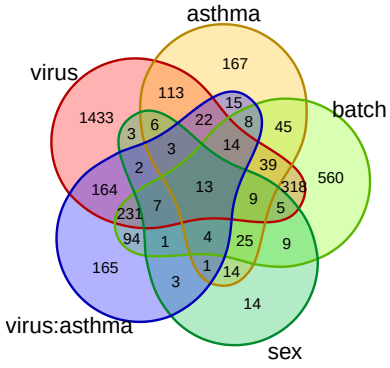
```
plot_venn_genes(model_result = virus_lme, model = "lme",
                fdr.cutoff = 0.001) +
plot_venn_genes(model_result = virus_batch_lme, model = "lme",
                fdr.cutoff = 0.001) +
plot_venn_genes(model_result = virus_batch_sex_lme, model = "lme",
                fdr.cutoff = 0.001)
```



FDR < 0.001



FDR < 0.001



FDR < 0.001

Importantly, some of the interaction genes (our main variable of interest) are significant for co-variates. This is further evidence to keep co-variates since they appear to play a role in the DEG we'll focus on in our results. The being said, the co-variates do not dramatically improve model fit and you could argue to remove them. So, this is a case where there is no clear right answer. As long as you understand the potential impacts on your results, you could continue your analyses with any of these 3 mixed effects models.

For simplicity, we'll move forward without any co-variates.

```
~ virus*asthma + (1|ptID)
```

3.3: Kinship co-variate

Kinship is a summative measure of genetic relatedness. It can be from 0 to 1 with 1 being 100% identical (monozygotic twins). Some other common values are 0.5 for parent-child, 0.25 grandparent-grandchild, 0.125 first cousins, etc. This measure is a pairwise measure with 1 value per pair of individuals.

```
kin <- kimma::example.kin

kin
```

```
##         donor1 donor2 donor3 donor4 donor5 donor6
## donor1    1.00   0.50   0.10   0.20   0.15   0.10
## donor2    0.50   1.00   0.10   0.11   0.23   0.09
## donor3    0.10   0.10   1.00   0.10   0.20   0.15
## donor4    0.20   0.11   0.10   1.00   0.11   0.13
## donor5    0.15   0.23   0.20   0.11   1.00   0.17
## donor6    0.10   0.09   0.15   0.13   0.17   1.00
```

Because it is not a single value per sample or individual, kinship cannot be added to a model with `+ kin`. Instead, it is used as a random effect where you block by individual so the model can identify an individual's kinship values relative to all other individuals in the data set.

`kimma` incorporates the `coxme` package's function `lmekin` to use kinship in linear mixed effects models. This feature is why `kimma` exists since pairwise kinship cannot be used in `limma`.

```
virus_kin <- kmFit(dat = dat, kin = kin,
                   model = "~ virus*asthma + (1|ptID)",
                   run.lmekin = TRUE)
```

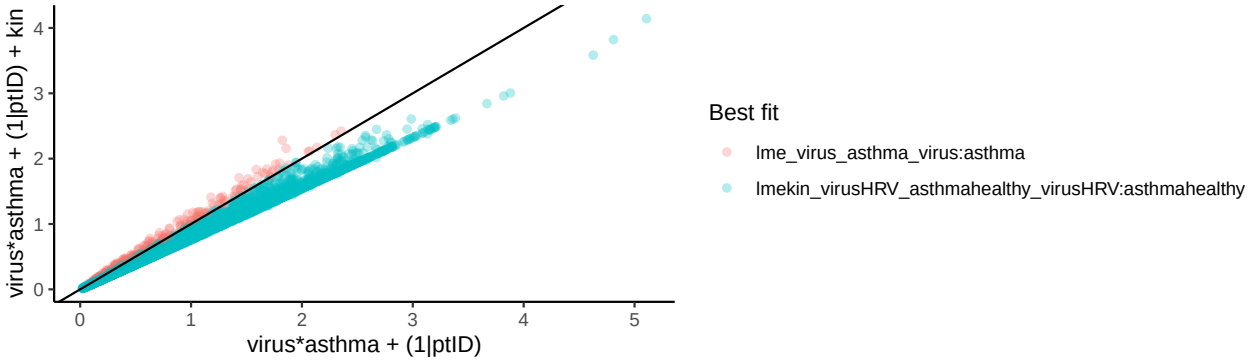
```
Running models on 5 individuals. 0 individuals missing kinship data.
lme/lmekin model: expression~virus*asthma+(1|ptID)
12830 genes complete
```

We see that kinship improves model fit for the majority of genes (> 95%) as well as increases the number of interaction significant genes.

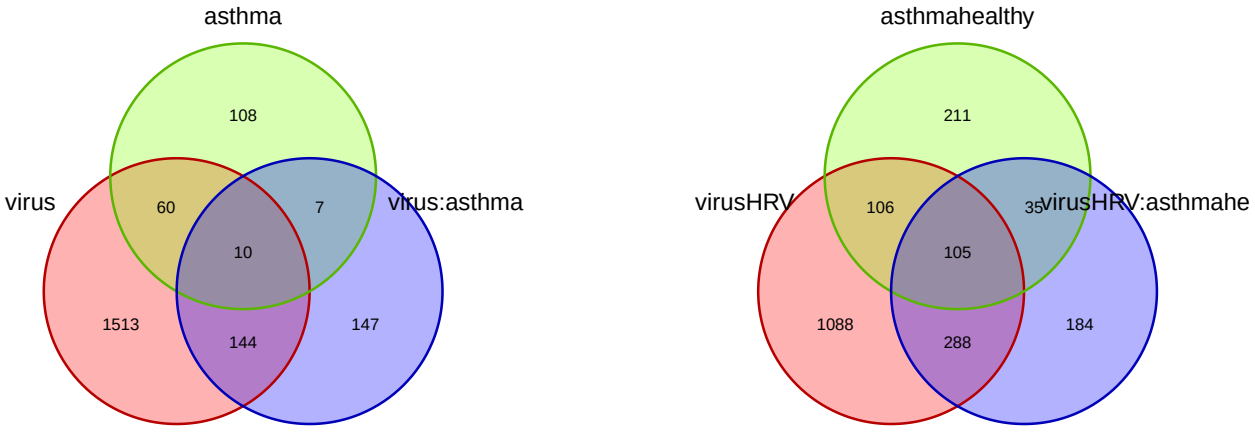
```
plot_sigma(model_result = virus_lme, model_result_y = virus_kin,
           x="lme", y="lmekin") +
  labs(x = "virus*asthma + (1|ptID)", y = "virus*asthma + (1|ptID) + kin")
```

```
## Total genes best fit by

##
##                          lme_virus_asthma_virus:asthma
##                                                    252
## lmekin_virusHRV_asthmahealthy_virusHRV:asthmahealthy
##                                                  12578
```




```
plot_venn_genes(model_result = virus_lme, model = "lme",
                fdr.cutoff = 0.001) +
plot_venn_genes(model_result = virus_kin, model = "lmekin",
                fdr.cutoff = 0.001)
```



FDR < 0.001 FDR < 0.001

Thus, we consider our final best fit model to be

`~ virus*asthma + (1|ptID) with kinship`

4. Post-hoc pairwise contrasts

If you have an interaction term in your model, you may wish to further probe the results to determine how the interaction is significant. Do healthy donors increase expression in response to virus while those with asthma show no change? Or does everyone decrease in expression but those with asthma decrease more? And other potential outcomes.

We will only run contrasts on genes that were significant for the interaction term. First, we select interaction DEG at FDR < 0.05.

```
subset.genes <- virus_kin$lmekin %>%
  filter(variable == "virusHRV:asthmahealthy" & FDR < 0.05) %>%
  distinct(gene) %>% unlist(use.names = FALSE)
```

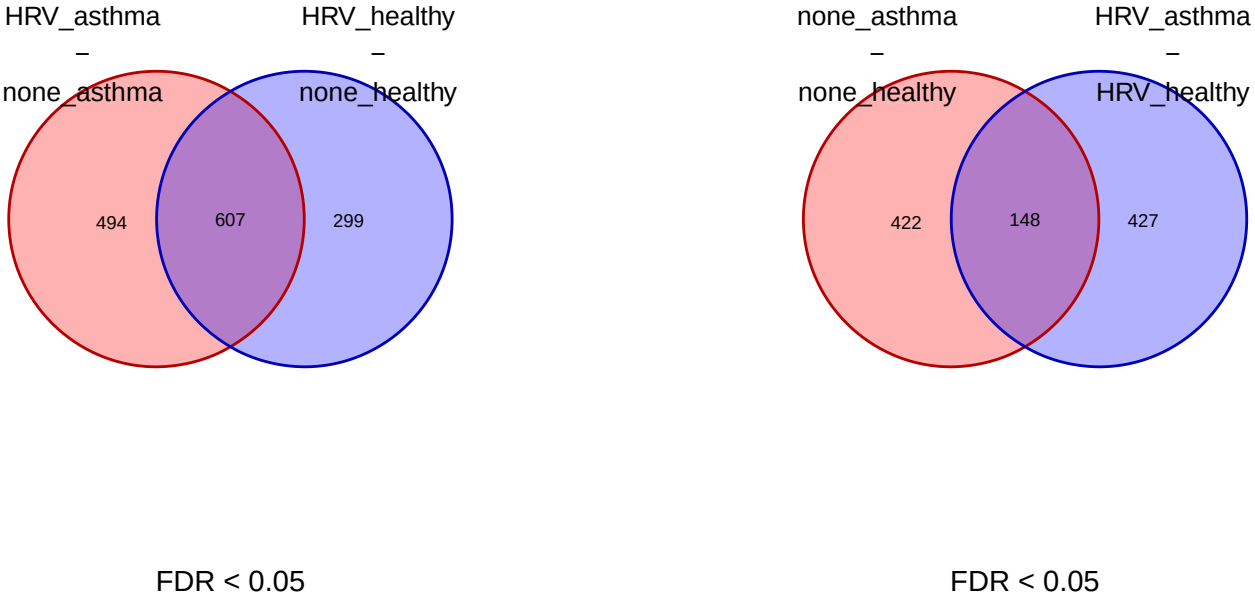
And then run the contrast model on these genes.

```
virus_contrast_kin <- kmFit(dat = dat, kin = kin,
                            model = "~ virus*asthma + (1|ptID)",
                            run.lmekin = TRUE, run.contrast = TRUE,
                            contrast.var = "virus:asthma",
                            subset.genes = subset.genes)
```

```
Running models on 5 individuals. 0 individuals missing kinship data.
lme/lmekin model: expression~virus*asthma+(1|ptID)
1403 genes complete
```

We see that a number of genes are only change significantly with virus in healthy or asthma individuals (left) and that some genes differ with asthma in the presence and/or absence of virus (right). The other two contrasts are not of interest in this analysis since they represent comparisons across both variables.

```
plot_venn_genes(model_result = virus_contrast_kin, model = "lmekin.contrast",
                fdr.cutoff = 0.05, contrasts=c("HRV_asthma - none_asthma",
                                               "HRV_healthy - none_healthy")) +
  plot_venn_genes(model_result = virus_contrast_kin, model = "lmekin.contrast",
                fdr.cutoff = 0.05, contrasts=c("none_asthma - none_healthy",
                                               "HRV_asthma - HRV_healthy"))
```



As noted in section 2.2, you may have instead chosen to use a contrast model for all genes and thus, would

not need this redundant post-hoc test.

5. Visualize DEG

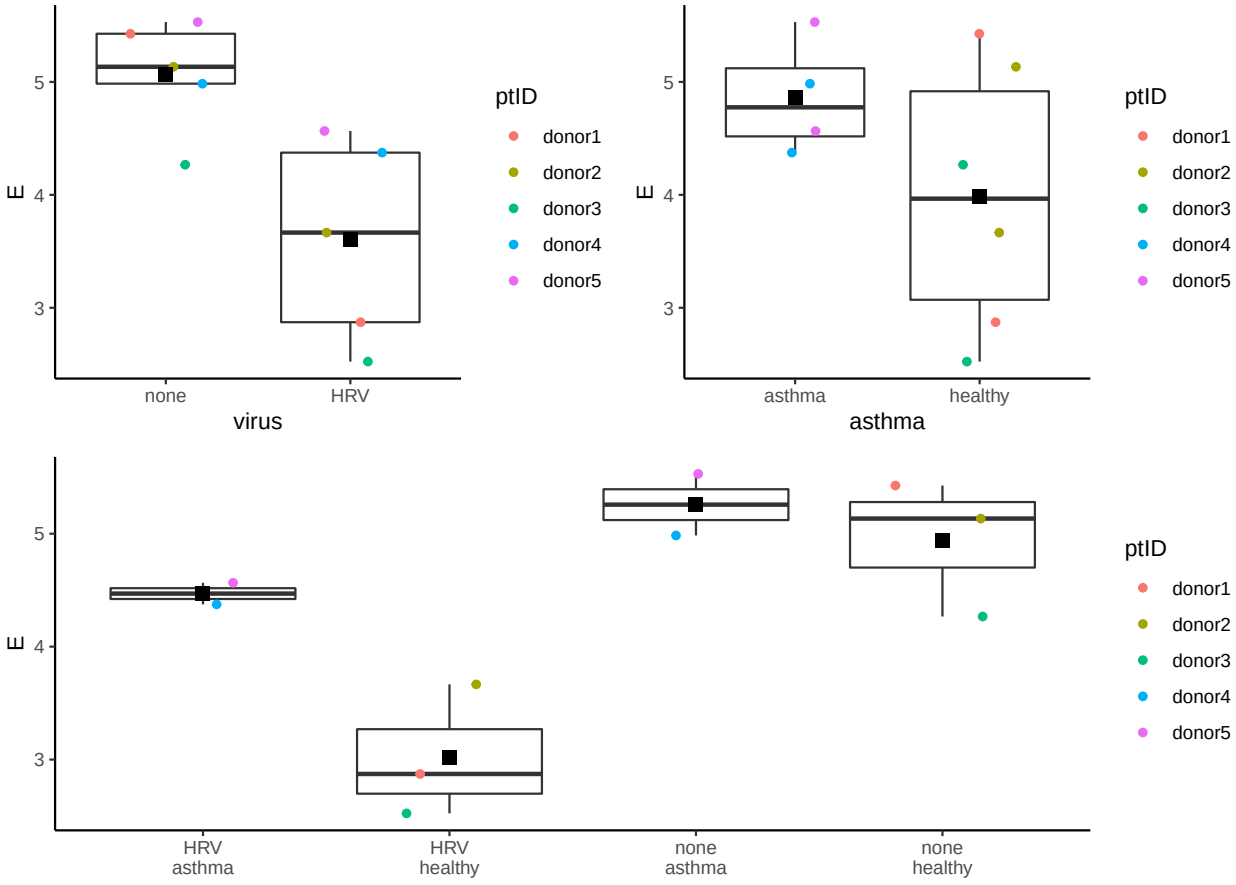
`BIGpicture` provides a function to plot expression for genes of interest across many variables. For example, we plot the first interaction significant gene.

```
plot_genes(dat = dat, fdr = virus_kin$lmekin, subset.genes=subset.genes[1],
           variables = "virus*asthma", geneID = "ensembl_gene_id",
           colorID="ptID")
```

```
## [[1]]
```

ENSG00000131584 ACAP3

nodel	gene	variable	estimate	pval	sigma	FDR
mekin	ENSG00000131584	(Intercept)	5.2484521	3.915336e−71	0.2616054	9.252858e−71
mekin	ENSG00000131584	virusHRV	−0.7876421	2.605583e−03	0.2616054	1.385522e−02
mekin	ENSG00000131584	asthmahealthy	−0.3710558	2.915657e−01	0.2616054	6.573165e−01
mekin	ENSG00000131584	virusHRV:asthmahealthy	−1.1342864	7.835274e−04	0.2616054	1.058174e−02

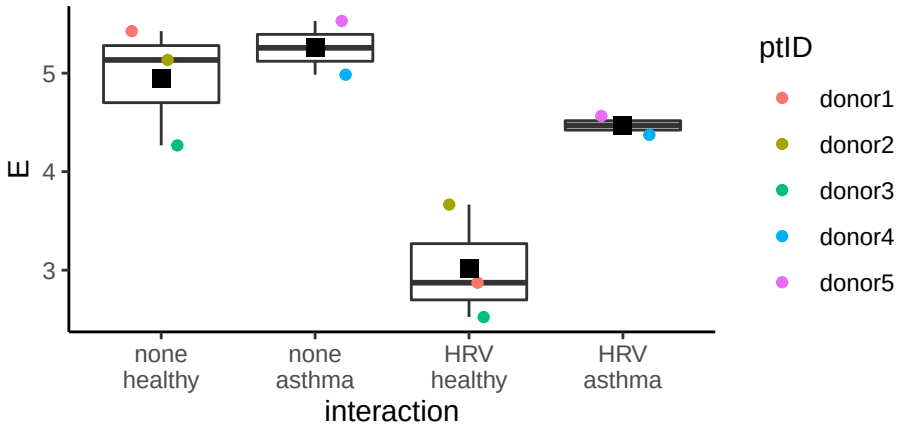


If you want custom ordering within variables, you need to create factors of the correct order.

```
dat$targets <- dat$targets %>%
  mutate(interaction = paste(virus, asthma, sep="\n"),
         interaction = factor(interaction, levels=c("none\nhealthy","none\nasthma",
                                                    "HRV\nhealthy","HRV\nasthma")))

plot_genes(dat = dat, subset.genes=subset.genes[1],
           variables = "interaction", geneID = "ensembl_gene_id",
           colorID="ptID")
```

```
## [[1]]
```



6. An actual analysis

This tutorial is meant to walk you through the majority of experimental designs you will encounter in RNAseq analysis. But you do not need to run all these models on an actual data set. For these example data, here is the actual pipeline I would take to select a best fit model.

1. What are the main variables of interest? Is an interaction necessary?
    - Yes -> I start with `~ virus*asthma`
2. Are the data from a paired sample design?
    - Yes -> I need a mixed effects model `~ virus*asthma + (1|ptID)`
3. Are there potential co-variates? Is one of them kinship?
    - Yes -> My most complicated model is `~ virus*asthma + batch + sex + (1|ptID) + kin`
4. Compare more complex models to the simplest version in #2
    - a. Add kinship `~ virus*asthma + (1|ptID) + kin`. I determine that it improves model fit enough to be included
    - b. Add co-variates `~ virus*asthma + batch + sex + (1|ptID) + kin`. I determine that sex should not be included but batch might.
    - c. Remove sex `~ virus*asthma + batch + (1|ptID) + kin`. I determine that batch does not improve the model fit enough to be included.
    - Note that the exclusion of co-variates was heavily influenced by the small data set size and a desire to use the simplest model possible
5. Run contrasts on interaction significant genes from best model (4a)

```
#4a
virus_kin <- kmFit(dat = dat, kin = kin,
                   model = "~ virus*asthma + (1|ptID)",
                   run.lme = TRUE, run.lmekin = TRUE)
```

```
plot_sigma(model_result = virus_kin, x="lme", y="lmekin")

# I don't both with venns since the improvement in fit is substantial

#4b
plot_pca(dat, vars = c("sex","batch")) %>%
  wrap_plots(ncol=2)

virus_kin_batch_sex <- kmFit(dat = dat, kin = kin,
                   model = "~ virus*asthma + batch + sex + (1|ptID)",
                   run.lmekin = TRUE)

plot_sigma(model_result = virus_kin, model_result_y = virus_kin_batch_sex,
           x="lmekin", y="lmekin")

plot_venn_genes(model_result = virus_kin, model = "lmekin",
                fdr.cutoff = c(0.05)) +
plot_venn_genes(model_result = virus_kin_batch_sex, model = "lmekin",
                fdr.cutoff = c(0.05))

#4c
virus_kin_batch <- kmFit(dat = dat, kin = kin,
                   model = "~ virus*asthma + batch + (1|ptID)",
                   run.lmekin = TRUE)

plot_sigma(model_result = virus_kin, model_result_y = virus_kin_batch,
           x="lmekin", y="lmekin")

plot_venn_genes(model_result = virus_kin, model = "lmekin",
                fdr.cutoff = c(0.05)) +
plot_venn_genes(model_result = virus_kin_batch, model = "lmekin",
                fdr.cutoff = c(0.05))

#5
subset.genes <- virus_kin$lmekin %>%
  filter(variable == "virusHRV:asthmahealthy" & FDR < 0.05) %>%
  distinct(gene) %>% unlist(use.names = FALSE)

contrast_model <- kmFit(dat = dat, kin = kin,
                        model = "~ virus*asthma + (1|ptID)",
                        run.lmekin = TRUE, run.contrast = TRUE,
                        contrast.var = "virus:asthma",
                        subset.genes = subset.genes)

plot_venn_genes(model_result = contrast_model, model = "lmekin.contrast",
                fdr.cutoff = c(0.05), contrasts=c("HRV_asthma - none_asthma",
                                                  "HRV_healthy - none_healthy"))
```

R session

```
sessionInfo()
```

```
## R version 4.1.1 (2021-08-10)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur 10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] DiagrammeR_1.0.6.1 patchwork_1.1.1    edgeR_3.34.1       limma_3.48.3
##  [5] BIGpicture_0.0.1   RNAetc_0.1.0       kimma_1.0.0        BIGverse_0.2.0
##  [9] forcats_0.5.1      stringr_1.4.0      dplyr_1.0.7        purrr_0.3.4
## [13] readr_2.0.2        tidyr_1.1.4        tibble_3.1.5       ggplot2_3.3.5
## [17] tidyverse_1.3.1
##
## loaded via a namespace (and not attached):
##  [1] nlme_3.1-153       fs_1.5.0           lubridate_1.8.0    doParallel_1.0.16
##  [5] webshot_0.5.2      RColorBrewer_1.1-2 httr_1.4.2         tools_4.1.1
##  [9] backports_1.2.1    utf8_1.2.2         R6_2.5.1           DBI_1.1.1
## [13] mgcv_1.8-38        colorspace_2.0-2   withr_2.4.2        tidyselect_1.1.1
## [17] processx_3.5.2     compiler_4.1.1     cli_3.0.1          rvest_1.0.2
## [21] xml2_1.3.2         labeling_0.4.2     scales_1.1.1       callr_3.7.0
## [25] digest_0.6.28      rmarkdown_2.11     pkgconfig_2.0.3    htmltools_0.5.2
## [29] dbplyr_2.1.1       fastmap_1.1.0      highr_0.9          htmlwidgets_1.5.4
## [33] rlang_0.4.12       readxl_1.3.1       rstudioapi_0.13    visNetwork_2.1.0
## [37] generics_0.1.0     farver_2.1.0       jsonlite_1.7.2     magrittr_2.0.1
## [41] Matrix_1.3-4       Rcpp_1.0.7         munsell_0.5.0      fansi_0.5.0
## [45] ggpolypath_0.1.0   lifecycle_1.0.1    stringi_1.7.5      yaml_2.2.1
## [49] grid_4.1.1         parallel_4.1.1     crayon_1.4.1       venn_1.10
## [53] lattice_0.20-45    cowplot_1.1.1      haven_2.4.3        splines_4.1.1
## [57] hms_1.1.1          locfit_1.5-9.4     knitr_1.36         ps_1.6.0
## [61] pillar_1.6.4       codetools_0.2-18   admisc_0.18        reprex_2.0.1
## [65] glue_1.4.2         evaluate_0.14      modelr_0.1.8       vctrs_0.3.8
## [69] tzdb_0.1.2         foreach_1.5.1      cellranger_1.1.0   gtable_0.3.0
## [73] assertthat_0.2.1   xfun_0.27          broom_0.7.9        iterators_1.0.13
## [77] statmod_1.4.36     ellipsis_0.3.2
```