Understanding meta-analysis through data simulation with application to power analysis

Supplementary materials

Filippo Gambarota Gianmarco Altoè

Contents

1 Introduction

In the main paper, we presented the basic meta-analysis models and simulations. However, in real-world meta-analyses, it is common to find complex data structures such as multilevel and multivariate. As reported in the paper, there could be a different source of dependencies when each study reports multiple effect sizes. Rephrasing the (Cheung, 2015, pp. 121–122) taxonomy, the first type of dependency, which we call *multilevel* dependency, arises when multiple independent experiments are associated with the same study. Level 1 (i.e., participants' level) is different, but the effects are likely correlated because the same research group conducted the experiments. The second type of dependency, *multivariate* dependency, arises when multiple effects are collected on the same pool of participants within each study. Now sampling errors within each study are correlated because the same participant is tested multiple times. Finally, we have a dependency on multiple effects at the population level. Two outcomes could be correlated (e.g., the type of memory tasks) even if the pool of participants is different. The first two types of dependency are the reasons to use a multilevel and multivariate model. The following sections will show two examples to simulate these complex data structures.

2 Simulations

```
library(dplyr)
library(tidyr)
seed <- 2023 # random seed for simulations
```

In the paper, we used the `mapply()` function, a particularly useful tool in R, to apply the same function (in parallel) with multiple parameters using multiple vectors. In our case, the `sim_study()` function has several parameters (`theta`, `nt`, among others), and we can apply the function multiple times with a combination of parameters. It can be considered essentially a `for` loop but more compact and less verbose.

The following code illustrates the idea of `mapply()` compared to a standard `for` loop. The sim data set contains our pool of k studies, and we want to apply the `sim_study()` function to each dataset row. We can use a `for` loop that is verbose but more explicit or the `mapply()` function that is more compact but less explicit. The computation time and results `for` the two approaches are equivalent, given the appropriate setup.

```
# using a for loop

# preallocate for speed (not necessary with mapply)
res <- vector(mode = "list", length = nrow(sim))
for(i in 1:length(res)){
  res[[i]] <- sim_study(theta = sim$theta[i], nc = sim$nc[i], nt = sim$nt[i])
}

# using mapply
sim <- mapply(sim_study, sim$theta, sim$nc, sim$nt)

# using sim_studies (with mapply inside)
sim <- sim_studies(sim$theta, sim$nc, sim$nt, data = sim)
```

3 Three-level model

Compared to the standard two-level model, the three-level model estimates another source of heterogeneity (ω^2) representing the variability within studies. We can easily extend the two-level *random-effects* equation

into equation (S1) from the main paper adding another "adjustment" to the overall effect.

$$
\begin{aligned}
d_{ij} &= \theta_r + \theta_i + \theta_{ij} + \epsilon_{ij} \\
\theta_i &\sim N(0, \tau^2) \\
\theta_{ij} &\sim N(0, \omega^2) \\
\epsilon_{ij} &\sim N(0, \sigma_i^2)
\end{aligned}
\tag{S1}
$$

The notation suggests that each effect size belongs to an experiment (effect size) (j) nested within a study (i). The variability between studies (level 3) is handled by the τ^2 parameter and the variability within studies (level 2) is handled by the ω^2 parameter. Practically, each study is composed of the average effect size θ_r plus a study-specific adjustment determined by τ^2. Then each effect size is composed of the study-specific effect ($\theta_r + \theta_i$) plus the adjustment determined by ω^2 thus $\theta_r + \theta_i + \theta_{\text{ij}}$. The number of effect sizes within each study will determine the estimation precision of the ω^2 parameter and the number of studies will determine the estimation precision of the τ^2 parameter. The relationship between τ^2 and ω^2 can be expressed using the intraclass correlation coefficient (ICC) expressed as $ICC = \frac{\tau^2}{\tau^2+\omega^2}$ representing the proportion of total heterogeneity associated with level 2 or level 3. For example, an ICC of 0.5 suggests that 50% of heterogeneity is caused by between-studies variability. On the extreme, an ICC close to 1 suggests that the between-studies variability causes most heterogeneity (i.e., the effect sizes within each study are very similar).

```
set.seed(seed)
i <- 50 # Number of studies (level 3)
j <- 5 # Number of effect sizes within each study (level 2)
tau2 <- 0.3 # heterogeneity between studies
omega2 <- 0.1 # heterogeneity within studies
icc <- tau2 / (tau2 + omega2) # real ICC
n <- 30 # sample size for each group, within each study
theta_r <- 0.3 # real average effect size

theta_i <- rnorm(i, 0, sqrt(tau2))
theta_ij <- rnorm(i*j, 0, sqrt(omega2))

sim <- tidyr::expand_grid(
  study = 1:i,
  effect = 1:j,
  nt = n,
  nc = n,
  theta_r = theta_r
)

sim$theta_i <- theta_i[sim$study]
sim$theta_ij <- theta_ij

head(sim)
```

```
## # A tibble: 6 x 7
##   study effect    nt    nc theta_r theta_i theta_ij
##   <int>  <int> <dbl> <dbl>   <dbl>   <dbl>    <dbl>
## 1     1      1    30    30     0.3 -0.0459    0.210
## 2     1      2    30    30     0.3 -0.0459   -0.345
## 3     1      3    30    30     0.3 -0.0459   -0.134
## 4     1      4    30    30     0.3 -0.0459    0.374
```

```
## 5     1      5    30    30      0.3 -0.0459    0.501
## 6     2      1    30    30      0.3 -0.538     0.721
```

Each study i is repeated j times[1] along the θ_i while each effect has a unique θ_{ij}. We can use the same approach as the main paper using the `sim_studies()` function.

```
set.seed(seed)
sim$theta_r_i_ij <- sim$theta_r + sim$theta_i + sim$theta_ij
sim <- sim_studies(theta = sim$theta_r_i_ij, nt = sim$nt, nc = sim$nc, data = sim)
```

Finally we fit a *three-level models* using the `rma.mv` from the `metafor` package. The syntax differs slightly from the rma function and is clearly explained here https://www.metafor-project.org/doku.php/analyses:konstantopoulos2011. The essential part is adding the nested random effect using the `random =` argument specifying that the `study` variable is nested within the `study` variable with `~ 1|study/experiment`.

```
res <- rma.mv(yi, vi, random = ~1|study/effect, data = sim)
summary(res)
```

```
##
## Multivariate Meta-Analysis Model (k = 250; method: REML)
##
##    logLik   Deviance        AIC        BIC       AICc
## -172.7068   345.4136   351.4136   361.9660   351.5116
##
## Variance Components:
##
##             estim    sqrt  nlvls  fixed        factor
## sigma^2.1  0.2992  0.5470     50     no         study
## sigma^2.2  0.0707  0.2659    250     no  study/effect
##
## Test for Heterogeneity:
## Q(df = 249) = 1403.6785, p-val < .0001
##
## Model Results:
##
## estimate      se    zval    pval   ci.lb   ci.ub
##   0.4056  0.0810  5.0105  <.0001  0.2470  0.5643  ***
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As explained above, the model estimates the average effect θ_r and the two heterogeneity parameters. To create a more realistic simulation, we can create some variability in the number of experiments within the same paper, similar to what we did for the number of participants in the main paper. We simulated a three-level model, but it is possible to extend to n levels for a more complicated nested structure. In this case, we need more heterogeneity parameters (i.e., more nested "adjustments"), but the overall simulation setup is the same.

[1]To create a more realistic simulation, the number of effect sizes for each paper could be heterogeneous (e.g., sampled from a Poisson distribution)

4 Multivariate model

Compared to a standard two-level random-effects model, the *multivariate model* must consider the correlation between effect sizes collected on the same pool of participants. Authors should report these correlations, but often we need to guess a plausible value for the meta-analysis. Compared to the *two-level* and *three-level* models, we now have multiple outcomes (p).

Following the example of the main paper, we have three memory tests collected on the same pool of participants. For this reason we have three real effects (θ_{r_1}, θ_{r_2} and θ_{r_3}), three heterogeneity (τ_1^2, τ_2^2 and τ_3^2) and the population-level correlations between these outcomes (ρ_{12}, ρ_{13} and ρ_{23}). Moreover, the sampling errors (ϵ_i) are now correlated; thus each study will have three variances for each outcome (σ_1^2, σ_2^2, and σ_3^2), and the correlations between sampling errors (ρ_{s12}, ρ_{s13} and ρ_{s23})[2]. Equation (S2) formalize the multivariate model for a single study i with three outcomes. The random-effects adjustments (θ_i) to the overall effect and sampling errors are now sampled from a multivariate normal distribution, respectively $\theta_i \sim \mathcal{MVN}(0, \mathrm{T}^2)$ and $\theta_i \sim \mathcal{MVN}(0, \mathrm{V}^2)$.

$$
\begin{aligned}
\begin{bmatrix} d_{i1} \\ d_{i2} \\ d_{i3} \end{bmatrix} &= \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} + \begin{bmatrix} \theta_{i1} \\ \theta_{i2} \\ \theta_{i3} \end{bmatrix} + \begin{bmatrix} \epsilon_{i1} \\ \epsilon_{i2} \\ \epsilon_{i3} \end{bmatrix} \\
\begin{bmatrix} \theta_{i1} \\ \theta_{i2} \\ \theta_{i3} \end{bmatrix} &\sim \mathcal{MVN}(0, \mathrm{T}^2) \\
\begin{bmatrix} \epsilon_{i1} \\ \epsilon_{i2} \\ \epsilon_{i3} \end{bmatrix} &\sim \mathcal{MVN}(0, \mathrm{V}^2) \\
\mathrm{T}^2 &= \begin{bmatrix} \tau_1^2 & & \\ \rho_{21}\tau_2\tau_1 & \tau_2^2 & \\ \rho_{31}\tau_3\tau_1 & \rho_{32}\tau_3\tau_2 & \tau_3^2 \end{bmatrix} \\
\mathrm{V}^2 &= \begin{bmatrix} \sigma_1^2 & & \\ \rho_{s21}\sigma_2\sigma_1 & \sigma_2^2 & \\ \rho_{s31}\sigma_3\sigma_1 & \rho_{s32}\sigma_3\sigma_2 & \sigma_3^2 \end{bmatrix}
\end{aligned}
\tag{S2}
$$

We can use the `MASS::mvrnorm()` to generate data from a multivariate normal distribution in R. Compared to `rnorm()`, `MASS::mvrnorm()` requires the vector of means (in this case zero) and the variance-covariance matrix.

```
set.seed(seed)
p <- 3 # number of outcomes
taus2 <- c(0.3, 0.1, 0.2) # vector of taus
rho <- 0.6 # correlation between outcomes
Tcmat <- rho + diag(1 - rho, nrow = p) # correlation matrix
Tcmat
```

```
##      [,1] [,2] [,3]
## [1,]  1.0  0.6  0.6
## [2,]  0.6  1.0  0.6
## [3,]  0.6  0.6  1.0
```

[2] We used the subscript s for sampling errors to distinguish between between-outcomes correlations and correlations between sampling errors

```r
# correlation matrix to variance-covariance matrix
Tvcov <- diag(sqrt(taus2)) %*% Tcmat %*% diag(sqrt(taus2))
round(Tvcov, 3)
```

```
##       [,1]  [,2]  [,3]
## [1,] 0.300 0.104 0.147
## [2,] 0.104 0.100 0.085
## [3,] 0.147 0.085 0.200
```

```r
multi_sim <- MASS::mvrnorm(n = 100, mu = rep(0, p), Sigma = Tvcov)
head(multi_sim)
```

```
##             [,1]         [,2]        [,3]
## [1,] -0.02016864  0.410185486 -0.13194700
## [2,]  0.54175504 -0.069826019  0.50505458
## [3,]  1.14644044  0.336528536  0.50838866
## [4,]  0.38035421 -0.482370414  0.01473713
## [5,]  0.52888869 -0.004733355  0.05339601
## [6,] -0.52921867 -0.413480727 -0.34537516
```

```r
# check the result, close to the simulated values
apply(multi_sim, 2, mean) # mean
```

```
## [1] -0.0003956354  0.0066828825 -0.0863630452
```

```r
apply(multi_sim, 2, var) # variance
```

```
## [1] 0.29557227 0.09757498 0.18773351
```

```r
cor(multi_sim) # correlation
```

```
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.6002067 0.6356537
## [2,] 0.6002067 1.0000000 0.6074336
## [3,] 0.6356537 0.6074336 1.0000000
```

The previous code simulated a *compound symmetry* (CS) structure where all variances (τ^2) and covariances ($\rho\tau_1\tau_2$) are the same. There could be different structures of the T^2 matrix that are clearly explained in the `metafor::rma.mv()` documentation (see https://wviechtb.github.io/metafor/reference/rma.mv.html#specifying-random-effects). For the sake of simplicity we will use the CS structure. We can use the `sim_T()` function that given the number of studies k, outcomes p, the vector of τ^2 and the correlation ρ generates the random-effect adjustments[3].

```r
sim_T <- function(k, p, taus2, rho){
  rhoM <- rho + diag(1 - rho, nrow = p)
  tau2M <- diag(sqrt(taus2)) %*% rhoM %*% diag(sqrt(taus2))
  c(t(MASS::mvrnorm(k, rep(0, p), tau2M)))
}
```

[3]The function allow to specify heterogeneous τ^2 values where the CS structure assume the same τ^2 for each outcome. Using a fixed ρ and different τ^2 will create what `rma.mv()` calls a *heteroscedastic compound symmetric structure* thus a matrix where the correlation is the same between outcomes and each outcome has a different τ^2 value

```r
set.seed(seed)
# compound symmetry
sim_T(k = 10, p = 3, taus2 = c(0.2, 0.2, 0.2), rho = 0.7)
```

```
##  [1] -0.137783060  0.006924584  0.030317250 -0.400980223
##  [5] -0.301731906 -0.476820366 -0.599321253 -1.017707804
##  [9] -0.633051729 -0.101130503 -0.181352426  0.059109337
## [13] -0.267400421 -0.117154089 -0.375628328  0.047702666
## [17]  0.692241336  0.569012955 -0.534720462 -0.313042059
## [21] -0.248710208  0.206484487  0.507666832  0.487816335
## [25]  0.007204637 -0.511638473  0.025313912  0.222197914
## [29] -0.721417315 -0.062528264
```

```r
# heteroscedastic compound symmetry
sim_T(k = 10, p = 3, taus2 = c(0.1, 0.3, 0.2), rho = 0.7)
```

```
##  [1]  0.20547847  0.04380675  0.14946731 -0.50545878
##  [5] -0.69452948 -0.34453427  0.15527888  0.35994483
##  [9]  0.43542728  0.21001803  0.01567005 -0.10341940
## [13]  0.45596919  0.22106059  0.20538547  0.54161212
## [17]  0.02342131  0.15806043 -0.39421052 -0.59991781
## [21]  0.41745313 -0.32040433 -0.75720059 -0.20041835
## [25] -0.18732400 -0.57476731 -0.33691260  0.18553571
## [29] -0.09171771  0.15354564
```

For the sampling errors, the matrix V^2 for the study, i is essentially a $p_i \times p_i$ matrix, where the diagonal contains the sampling variances, and off-diagonal elements are the covariances. Crucially, sampling errors are correlated within a study but uncorrelated between studies. For this reason, stacking all matrices from k studies will create a block variance-covariance matrix where the correlations between sampling errors of different studies are fixed to zero. Figure S1 depicts the idea of the block-variance covariance matrix for three studies with a different number of outcomes. The block variance-covariance matrix can be easily created using the `metafor::vcalc()` function.

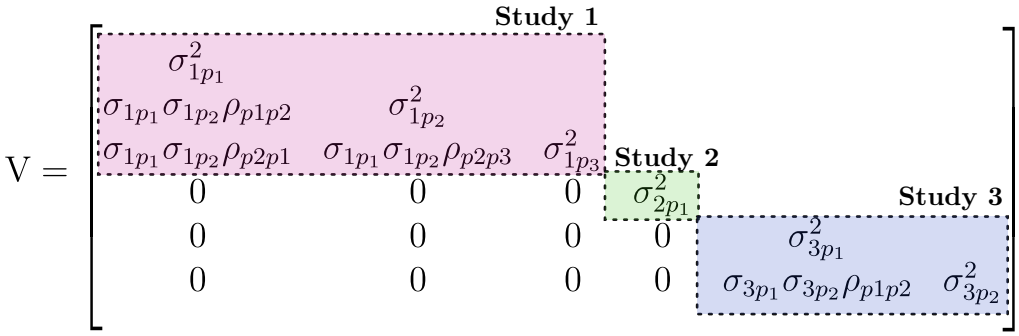

Figure S1: Example of a block variance-covariance matrix for three hypothetical studies. In the example, there are three outcomes ($p = 3$), but not all studies have all outcomes. For this reason, the matrix represents each study as a "block," and covariances between different studies are fixed to zero (because sampling errors are not correlated for different participants). The first study (pink) has three outcomes, thus a 3×3 matrix where the diagonal contains the sampling variances and off-diagonal elements are the sampling covariances. The second study has only one outcome; thus, the matrix is a single value for the sampling variance. The last study follows the same logic as the first study, with only two outcomes.

We have all pieces to simulate the multivariate model. We can use the `sim_study_m()` that follows the same idea of the `sim_study()` function but extends to multiple outcomes. We generate a single study with p outcomes where sampling errors are correlated according to a compound symmetry structure[4]. In practical terms, we assume that the correlation at the participant level between multiple outcomes is r and fixed between studies.

```
sim_study_m <- function(thetas, nt, nc, r){

  p <- length(thetas)

  # variance covariance matrix with sigma2 = 1
  Vcm <- r + diag(1 - r, nrow = p)

  yc <- MASS::mvrnorm(nt[1], rep(0, p), Vcm)
  yt <- MASS::mvrnorm(nc[1], thetas, Vcm)

  ytm <- apply(yt, 2, mean)
  ycm <- apply(yc, 2, mean)

  ytv <- apply(yt, 2, var)
  ycv <- apply(yc, 2, var)

  sp <- sqrt((ytv*(nt - 1) + ycv*(nc - 1)) / (nt + nc - 2))
  yi <- (ytm - ycm) / sp
  vi <- (nt + nc)/(nt * nc) + yi^2/(2 * (nt + nc - 2))

  data.frame(yi = yi, vi = vi)
}
```

```
set.seed(seed)
thetas <- c(0.1, 0, 0.7) # three real average outcomes
n <- 30 # number of participant per study
r <- 0.6 # correlation between sampling errors
sim_study_m(thetas, nt = n, nc = n, r = r)
```

```
##            yi         vi
## 1  0.17864476 0.06694179
## 2 -0.03272746 0.06667590
## 3  0.47827728 0.06863864
```

As in the main paper, we can create a dataframe with simulation parameters and iterating the `sim_study_m()` producing a meta-analysis dataframe.

```
set.seed(seed)
k <- 100 # number of studies
p <- 3 # number of outcomes per study
rho <- 0.7 # population level correlation between outcomes
r <- 0.5 # correlation between sampling errors
n <- 30 # number of participants per group per study

thetas_r <- c(0.1, 0, 1) # population level real effect sizes
```

[4]Also for the V^2 matrix we can assume different variance-covariance structures. In this example we are using a CS structure

```r
taus2 <- c(0.3, 0.3, 0.3) # population level real tau2s

# creating the dataframe structure
sim <- expand_grid(
  id = 1:k,
  p = 1:p,
  nt = n,
  nc = n
)

head(sim)
```

```
## # A tibble: 6 x 4
##      id     p    nt    nc
##   <int> <int> <dbl> <dbl>
## 1     1     1    30    30
## 2     1     2    30    30
## 3     1     3    30    30
## 4     2     1    30    30
## 5     2     2    30    30
## 6     2     3    30    30
```

```r
# adding the real effects and the random-effect adjustments
sim$theta_r <- rep(thetas_r, k)
sim$theta_i <- sim_T(k, p, taus2, rho)

# average effect + random-effect adjustment
sim$theta_r_i <- sim$theta_r + sim$theta_i

# splitting to list of studies for improving the simulation speed
siml <- split(sim, sim$id)

# simulating the effect sizes for each study
resl <- lapply(siml, function(x) sim_study_m(x$theta_r_i, x$nt, x$nc, r))
res <- dplyr::bind_rows(resl) # combine everything
sim <- cbind(sim, res) # append to the sim dataframe

head(sim)
```

```
##   id p nt nc theta_r    theta_i  theta_r_i          yi
## 1  1 1 30 30     0.1 -0.4351134 -0.3351134 -0.06813267
## 2  1 2 30 30     0.0  0.2623641  0.2623641  0.45974732
## 3  1 3 30 30     1.0  0.2958867  1.2958867  1.48355608
## 4  2 1 30 30     0.1  0.8548680  0.9548680  0.86504584
## 5  2 2 30 30     0.0  0.3180528  0.3180528  0.33427264
## 6  2 3 30 30     1.0  0.2717055  1.2717055  1.35444420
##           vi
## 1 0.06670668
## 2 0.06848880
## 3 0.08564028
## 4 0.07311757
## 5 0.06762993
## 6 0.08248149
```

Now, we can use the `metafor::rma.mv()` function to fit the multivariate random-effects model. Compared to the multilevel model, the `metafor::rma.mv()` for a multivariate model uses the block variance-covariance matrix as the `vi` argument and the variable representing each outcome as a moderator estimating the average effect size. We can create the block variance-covariance matrix. Given that we simulated the raw data, we can calculate and use the observed correlations (within the `sim_study_m()` function) and use a different correlation for each study within the `vcalc()` function (see https://www.metafor-project.org/doku.php/analyses:berkey1998 for an example). Often, these values are not reported; thus, we need to guess a plausible correlation to compute the block variance-covariance matrix. Here we use the r value from our simulation and use the same for each study. We know the underlying model in this case, so our guessed value is correct.

```
# create the block variance-covariance matrix
# cluster is the study (i.e. each "block")
# obs identify each outcome (i.e., each "block" number of rows/columns)
# r is the fixed sampling errors correlation
V <- vcalc(vi = vi, cluster = id, obs = p, rho = r, data = sim)

# first two studies (notice the off diagonal zeros for the correlation between different studies)
round(V, 3)[1:6, 1:6]
```

```
##       [,1]  [,2]  [,3]  [,4]  [,5]  [,6]
## [1,] 0.067 0.034 0.038 0.000 0.000 0.000
## [2,] 0.034 0.068 0.038 0.000 0.000 0.000
## [3,] 0.038 0.038 0.086 0.000 0.000 0.000
## [4,] 0.000 0.000 0.000 0.073 0.035 0.039
## [5,] 0.000 0.000 0.000 0.035 0.068 0.037
## [6,] 0.000 0.000 0.000 0.039 0.037 0.082
```

```
# outcome to character with a meaningful prefix
sim$p <- paste0("outcome", sim$p)

# fitting the model
res <- rma.mv(yi, V, mods = ~ 0 + p, random = ~p|id, data = sim, struct = "CS")
summary(res)
```

```
##
## Multivariate Meta-Analysis Model (k = 300; method: REML)
##
##    logLik   Deviance        AIC        BIC       AICc
## -211.9677   423.9353   433.9353   452.4040   434.1415
##
## Variance Components:
##
## outer factor: id (nlvls = 100)
## inner factor: p  (nlvls = 3)
##
##             estim    sqrt  fixed
## tau^2      0.2755  0.5249     no
## rho        0.6508             no
##
## Test for Residual Heterogeneity:
## QE(df = 297) = 1211.1472, p-val < .0001
##
```

```
## Test of Moderators (coefficients 1:3):
## QM(df = 3) = 592.4324, p-val < .0001
##
## Model Results:
##
##            estimate      se     zval    pval    ci.lb
## poutcome1    0.0693  0.0587   1.1794  0.2382  -0.0458
## poutcome2   -0.0894  0.0588  -1.5217  0.1281  -0.2047
## poutcome3    1.0367  0.0595  17.4131  <.0001   0.9200
##             ci.ub
## poutcome1  0.1843
## poutcome2  0.0258
## poutcome3  1.1534  ***
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The output is more complicated compared to previous models. The `Variance Components` is the estimation of the T^2 matrix. In the two-level random-effects model, we have a single τ^2 while in the multivariate model, we have a full variance-covariance matrix. Using the `struct = "CS"` argument, we are forcing the `rma.mv()` function to use a compound symmetry structure, thus estimating a single τ^2 and a single ρ. Again, this is consistent with our simulation but could be a stringent assumption in a real-world analysis. The crucial part is the `random = ~ p|id` argument that specifies the random-effects structure. The syntax is clearly explained here https://wviechtb.github.io/metafor/reference/rma.mv.html#specifying-random-effects The basic idea is to specify the random effect structure as `~ inner|outer` where inner is the outcome variable and outer is the paper variable. The mods = `~ 0 + p` argument specify to estimate the average effect for each outcome (i.e., the θ_r vector of average effects). We are removing the intercept (`~ 0 + p` or equivalently `~ p - 1`) and the model will estimate the mean for each level of the `p` variable.

We could use also another structure for the T^2 matrix. We know that our data are generated using a CS structure thus using another structure will be less appropriate. In reality, is not known which is the real data generation model thus we can try different structure. Setting the `struct = "UN"` (unstructured) argument, `rma.mv` try to estimate a variance-covariance matrix where all correlations and variances need to be estimated. Compared to the previous model, we are know estimating 3 heterogeneity and three correlation parameters.

We could also use another structure for the T^2 matrix. We know that our data are generated using a CS structure; thus, using another structure will be less appropriate. In reality, it is not known which is the real data generation model; thus, we can try different structures. Setting the struct = "UN" (unstructured) argument, rma.mv tries to estimate a variance-covariance matrix where all correlations and variances need to be estimated. We are now estimating three heterogeneity and three correlation parameters compared to the previous model.

```r
# fitting the model
res <- rma.mv(yi, V, mods = ~ 0 + p, random = ~p|id, data = sim, struct = "UN")
summary(res)
```

```
##
## Multivariate Meta-Analysis Model (k = 300; method: REML)
##
##    logLik   Deviance        AIC        BIC       AICc
## -209.3116   418.6233   436.6233   469.8669   437.2505
##
## Variance Components:
```

```
##
## outer factor: id (nlvls = 100)
## inner factor: p  (nlvls = 3)
##
##             estim    sqrt  k.lvl  fixed     level
## tau^2.1    0.2298  0.4794    100     no  outcome1
## tau^2.2    0.3169  0.5630    100     no  outcome2
## tau^2.3    0.2810  0.5301    100     no  outcome3
##
##           rho.otc1  rho.otc2  rho.otc3    otc1  otc2  otc3
## outcome1         1                          -   100   100
## outcome2    0.5786         1                no     -   100
## outcome3    0.6872    0.7004         1      no    no     -
##
## Test for Residual Heterogeneity:
## QE(df = 297) = 1211.1472, p-val < .0001
##
## Test of Moderators (coefficients 1:3):
## QM(df = 3) = 673.1579, p-val < .0001
##
## Model Results:
##
##            estimate      se     zval    pval    ci.lb
## poutcome1    0.0690  0.0547   1.2613  0.2072  -0.0382
## poutcome2   -0.0897  0.0622  -1.4425  0.1492  -0.2117
## poutcome3    1.0364  0.0600  17.2756  <.0001   0.9188
##             ci.ub
## poutcome1  0.1762
## poutcome2  0.0322
## poutcome3  1.1540  ***
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

5 Useful resources

- https://www.jepusto.com/simulating-correlated-smds/
- https://www.metafor-project.org/doku.php/metafor
- https://stat.ethz.ch/mailman/listinfo/r-sig-meta-analysis
- https://bookdown.org/MathiasHarrer/Doing_Meta_Analysis_in_R/

6 References

Cheung, M. W.-L. (2015). *Meta-Analysis: A structural equation modeling approach.* John Wiley & Sons.