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

An R function getVcEDF() was developed for improving the estimates of the variance components and generating the effective degrees of freedom (EDF). This function improves 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 (paper citation) 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 function getVcEDF()consists of three steps. The first step extracts the mean squares (MS) and degrees of freedom (DF) from the ANOVA table generated by the summary.aov.twoPhase() function from the inforDecompuTE package. The second step obtains the G matrix, which is used for changing the variables of interest. The last step is to improve the variance component estimates and to compute the EDF.

The first step is to take the data and the experimental then extract the MS and its associated DF. The MS and DF are extracted from the ANOVA table generated from summary.aov.twoPhase() function of the inforDecompuTE package. 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.

The first part of getVcEDF() works by calling the summary.aof.twoPhase() function from the infoDecompuTE package to obtain the MS and degrees of freedom (DF) from the ANOVA table. This function works by using the R function aov(). 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 is applied twice, i.e. once for each stage of decomposition.

We will show here that the expectation of the second derivative of the likelihood function is equal to the 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 , also known as the score function, 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 or 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 second step is to construct the G matrix. This step also uses the summary.aov.twoPhase() function from the infoDecompuTE package. Note that the score function and the expected Fisher’s information matrix are with respect to the MS, but what we want to estimate are the individual variance components which make up the EMS. Hence, we want to transform the score function and the expected Fisher’s information matrix with respect to MS, to with respect to the variance component estimates, denoted by a vector . This transformation can be achieved by using the m-by-k G matrix, if there are k variance components to be estimated, where each element of the G matrix is. Hence, the expected mean squares can also be written as . This technique is also known as change of variables.

The G matrix that is extracted here is different to (cite). To enable the R function to be used for every experimental design, the G matrix that is generated here also contains the coefficients of the variance components. This is different to (cite), who used a binary G matrix. Having the coefficients in the G matrix, it allows parameter of interest, a vector , to contain the individual variance components, and each with coefficient of one. This G matrix is used, because sometimes the structure and the coefficients of the variance components of the EMS are not always what we expect for different sources of variation in the ANOVA table. Hence, by using this type of G matrix, it avoids the need to study every theoretical ANOVA table and adjust these coefficients with different linear combination of the variance component for a complicated analysis.

Note the previous step only extract the MS and DF of the source of variation without the treatment information. Hence, the variance components structure extracted in this step has to match the sources of variation that were extracted in the previous step.

The third step is to estimate and optimise the variance components and compute the EDF. The variance components can be estimated based on the linear combination structure of the EMS and the calculated mean square based on given data. However, the estimation of the variance components can be further improved using the REML which requires the construction of the Fisher’s information matrix and score function. The EDF can then be approximated as twice the square of the mean divided by the variance.

We will show the mathematical procedure on improving the variance component estimates using the REML technique. Note the expected Fisher’s information matrix with respect to the expected mean squares, , can be written as

.

Since what we are interested in is the variance components, , 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 respect to is obtained by multiplying the transpose of the G matrix by the first derivative of the likelihood function, this can be written as

.

From this, the Fisher’s scoring algorithm in REML, also known as the iterative scheme for estimating the optimised variance components, , can be derived by

.

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.

In order to calculate the EDF, it is necessary to know the variances of the parameters of interest. 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 second part of this step is to approximate the EDF which is twice the square of the mean divided by the variance. Both mean and variance are obtained from the newly optimised variance components.

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

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.

Design

> phase2designEX4

Run Ani Tag Trt

1 1 A 114 Con

2 1 B 115 Con

3 1 C 116 Dis

4 1 D 117 Dis

5 2 B 114 Con

6 2 C 115 Dis

7 2 D 116 Dis

8 2 A 117 Con

9 3 C 114 Dis

10 3 D 115 Dis

11 3 A 116 Con

12 3 B 117 Con

13 4 D 114 Dis

14 4 A 115 Con

15 4 B 116 Con

16 4 C 117 Dis

Initialise the values

gamma.run = 0.1

gamma.ani = 100

run.eff = rnorm(4, mean = 0, sd = sqrt(gamma.run \* 1))

ani.eff = rnorm(4, mean = 0, sd = sqrt(gamma.ani \* 1))

trt.eff = c(1, 2)

tag.eff = c(0,0,0,0)

res.eff = rnorm(16, mean = 0, sd = 1)

real.VC = c(1, (gamma.ani \* 1),(gamma.run \* 1))

y = with(design, run.eff[Run] + ani.eff[Ani] + tag.eff[Tag] + trt.eff[Trt]) + res.eff

First function is summary.aov.twoPhase(). The inputs of this function has the response for computing the means squares, the remaining input of this function is exactly the same as the getVCs.twoPhase() function. The output of the summary.aov.twoPhase() function contains the mean squares in the last column of the random effects table with the variance component structure of the expected mean squares.

> aov.table = summary.aov.twoPhase(design.df = design, blk.str1 = "Ani",

+ blk.str2 = "Run", trt.str = "Trt + Tag", response = y)$ANOVA

DF e Ani Run MS

Between Run 3 1 0 4 0.36722

Within

Between Ani

Trt 1 1 4 0 183.05249

Residual 2 1 4 0 156.99304

Residual

Tag 3 1 0 0 0.2709

Residual 6 1 0 0 2.04342

The second function is getVcEDF(). This function takes the ANOVA table from the previous function and generates the effective degrees of freedom and the variance component estimates. Note that the user has to specify which rows of the mean squares will be used for computation.

> getVcEDF(aov.table = aov.table, row.MS =c(1,5,8), real.VC = real.VC)

$Stratum

DF e Ani Run MS EDF r.EDF

Between Run 3 1 0 4 0.36722 3 3

WithinBetweenAniResidual 2 1 4 0 156.99304 2 2

WithinResidualResidual 6 1 0 0 2.04342 6 6

$Var.comp

[,1]

e 2.04342

Ani 38.73741

Run -0.41905

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