**R functions for obtaining the effective degrees of freedom**

Three R functions were developed for improving the estimates of the variance components and generating the effective degrees of freedom (EDF). These functions improve the estimation of the variance components by using the restricted maximum likelihood (REML) technique.The EDF is used to assess the effectiveness of the improved estimates. The formula for computing the EDF from Richard and Kathy’s paper is calculated as twice the square of the mean divided by the variance. These variances can be obtained by calculating the sum of the elements of interest from the variance covariance matrix. The variance covariance matrix is generated from the inverse of the Fisher’s information matrix, which is the expectation of the second derivative of the likelihood function. The implementations of the three functions for two-phase experiments are described below.

The first R function is called getMSEst(). This function extracts the mean squares (MS) and the degrees of freedom (DF) from the ANOVA table of the aov() function. As mentioned in the previous paragraph, the expectation of the second derivative of the likelihood functions has to be defined to constrict the inverse of the Fisher’s information matrix. We will show that the expectation of the second derivative of the likelihood function is equal DF divided by the twice of the square of the MS.

Suppose there are *m* set of MS from the ANOVA table, these MS are assumed to have a chi-square distribution. Let these MS be denoted by , the distribution can be written as,

where the denotes the expected MS and is the DF for MS . The likelihood function can be then be shown as

L = constant - .

The first derivative with respect to can then be written as

and the expectation of the negative of the second derivative written as

As, the expected Fisher’s information matrix for the MS is the diagonal matrix containing , hence, the MS and DF can be extracted from the ANOVA table to generate Fisher’s information matrix.

The sources of variation in the ANOVA table can be either fixed for random. The MS and DF are extracted from the sources of variation should not contain any fixed effect. This is because the variances should only be estimated from the sources of variation containing the random effects. However, there are some cases where the fixed effects are confounded with the random effects, i.e. balanced incomplete block design. In these cases, the amount of confounding treatment information can be small enough to be neglected. This issue is out of scope for this write-up and therefore will not be addressed.

The MS and DF are normally be extracted from the ANOVA table using the R function aov(). However, because the aov() function only implements a single stage of decomposition, this cannot be applied directly to two-phase experiments. Two-phase experiments require two stages of decomposition; decomposition of the information from the Phase 1 block structure in the Phase 2 bock structure, and decomposition the information from the treatment structure in the Phase 1 block structure. Based on this idea of two stages of decomposition the aov() function can be applied twice, i.e. once for each stage of decomposition.

By applying the aov() function twice, the MS and DF can be extracted from each stage of decomposition by:

1. aov() is applied the first time where the Phase 1 block structure is fitted as the fixed effects and Phase 2 block structure as random effects. The MS and DF from the decomposition of the Phase 1 block structure in the Phase 2 block structure are extracted.

2. aov() is then applied the second time where the treatment structure is fitted as the fixed effects and the sum of the Phase 2 and Phase 1 block structures as the random effects. The MS and DF are again extracted from the sources of variation that were not obtained the first stage of decomposition.

The second R function is getGMat(). This function constructs the G matrix, which is used to transform the score function and the expected Fisher’s information matrix with respect to MS, to with respect to the parameters of interest, denoted by a vector . Hence, if there are *k* elements in vector , the G matrix will have a dimension of m-by-k, where the element is equal to .

The getGMat() function constructs the G matrix by using the getVCs.twoPhase()function from the infoDecompuTE package., because the G matrix is basically the variance components structure to each source of variation. The variance components structure is the coefficients of the variance components of the expected mean squares in the ANOVA table. Note the getMSEst() function only extract the MS and DF of the source of variation without the treatment information. Hence, the variance components structures extracted in getGMat() function has to match the sources of variation that were extract from the output of getMSEst() function.

The G matrix generated from getGMat() function is different from Richard and Kathy’s paper. In Richard and Kathy’s paper, their G matrix is a binary matrix of 0 and 1. To allow our function be more general, the G matrix that is generated by getGMat() contains the coefficients of the variance components. Having the coefficients in the G matrix, it allows parameter of interest, , to be vector containing every single term of the variance components and each with coefficient of one. This G matrix is used, because sometime the coefficients of the variance components are not always identical for different sources of variation in a same ANOVA table. Hence, this G matrix avoid of adjusting these coefficients with different linear combination of the variance component for a complicated analysis. Therefore, the expected mean squares is .

The third and final R function is getVcEDF(). This function consists of two main steps, first is to generate the newly optimised variance components, and the second to calculate the effective degrees of freedom.

The improved estimation of the variance components uses the REML technique which requires the construction of the Fisher’s information matrix and score function. The Fisher’s information matrix and score function with respect to is known, however, since what we are interested in is the variance components , the first step of this function is to transform the score function and the expected Fisher’s information matrix with respect to to . Record the expected Fisher’s information matrix with respected to can be written as

.

The expected Fisher’s information matrix with respect to , denoted by , can be generated from pre- and post-multiplying the by the G matrix, i.e.

The score function with respected to is also obtained from pre-multiplying first derivative of the likelihood function by the transpose of the G matrix, this can be written as

From this, the iterative scheme for estimating the optimised variance components, , can be derived by:

.

This iterative scheme is also known as the Fisher’s scoring algorithm in the REML method. The Fisher’s information matrix and score function are continuously updated using the newly optimised variance components . Note that the expected mean squares , are also continuously updated as

.

This iterative algorithm will stop when the have converged.

The variances can be obtained by calculating the sum of the elements of interest from the variance covariance matrix. The variance covariance matrix is generated from the inverse of the Fisher’s information matrix. However, since the variance components that are estimated only have coefficients of one, these coefficients have to be re-adjusted based on the variance components structure from the ANOVA table. This adjustment is based on the idea for calculating the sum of the variances with coefficients, which its formula can be written as

.

The effective degrees of freedom are then approximated as twice the square of the mean divided by the variance. Both mean and variance are obtained from the newly optimised parameters.

The end result of the function getVcEDF() is the EDF for every source of variation without the treatment information and the newly optimised variance components.

These three functions can still be further improved. One issue is that the matching between the ANOVA table from the aov() function and getVCs.twoPhase() function from the infoDecompuTE package can sometimes be problematic. This is because sometimes the names for sources of variation can become confusing with a more complicated analysis. To prevent this issue, we can write our own functions for generating the G matrix and calculating the mean squares. Using custom functions, it ensures the names for both outputs are properly matched.

Example

I will start with a simple example consisting of a completely randomised design with 4 animals and 2 treatments for first phase, and 4-by-4 iTRAQ experiment for the second phase experiment.

MS.likelihood =

getMSEst(response = y, trt.str = "Trt + Tag", blk.str1 = "Ani",

blk.str2 = "Run", design = design)

sv.name DFF MSS m

1 Between Run 3 698.368899 3.075541e-06

2 WithinBetweenAniResidual 2 32.745257 9.326167e-04

3 WithinResidualResidual 6 1.393081 1.545853e+00

G.mat = getGMat(MS.likelihood = MS.likelihood, trt.str = "Trt + Tag", blk.str1 = "Ani",

blk.str2 = "Run", design = design)

DF e Ani Run

Between Run 3 1 0 4

Within

Between Ani

Trt 1 1 4 0

Residual 2 1 4 0

Residual

Tag 3 1 0 0

Residual 6 1 0 0

e Ani Run

Between Run 1 0 4

WithinBetweenAniResidual 1 4 0

WithinResidualResidual 1 0 0

getVcEDF(MS.likelihood = MS.likelihood, G.mat = G.mat)

$Stratum

sv.name DFF MSS m EDF

1 Between Run 3 698.368899 3.075541e-06 3.000012

2 WithinBetweenAniResidual 2 32.745257 9.326167e-04 2.002416

3 WithinResidualResidual 6 1.393081 1.545853e+00 6.000000

$Var.comp

[,1]

e 1.393081

Ani 7.838044

Run 174.243954



Using the realisation of the variances when generating he random numbers



Using the realisation of the variances when generating he random numbers

Converge at EDF = 3