variancePartition: Quantifying and interpreting drivers of variation in multilevel gene expression experiments

Gabriel Hoffman

Icahn Institute for Genomics and Multiscale Biology
Department of Genetics and Genomic Sciences
Icahn School of Medicine at Mount Sinai

October 15, 2015

Abstract

Gene expression datasets are complicated and have multiple sources of biological and technical variation. These datasets have recently become more complex as it is now feasible to assay gene expression from the same individual in multiple tissues or at multiple time points. The *variancePartition* package implements a statistical method to quantify the contribution of multiple sources of variation and decouple within/between-individual variation. In addition, *variancePartition* produces results at the gene-level to identity genes that follow or deviate from the genome-wide trend.

variancePartition version: 1.0.0

Contents

1 Overview

The *variancePartition* provides a general framework for understanding drivers of variation in gene expression in experiments with complex designs. A typical application would consider a dataset of gene expression from individuals sampled in multiple tissues or multiple time points where the goal is to understand variation within versus between individuals and tissues. *variancePartition* use a linear mixed model to partition the variance attributable to multiple variables in the data. The analysis is built on top of the *lme4* package [1], and some basic knowledge about linear mixed models will give you some intution about the behavior of *variancePartition* [2, 3]

There are two components to an analysis: 1) gene expression data, 2) meta-data about each sample such as patient ID, tissue, sex, disease state, etc. *variancePartition* will assess the contribution of each meta-data variable to variation in gene expression and can report the intra-class correlation for each variable.

2 Running an analysis

A typical analysis with *variancePartition* is only a few lines of *R* code, assuming the expression data has already been normalized. Normalization is a separate topic, and I address it briefly in Section 5.

2.1 Standard

```
# load library
library(variancePartition)

# optional step to run analysis in parallel on multicore machines
# Here use 4 threads
# This is strongly recommended since the analysis
# can be computationally intensive
library(doParallel)
cl <- makeCluster(4)
registerDoParallel(cl)

# load simulated data:
# geneExpr: matrix of gene expression values
# info: information/metadata about each sample
data(varPartData)

# Specify variables to consider
# Age is continuous so model it as a fixed effect
# Individual and Tissue are both categorical, so model them as random effects
# Note the syntax used to specify random effects
form = ~ Age + (1|Individual) + (1|Tissue)

# Fit model and extract results
# 1) fit linear mixed model on gene expresson
# If categorical variables are specified, a linear mixed model is used
# If all variables are modeled as fixed effects, a linear model is used
# each entry in results is a regression model fit on a single gene
# 2) extract variance fractions from each model fit
# for each gene, returns fraction of variation attributable to each variable
# Interpretation: the variance explained by each variables after correcting
# for all other variables
# Note that geneExpr can either be a matrix,
# or the output of voom() in the limma package
varPart = fitExtractVarPartModel( geneExpr, form, info )

# Figure 1a
```

```
# violin plot of contribution of each variable to total variance
# sort variables by median fraction of variance explained
plotVarPart( sortCols(varPart) )
```

The core functions of *variancePartition* work seemlessly with gene expression data stored as a `matrix`, `data.frame`, `EList` from *limma* or `ExpressionSet` from *Biobase*.

2.2 Advanced

Advanced users may want to perform the model fit and extract results in separate steps so you can examine the fit of the model for each gene. Note that storing the model fits can use a lot of memory (~10Gb with 20K genes and 1000 experiments). I recommend using the one step `fitExtractVarPart` unless you have a specific need for the two step approach:

```
# Fit model
results = fitVarPartModel( geneExpr, form, info )

# Extract results
varPart = extractVarPart( results )
```

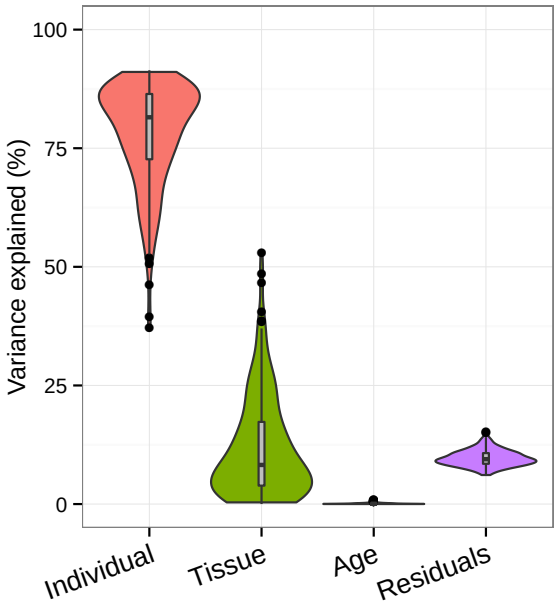


Figure 1: *variancePartition* example on simulated data

2.3 Plot expression stratified by other variables

Users can also plot a gene expression trait stratified by Tissue or Individual.

```
# get gene with the highest variation across Tissues
# create data.frame with expression of gene i and Tissue type for each sample
i = which.max( varPart$Tissue )
GE = data.frame( Expression = geneExpr[i,], Tissue = info$Tissue)

# Figure 2a
# plot expression statified by Tissue
plotStratifyBy( GE, "Tissue", "Expression", main=rownames(geneExpr)[i])
#
# get gene with the highest variation across Individuals
# create data.frame with expression of gene i and Tissue type for each sample
i = which.max( varPart$Individual )
GE = data.frame( Expression = geneExpr[i,], Individual = info$Individual)

# Figure 2b
# plot expression statified by Tissue
label = paste("Individual:", format(varPart$Individual[i]*100, digits=3), "%")
main = rownames(geneExpr)[i]
plotStratifyBy( GE, "Individual", "Expression", colorBy=NULL, text=label, main=main)
```

(a) Tissue (b) Individual

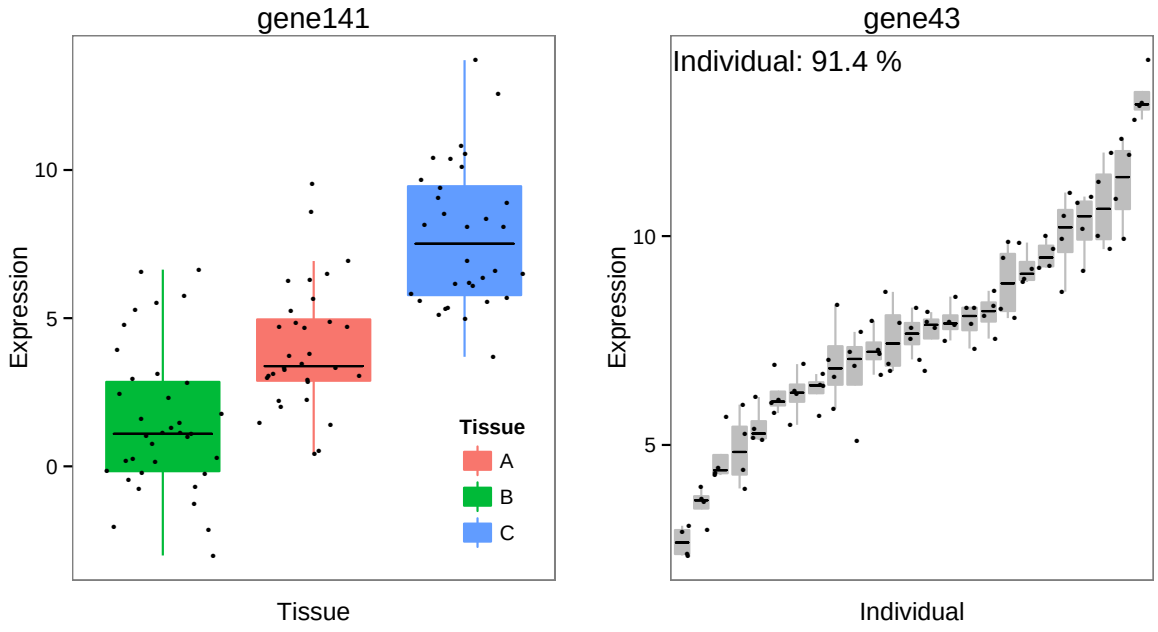


Figure 2: Plot gene expression statified by **a)** Tissue and **b)** Individual

For gene141, variation across tissues explains 52.9% of variance in gene expression. For gene43, variation across Individuals explains 91.4% of variance in gene expression.

3 Interpretation

variancePartition fits a linear (mixed) model thats jointly considers the contribution of all specified variables on the expression of each gene. It is similar to multiple regression in that the effect of each variable is assessed while correcting for all others. Therefore, fitting the model with each variable separtely will give very different results from the considering all variables jointly. I have found joint analysis the best option in most cases.

The results of *variancePartition* give insight into the expression data at multiple levels. Moreover, a single statistic often has multiple equivalent interpretations while only one is relevant to the biological question. Here I give a description of the data from the previous section based on the results in Figure 1a, but these conclusions are applicable to standard analyses with *variancePartition*.

Averaging across all genes,

1) variation between individuals explains 78% of the variation in expression, after correcting for tissue and age
2) variation between tissues explains 12% of the variation in expression, after correcting for individual and age
3) after correcting for individual and tissue, the effect of age is negligible.
4) correcting for individual, tissue and age leaves 10% of the total variance in expression.

These statistics also have a natural interpretation in terms of the intra-class correlation (ICC), the correlation between observations made from samples in the same group.

On average across all genes and all experiments,

1) the ICC for individual is 78%.
2) the ICC for tissue is 12%.
3) two randomly selected gene measurements from same individual, but regardless of tissue or age, have a correlation of 78%.
4) two randomly selected gene measurements from same tissue, but regardless of individual or age, have a correlation of 12%.
5) two randomly selected gene measurements from the same individual *and* same tissue, but regardless of age, have an correlation of $78\% + 12\% = 90\%$.

Note that that the ICC here is interpretated as the ICC after correcting for all other variables in the model.

These conclusions have been averaged across all genes, but the real power of *variancePartition* is to identify specific genes that follow or deviate from the genome-wide trend. The gene-level statistics can be used to identify a subset of genes that are enriched for specific biological functions. For example, we can ask if the 500 genes with the highest variation in expression across tissues (i.e. the long tail for tissue in Figure 1a) are enriched for genes known to have high tissue-specificity.

3.1 Should a variable be modeled as fixed or random effect?

Catagorical variables should (almost) always be modeled as a random effect. The difference between modeling a catagorical variable as a fixed versus random effect is minimal when the sample size is large compared to the number of categories (i.e. levels). So variables like disease status, sex or time point will not be sensitive to modeling as a fixed versus random effect. However, variables with many categories like Individual *must* be modeled as a random effect in order to obtain statistically valid results. So to be on the safe side, categorical variable should be modeled as a random effect.

variancePartition fits two types of models:

1) linear mixed model where all categorical variables are modeled as random effects and all continuous variables are fixed effects. The function `lmer` from *lme4* is used to fit this model.
2) fixed effected model, where all variables are modeled as fixed effects. The function `lm` is used to fit this model.

3.2 Which variables should be included?

In my experience, it is useful to include all variables in the first analysis and then drop variables that have minimal effect. However, like all multiple regression methods, *variancePartition* will divide the contribution over multiple variables that are strongly correlated. So, for example, including both sex and height in the model will show sex having a smaller contribution to variation gene expression than if height were omitted, since there variables are strongly correlated. This is a simple example, but should give some intuition about a common issue that arises in analyses with *variancePartition*.

variancePartition can naturally assess the contribution of both individual and sex in a dataset. As expected, genes for which sex explains a large fraction of variation are located on chrX and chrY. If the goal is to interpret the impact of sex, then there is no issue. But recall the issue with correlated variables and note that individual is correlated with sex, because each individual is only one sex regardless of how many samples are taken from a individual. It follows that including sex in the model reduces the apparent contribution of individual to gene expression. In other words, the ICC for individual will be different if sex is included in the model.

In general, including variables in the model that do not vary within individual will reduce the apparent contribution of individual as estimated by *variancePartition*. For example, sex and ethnicity never vary between multiple samples from the same individual and will always reduce the apparent contribution of individual. However, disease state and age may or may not vary depending on the study design.

3.3 Detecting problems caused by colinearity of variables

Including variables that are highly correlated can produce misleading results and overestimate the contribution of variables modeled as fixed effects. This is usually not an issue, but can arise when statistically redundant variables are included in the model. In this case, the model is "degenerate" or "computationally singular" and parameter estimates from this model are not meaningful. Dropping one or more of the covariates will fix this problem.

A check of colinearity is built into `fitVarPartModel` and `fitExtractVarPartModel`, so the user will be warned if this is an issue.

Alternatively, the user can use the `colinearityScore` function to evaluate whether this a an issue for a single model fit:

```
form = ~ (1|Individual) + (1|Tissue) + Age + Height

# fit model
res = fitVarPartModel( geneExpr[1:4,], form, info )

# evaluate the colinearity score on the first model fit
# this reports the correlation matrix between coefficients estimates
# for fixed effects
# the colinearity score is the maximum absolute correlation value
# If the colinearity score > .99 then the variance parition
# estimates may be problematic
# In that case, a least one variable should be omitted
colinearityScore(res[[1]])
```

```
## [1] 0.777
## attr(,"vcor")
## 3 x 3 Matrix of class "dpoMatrix"
##             (Intercept)     Age  Height
## (Intercept)       1.000 -0.4191 -0.7774
## Age              -0.419  1.0000 -0.0575
## Height           -0.777 -0.0575  1.0000
```

4 Advanced analysis

4.1 Removing batch effects before fitting model

Gene expression studies often have substantial batch effects, and *variancePartition* can be used to understand the magnitude of the effects. However, we often want to focus on biological variables (i.e. individual, tissue, disease, sex) after removing the effect of technical variables. Depending on the size of the batch effect, I have found it useful to correct for the batch effect first and then perform a *variancePartition* analysis afterward. Subtracting this batch effect can reduce the total variation in the data, so that the contribution of other variables become clearer.

Standard analysis:

```
form = ~ (1|Tissue) + (1|Individual) + (1|Batch) + Age
varPart = fitExtractVarPartModel( geneExpr, form, info )
```

Analysis on residuals:

```
library(limma)
# subtract out effect of Batch
fit = lmFit( geneExpr, model.matrix(~ Batch, info))
res = residuals( fit, geneExpr)

# fit model on residuals
form = ~ (1|Tissue) + (1|Individual) + Age

varPartResid = fitExtractVarPartModel( res, form, info )
```

Remove batch effect with linear mixed model

```
# subtract out effect of Batch with linear mixed model
modelFit = fitVarPartModel( geneExpr, ~ (1|Batch), info )
res = residuals( modelFit )

# fit model on residuals
form = ~ (1|Tissue) + (1|Individual) + Age

varPartResid = fitExtractVarPartModel( res, form, info )
```

4.2 Variation within multiple subsets of the data

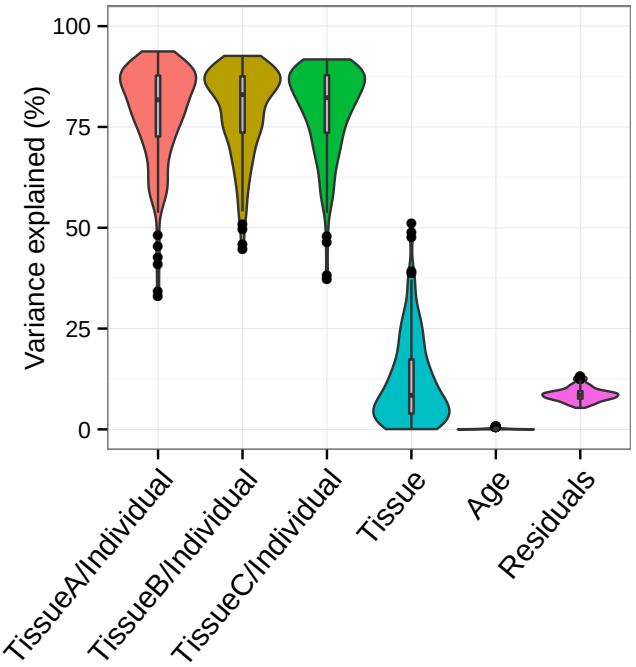
So far, we have focused on interpreting one variable at a time. But the linear mixed model behind *variancePartition* is a very powerful framework for analyzing variation at multiple levels. We can easily extend the previous analysis of the contribution of individual and tissue on variation in gene expression to examine the contribution of individual *within* each tissue. This analysis is as easy as specifying a new formula and rerunning *variancePartition*. Note that is analysis will only work when each individual is observed in each tissue and when there are multiple replicates within each tissue.

```
# specify formula to model within/between individual variance
# separately for each tissue
# Note that including +0 ensures each tissue is modeled explicitly
# Otherwise, the first tissue would be used as baseline
form = ~ (Tissue+0|Individual) + Age + (1|Tissue)

# fit model and extract variance percents
varPart = fitExtractVarPartModel( geneExpr, form, info )

# violin plot
plotVarPart( varPart, label.angle=50 )
```

Figure 3: Variation across individuals within each tissue



This analysis correspnds to a varying coefficient model, where the correlation between individuals varies for each tissue [2]. Since the variation across individuals is modeled within each tissue, the total

variation explained does not sum to 1 as it does for standard analysis with *variancePartition*. However, the interpretation in terms of the intra-class correlation still applies.

See Section 7.3 for statistical details.

5 Normalizing RNA-Seq data

variancePartition works with gene expression data that has already been normalized. Expression should be normalized in exactly the same way as for differential expression analysis. Here I discuss normalization specifically for RNA-Seq data, while the process for microarray data may depend on the platform and manufacturer

Read RNA-Seq counts into *R*, normalize for library size within and between experiments with TMM [4], estimates precision weights with limma/voom [5]

```
## not run
library(limma)
library(edgeR)

# identify genes that pass expression cutoff
isexpr = rowSums(cpm(geneCounts)>.1) >= 0.5 * ncol(geneCounts)

# create data structure with only expressed genes
geneExpr = DGEList(counts=geneCounts[isexpr,])

# Perform TMM normalization
geneExpr = calcNormFactors(geneExpr)

# Specify variables to be included in the voom() estimates of uncertainty
# recommand including variables with a small number of categories
# that explain a substantial amount of variation
design = model.matrix( ~ Batch, info)

# Estimate precision weights for each gene and sample
# This models uncertainty in expression measurements
vobjGenes = voom(geneExpr, design )

# fit variancePartition
form = ~ (1|Individual) + (1|Tissue) + Age

# variancePartition seamlessly deals with the result of voom()
# by default, it seamlessly models the precision weights
# This can be turned off with useWeights=FALSE
varPart = fitExtractVarPartModel( vobjGenes, form, info )
```

6 Comparison with other methods on simulated data

Characterizing drivers of variation in gene expression data has typically relied on principal components analysis (PCA) and hierarchical clustering. Here I apply these methods to two simulated datasets to demonstrate the additional insight from an analysis with *variancePartition*. Each simulated dataset comprises 60 experiments from 10 individuals and 3 tissues wih 2 biological replicates. In the first dataset, tissue is the major driver of variation in gene expression(Figure 4). In the second dataset, individual is the major driver of variation in gene expression (Figures 5).

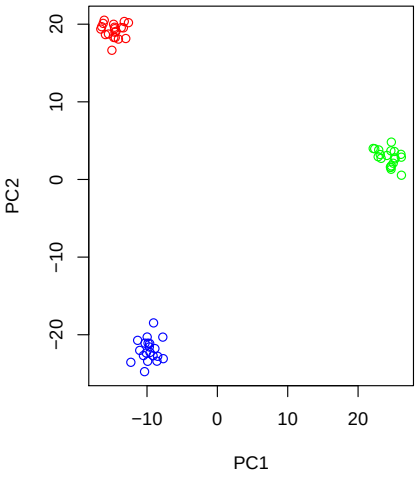
Analysis of simulated data illustrates that PCA identifies the major driver of variation when tissue is dominant and there are only 3 catagories. But the results are less clear when individual is dominant because there are now 10 categories. Meanwhile, hierarchical clustering identifies the major driver of variation in both cases, but does not give insight into the second leading contributor.

Analysis with *variancePartition* has a number of advantages over these standard methods:

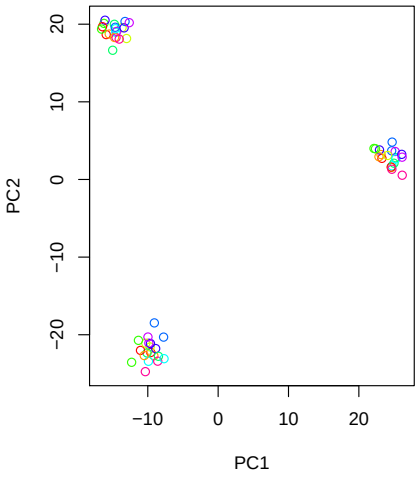
- *variancePartition* provides a natural interpretation of multiple variables
  - figures from PCA/hierarchical clustering allow easy interpretation of only one variable
- *variancePartition* quantifies the contribution of each variable
  - PCA/hierarchical clustering give only a visual representation
- *variancePartition* interprets contribution of each variable to each gene individually for downstream analysis
  - PCA/hierarchical clustering produces genome-wide summary and does not allow gene-level interpretation
- *variancePartition* can assess contribution of one variable (i.e. Individual) separately in subset of the data defined by another variable (i.e. Tissue)

Figure 4: **Similarity within Tissue is dominant**

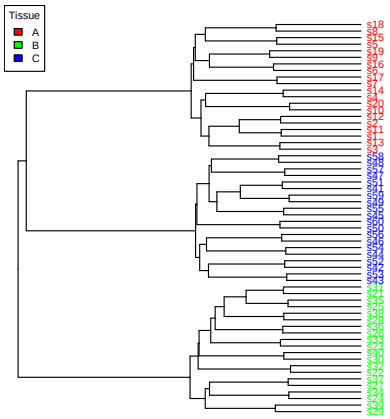
(a) PCA - colored by Tissue



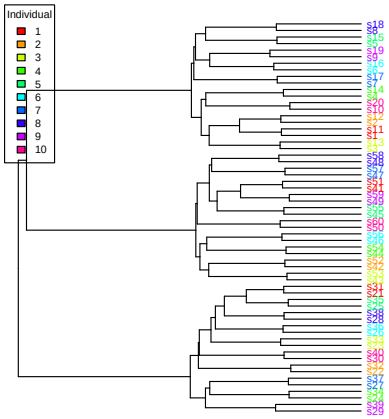
(b) PCA - colored by Individual



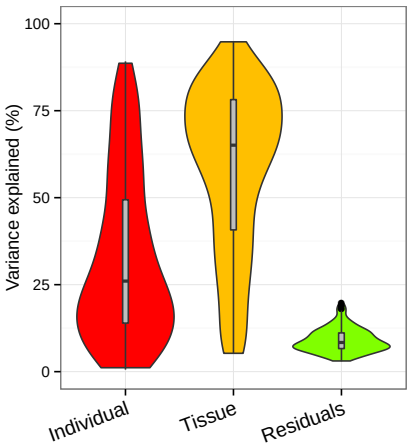
(c) hclust - colored by Tissue



(d) hclust - colored by Individual



(e) variancePartition



(f) variancePartition - within Tissue

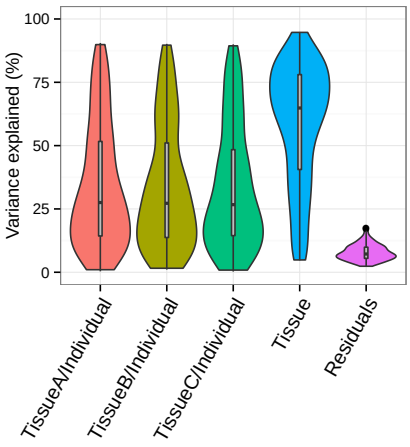
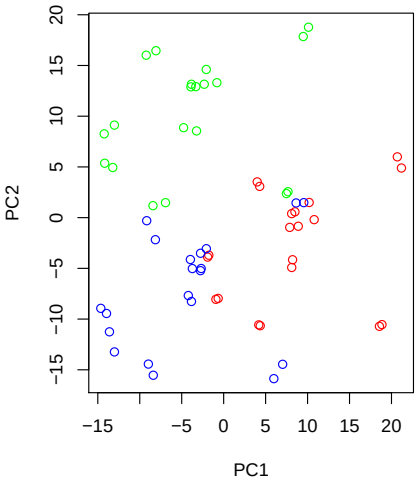
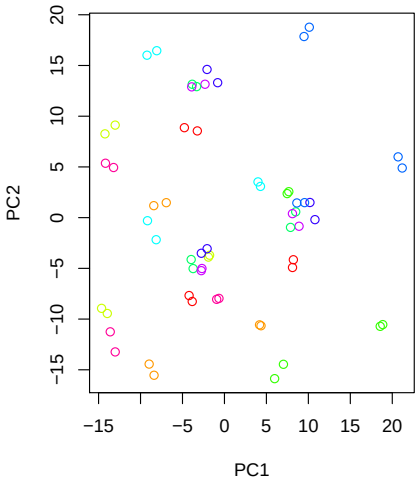


Figure 5: **Similarity within Individual is dominant**

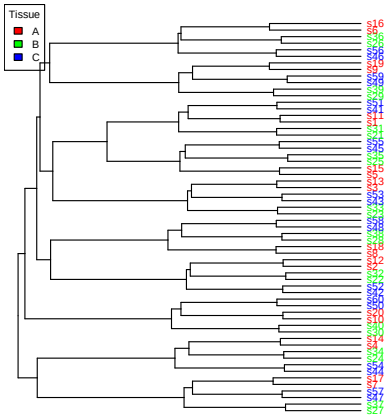
(a) PCA - colored by Tissue



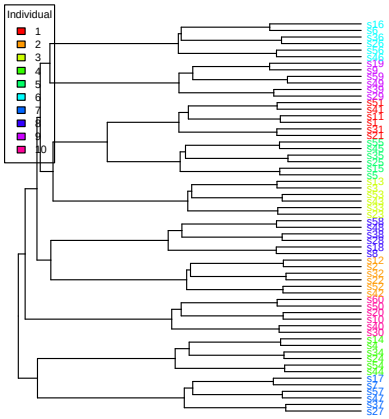
(b) PCA - colored by Individual



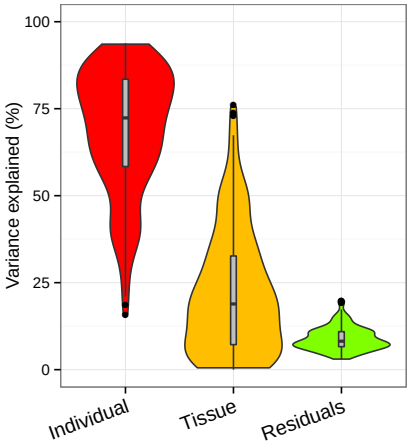
(c) hclust - colored by Tissue



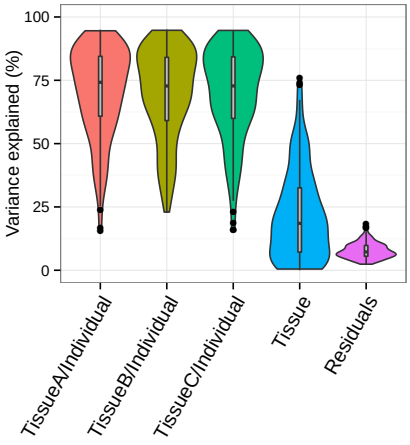
(d) hclust - colored by Individual



(e) variancePartition



(f) variancePartition - within Tissue



7 Statistical details

A *variancePartition* analysis evaluates the linear (mixed) model

$$y = \sum_j X_j \beta_j + \sum_k Z_k \alpha_k + \varepsilon \quad (1)$$

$$\alpha_k \sim \mathcal{N}(0, \sigma^2_{\alpha_k}) \quad (2)$$

$$\varepsilon \sim \mathcal{N}(0, \sigma^2_{\varepsilon}) \quad (3)$$

where y is the expression of a single gene across all samples, X_j is the matrix of j^{th} fixed effect with coefficients β_j, Z_k is the matrix corresponding to the k^{th} random effect with coefficients α_k drawn from a normal distribution with variance $\sigma^2_{\alpha_k}$. The noise term, ε, is drawn from a normal distribution with variance σ^2_{ε}. Parameters are estimated with maximum likelihood, rather than REML, so that fixed effect coefficients, β_j, are explicitly estimated.

I use the term "linear (mixed) model" here since *variancePartition* works seamlessly when a fixed effects model (i.e. linear model) is specified.

Variance terms for the fixed effects are computed using the *post hoc* calculation

$$\hat{\sigma}^2_{\beta_j} = var(X_j \hat{\beta}_j). \quad (4)$$

For a fixed effects model, this corresponds to the sum of squares for each component of the model.

For a standard application of the linear mixed model, where the effect of each variable is additive, the fraction of variance explained by the j^{th} fixed effect is

$$\frac{\hat{\sigma}^2_{\beta_j}}{\sum_j \hat{\sigma}^2_{\beta_j} + \sum_k \hat{\sigma}^2_{\alpha_k} + \hat{\sigma}^2_{\varepsilon}}, \quad (5)$$

by the k^{th} random effect is

$$\frac{\hat{\sigma}^2_{\alpha_k}}{\sum_j \hat{\sigma}^2_{\beta_j} + \sum_k \hat{\sigma}^2_{\alpha_k} + \hat{\sigma}^2_{\varepsilon}}, \quad (6)$$

and the residual variance is

$$\frac{\hat{\sigma}^2_{\varepsilon}}{\sum_j \hat{\sigma}^2_{\beta_j} + \sum_k \hat{\sigma}^2_{\alpha_k} + \hat{\sigma}^2_{\varepsilon}}. \quad (7)$$

7.1 Implementation in R

An *R* formula is used to define the terms in the fixed and random effects, and `fitVarPartModel` fits the specified model for each gene separately. If random effects are specified, `lmer` from *lme4* is used behind the scenes to fit the model, while `lm` is used if there are only fixed effects. `fitVarPartModel` returns a list of the model fits, and `extractVarPart` returns the variance partition statistics for each model in the list. `fitExtractVarPartModel` combines the actions of `fitVarPartModel` and `extractVarPart` into one function call. `calcVarPart` behind the scenes to compuate variance fractions for both fixed and mixed effects models.

7.2 Interpretation of percent variance explained

The percent variance explained can be interpreted as the intra-class correlation (ICC) when a special case of Equation 1 is used. Consider the simplest example of the i^{th} sample from the k^{th} individual

$$y_{i,k} = \mu + Z\alpha_{i,k} + e_i \tag{8}$$

with only an intercept term, one random effect corresponding to individual, and an error term. In this case ICC corresponds to the correlation between two samples from the same individual. This value is equal to the fraction of variance explained by individual. For example, consider the correlation between samples from the same individual:

$$\begin{aligned}
ICC &= cor(y_{1,k}, y_{2,k}) && (9)\\
&= cor(\mu + Z\alpha_{1,k} + e_{1,k}, \mu + Z\alpha_{2,k} + e_{2,k}) && (10)\\
&= \frac{cov(\mu + Z\alpha_{1,k} + e_{1,k}, \mu + Z\alpha_{2,k} + e_{2,k})}{\sqrt{var(\mu + Z\alpha_{1,k} + e_{1,k}) var(\mu + Z\alpha_{2,k} + e_{2,k})}} && (11)\\
&= \frac{cov(Z\alpha_{1,k}, Z\alpha_{2,k})}{\sigma^2_\alpha + \sigma^2_\varepsilon} && (12)\\
&= \frac{\sigma^2_\alpha}{\sigma^2_\alpha + \sigma^2_\varepsilon} && (13)
\end{aligned}$$

The correlation between samples from different individuals is:

$$\begin{aligned}
&= cor(y_{1,1}, y_{1,2}) && (14)\\
&= cor(\mu + Z\alpha_{1,1} + e_{1,1}, \mu + Z\alpha_{1,2} + e_{1,2}) && (15)\\
&= \frac{cov(Z\alpha_{1,1}, Z\alpha_{1,2})}{\sigma^2_\alpha + \sigma^2_\varepsilon} && (16)\\
&= \frac{0}{\sigma^2_\alpha + \sigma^2_\varepsilon} && (17)\\
&= 0 && (18)
\end{aligned}$$

This interpretation in terms of fraction of variation explained (FVE) naturally generalizes to multiple variance components. Consider two sources of variation, individual and cell type with variances σ^2_{id} and σ^2_{cell}, respectively. Applying a generalization of the the previous derivation, two samples are correlated according to:

Individual	cell type	variance	Interpretation	Correlation value
same	different	$\frac{\sigma^2_{id}}{\sigma^2_{id}+\sigma^2_{cell}+\sigma^2_\varepsilon}$	FVE by individual	$ICC_{individual}$
different	same	$\frac{\sigma^2_{cell}}{\sigma^2_{id}+\sigma^2_{cell}+\sigma^2_\varepsilon}$	FVE by cell type	ICC_{cell}
same	same	$\frac{\sigma^2_{id}+\sigma^2_{cell}}{\sigma^2_{id}+\sigma^2_{cell}+\sigma^2_\varepsilon}$	sum of FVE by individual & cell type	$ICC_{individual,cell}$
different	different	$\frac{0}{\sigma^2_{id}+\sigma^2_{cell}+\sigma^2_\varepsilon}$	sample are independent	

Notice that the correlation between samples from the same individual and same cell type corresponds to the sum of the fraction explained by individual + fraction explained by cell type. This defines ICC for individual and tissue, as well as the combined ICC and relates these values to FVE.

In order to illustrate how this FVE and ICC relate to the correlation between samples in multilevel datasets, consider a simple example of 5 samples from 2 individuals and 2 tissues:

Sample	Individual	Cell type
a	1	T-Cell
b	1	T-Cell
c	1	monocyte
d	2	T-Cell
e	2	monocyte

Modeling the separate effects of individual and tissue gives the following covariace structure between samples when a linear mixed model is used:

$$cov(y) = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{pmatrix} \sigma^2_{id} + \sigma^2_{cell} + \sigma^2_\varepsilon & & & & \\ \sigma^2_{id} + \sigma^2_{cell} & \sigma^2_{id} + \sigma^2_{cell} + \sigma^2_\varepsilon & & & \\ \sigma^2_{id} & \sigma^2_{id} & \sigma^2_{id} + \sigma^2_{cell} + \sigma^2_\varepsilon & & \\ \sigma^2_{cell} & \sigma^2_{cell} & 0 & \sigma^2_{id} + \sigma^2_{cell} + \sigma^2_\varepsilon & \\ 0 & 0 & 0 & \sigma^2_{id} & \sigma^2_{id} + \sigma^2_{cell} + \sigma^2_\varepsilon \end{pmatrix} \end{array}$$

The covariance matrix is symmetric so that blank entries take the value on the opposite side of the diagonal. The covariance can be converted to correlation by dividing by $\sigma^2_{id} + \sigma^2_{cell} + \sigma^2_\varepsilon$, and this gives the results from above. This example generalizes to any number of variance components [2].

7.3 Variation with multiple subsets of the data

The linear mixed model underlying *variancePartition* allows the effect of one variable to depend on the value of another variable. Statistically, this is called a varying coefficient model [2, 3]. This model arises in *variancePartition* analysis when the variation explained by individual depends on tissue or cell type.

A given sample is only from one cell type, so this analysis asks a question about a subset of the

data. The the data is implicitly divided into subsets base on cell type and variation explained by individual is evaluated within each subset. The data isn't actually divided onto subset, but the statistical model essentially examples samples with each cell type. This subsetting means that the variance fractions don't sum to 1.

Consider a concrete example with variation from across individual and cell types (T-cells and monocytes). Modeling the variation across individuals within cell type corresponds to

$$y_{i,s,c} = \mu + Z^{(sex)}\alpha_{i,s} + Z^{(T-cell|id)}\alpha_{i,c} + Z^{(monocyte|id)}\alpha_{i,c} + e_{i,s,c} \tag{19}$$

with corresponding variance components

Variance component	Interpretation
σ^2_{sex}	variance across sex
$\sigma^2_{(T-cell\|id)}$	variance across individuals within T-cells
$\sigma^2_{(monocyte\|id)}$	variance across individuals within monocytes
σ^2_{ε}	residual variance

7.4 Relationship between variancePartition and differential expression

Differential expression (DE) is widely used to identify gene which show difference is expression between two subsets of the data (i.e. case versus controls). For a single gene, DE analysis measures the difference in mean expression between the two subsets. (Since expression is usually analyzed on a log scale, DE results are show in terms of log fold changes between the two subsets). In Figure 6, consider two simulated examples of a gene whose expression differs between males and females. The mean expression in males is 0 and the mean expression in females is 2 in both cases. Therefore, the fold change is 2 in both cases.

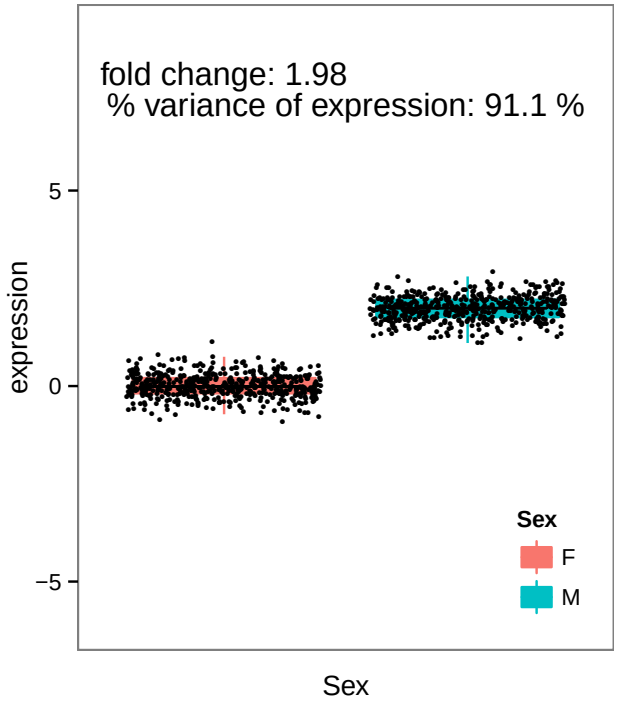
However, the fraction of expression variation explained by sex is very different in these two examples. In example A, there is very little variation *within* each sex, so that variation *between* sexes is very high at 91.1%. Conversely, Example B show high variation *within* sexes, so that variation *between* sexes is only 19.4%.

Fact that the fold change or the fraction of variation is significantly different from 0 indicates differential expression between the two sexes. Yet these two statistics have different interpretations. The fold change from DE analysis tests a difference in means between two sexes. The fraction of variation explained compares the variation explained by sex to the total variation.

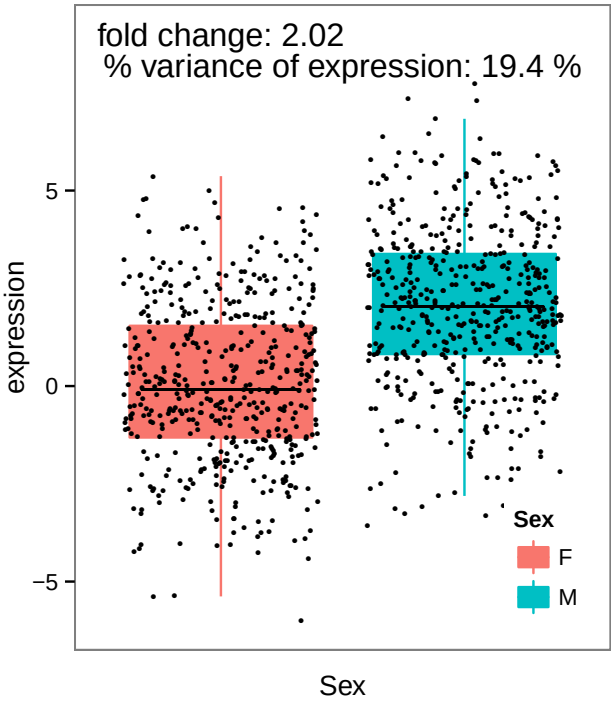
Thus the fraction of variation explained reported by *variancePartition* reflects as different aspect of the data not captured by DE analysis.

Figure 6: **Compare variancePartition and differential expression**

(a) Example A



(b) Example B



7.5 Modelling error in gene expression measurements

Uncertainty in the measurement of gene expression can be modeled with precision weights and tests of differentially expression using `voom` in *limma* model this uncertainty directly with a heteroskadastic linear regression [5]. *variancePartition* can use these precision weights in a heteroskadastic linear mixed model implemented in *lme4* [1]. These precision weights are used seamlessly by calling `fitVarPartModel` or `fitExtractVarPartModel` on the output of `voom`. Otherwise the user can specify the weights with the `weightsMatrix` parameter.

8 Frequently asked questions

8.1 Warnings and errors

Interpreting warnings and errors from `fitVarPartModel` and `fitExtractVarPartModel`:

- `Colinear score > .99: Covariates in the formula are so strongly correlated that the parameter estimates from this model are not meaningful. Dropping one or more of the covariates will fix this problem`
- `Error in asMethod(object) : not a positive definite matrix`
- `In vcov.merMod(fit) : Computed variance-covariance matrix problem: not a positive definite matrix; returning NA matrix`
- `fixed-effect model matrix is rank deficient so dropping 26 columns / coefficients`

Including variables that are highly correlated can produce misleading results (see Section 3.3). In this case, parameter estimates from this model are not meaningful. Dropping one or more of the covariates will fix this problem.

- `No Intercept term was specified in the formula: The results will not behave as expected and may be very wrong!!`

An intercept (i.e. mean term) must be specified order for the results to be statistically valid. Otherwise, the variance percentages will be *very* overestimated.

- `Categorical variables modeled as fixed effect: The results will not behave as expected and may be very wrong!!`

If a linear mixed model is used, all categorical variables must be modeled as a random effect. Alternatively, a fixed effect model can be used by modeled all variables as fixed.

- `executing %dopar% sequentially: no parallel backend registered`

These functions are optmized to run in parallel using doParallel/doMC. This warning indicates that a parallelization was not enabled. This is not a problem, but analysis will take more time.

- `fatal error in wrapper code`
- `Error in mcfork() : unable to fork, possible reason: Cannot allocate memory`
- `Error: cannot allocate buffer`

This error occurs when `fitVarPartModel` uses too many threads and takes up too much memory. The easiest solution is to use `fitExtractVarPartModel` instead. Occasionally there is an issue in the parallel backend that is out of my control. Using fewer threads or restarting *R* will solve the problem.

8.2 Problems removing samples with NA/NaN/Inf values

variancePartition fits a regression model for each gene and drops samples that have NA/NaN/Inf values in each model fit. This is generally seamless but can cause an issue when a variable specified in the formula no longer varies within the subset of samples that are retained. Consider an example with variables for sex and age where age is NA for all males samples. Dropping samples with invalid values for variables included in the formula will retain only female samples. This will cause *variancePartition* to throw an error because there is now no variation in sex in the retained subset of the data. This can be resolved by removing either age or sex from the formula.

This situtation is indicated by the following errors

- `Error: grouping factors must have > 1 sampled level`
- `Error: Invalid grouping factor specification, Individual`
- `Error in ‘contrasts<-‘(‘*tmp*‘, value = contr.funs[1 + isOF[nn]]): contrasts can be applied only to factors with 2 or more levels`
- `Error in checkNlevels(reTrms$flist, n = n, control): grouping factors must have > 1 sampled level`

Session Info

- R version 3.2.2 Patched (2015-10-08 r69496), `x86_64-apple-darwin10.8.0`
- Locale: `en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8`
- Base packages: base, datasets, graphics, grDevices, methods, parallel, stats, utils
- Other packages: Biobase 2.30.0, BiocGenerics 0.16.0, dendextend 1.1.0, doParallel 1.0.10, edgeR 3.12.0, foreach 1.4.3, ggplot2 1.0.1, iterators 1.0.8, knitr 1.11, limma 3.26.0, lme4 1.1-10, Matrix 1.2-2, variancePartition 1.0.0
- Loaded via a namespace (and not attached): BiocStyle 1.8.0, codetools 0.2-14, colorspace 1.2-6, compiler 3.2.2, digest 0.6.8, evaluate 0.8, formatR 1.2.1, grid 3.2.2, gtable 0.1.2, highr 0.5.1, labeling 0.3, lattice 0.20-33, magrittr 1.5, MASS 7.3-44, minqa 1.2.4, munsell 0.4.2, nlme 3.1-122, nloptr 1.0.4, plyr 1.8.3, proto 0.3-10, Rcpp 0.12.1, reshape 0.8.5, reshape2 1.4.1, scales 0.3.0, splines 3.2.2, stringi 0.5-5, stringr 1.0.0, tools 3.2.2, whisker 0.3-2

References

[1] D. Bates, M. Mächler, and B. Bolker. Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, page in press, 2014. arXiv:1406.5823.

[2] J. C. Pinheiro and D. M. Bates. *Mixed-Effects Models in S and S-Plus*. Springer, New York, 2000.

[3] A. Galecki and T. Burzykowski. *Linear Mixed Effects Modeling using R*. Springer, 2013.

[4] M. D. Robinson and A. Oshlack. A scaling normalization method for differential expression analysis of RNA-seq data. *Genome Biology*, 11(3):R25, 2010. doi:10.1186/gb-2010-11-3-r25.

[5] C. W. Law, Y. Chen, W. Shi, and G. K. Smyth. Voom: precision weights unlock linear model analysis tools for RNA-seq read counts. *Genome Biology*, 15(2):R29, 2014. doi:10.1186/gb-2014-15-2-r29.