# Finite sample properties of the estimator for the population average effect in univariate meta-analyses

*Han Bossier*

*4-10-2017*

## Contents

## 1 Introduction

Using Monte Carlo simulations, we investigate finite-sample properties of the estimator for the average population effect in univariate meta-analyses. The population effect is estimated using a weighted average of the study level effect sizes. We will use Hedges' $g$ as a standardized measure of effect in a one-sample setting. We use random effects meta-analyses to estimate the population effect. Calculations are provided for estimating within-study variability whilst using the DerSimonian and Laird estimator for the between-study variability.

We will generate data at subject level using a normal linear model. We thus start with the known true value of $\beta$, the parameter in the normal linear model. We then introduce within and between study variability. From there we work out the true value of the effect size and the weighted average, estimated at the meta-analysis level.

In the following sections, we give a theoretical background of the univariate meta-anlaysis. Then we explain the set-up for our Monte Carlo simulation. We look at the distribution of the estimator, the normal approximation, the average type I error rate and the empirical power in the results section. We conclude with one sentence in the conclusion.

## 2 Univariate meta-analysis

We focus on the univariate meta-analysis with Hedges' $g$ as a measure of the effect size. To estimate within-study variance, we will first provide our own calculations. Then we will compare these with the expression found in Hedges (1989).

First we discuss the effect size, then the calculation for estimating within- and between-study variance and then the statistical test procedure.

### 2.1 Effect size

Consider a one-dimensional response vector $Y$. In this report, we will generate data using a normal linear model. In this case, it is possible to convert the $t$-value associated to each study, obtained with OLS, using the following:

$$g = \frac{t}{\sqrt{N}} \times J, \tag{1}$$

with $N$ the study sample size and $J$ a correction factor, defined as:

$$J = 1 - \left(\frac{3}{(4 \times (N-1)) - 1}\right). \tag{2}$$

Note, this corresponds to the general form of a standardized effect size:

$$g = \frac{\frac{\overline{X}}{S_X} \times \sqrt{N}}{\sqrt{N}} \times J \tag{3}$$

$$= \frac{\overline{X}}{S_X} \times J, \tag{4}$$

with $S_X$ the (unbiased) standard deviation estimating $\sigma$. For each study $k = 1 \dots K$, we estimate an effect size (denoted as $g_k$).

### 2.2 Weighted average

A meta-analysis attempts to estimate a population effect ($\theta$) using the weighted average ($M^*$) over K studies. Consider the set of weights, denoted as $W$. Then we have:

$$M^* = \frac{\sum_{k=1}^{K} W_k \times g_k}{\sum_{k=1}^{K} W_k}. \tag{5}$$

In the case of a random effects meta-analysis (which we consider in this report), the weights correspond to the inverse of the sum of within and between study variability. Estimation of these variances are discussed in the next two sections.

## 2.3 Within study variance

### 2.3.1 Own calculation

In this section, I try to formulate an expression for $\mathrm{Var}(g_k)$, from which I drop the subscript $k$ for ease of notation. First, note that

$$\begin{aligned}\mathrm{Var}(g) &= \mathrm{Var}\left[\frac{\overline{X}}{S_X} \times J\right] &(6)\\ &= J^2 \mathrm{Var}(\overline{X} S_X^{-1}). &(7)\end{aligned}$$

Next we use the following property for two random independent variables $X$ and $Y$:

$$\mathrm{Var}(XY) = \mathrm{Var}(X)\mathrm{Var}(Y) + \mathrm{Var}(X)[E(Y)]^2 + \mathrm{Var}(Y)[E(X)]^2. \tag{8}$$

Thus, we get:

$$\mathrm{Var}(\overline{X} S_X^{-1}) = \mathrm{Var}(\overline{X})\mathrm{Var}(S_X^{-1}) + \mathrm{Var}(\overline{X})[E(S_X^{-1})]^2 + \mathrm{Var}(S_X^{-1})[E(\overline{X})]^2. \tag{9}$$

In the case of $\overline{X}$, we have

$$\begin{aligned}\mathrm{E}(\overline{X}) &= \mu &(10)\\ \mathrm{Var}(\overline{X}) &= \frac{\sigma^2}{N}. &(11)\end{aligned}$$

The case of $S_X^{-1}$ is a bit more difficult. First we consider the distribution of the *biased sample standard deviation*[1]:

$$f_N(s) = 2\frac{\left(\frac{N}{2\ \sigma^2}\right)^{(N-1)/2}}{\Gamma\left(\frac{1}{2}(N-1)\right)} e^{-Ns^2/(2\sigma^2)} s^{N-2}. \tag{12}$$

Using the first two raw moments, it is possible to define the mean and variance. We get:

$$\begin{aligned}\mathrm{E}\left(S_X\right) &= \sqrt{\frac{2}{N}}\ \frac{\Gamma(\frac{1}{2}N)}{\Gamma\left(\frac{1}{2}\left(N-1\right)\right)}\sigma &(13)\\ \mathrm{Var}\left(S_X\right) &= \frac{1}{N}\ \left[N-1-\frac{2\ \Gamma^2\left(\frac{N}{2}\right)}{\Gamma^2\left(\frac{N-1}{2}\right)}\right]\sigma^2. &(14)\end{aligned}$$

Note that the expected value of a random variable does not equal the expectation of the inverse of a random variable. For larger $N$, the distribution of $S_x$ is nearly symmetric. Using a Taylor expansion around $\mathrm{E}(S_X)$, we could use[2]:

[1] http://mathworld.wolfram.com/StandardDeviationDistribution.html

[2] see https://stats.stackexchange.com/questions/80874/expectation-of-reciprocal-of-a-variable

$$\mathrm{E}\left(\frac{1}{S_X}\right) = \frac{1}{\mathrm{E}(S_X)} + \frac{1}{\mathrm{E}(S_X)^3}\mathrm{Var}(S_X) \tag{15}$$

Likewise, the variance of a random variable is not equal to the variance of the inverse of the random variable. For now, I use the following:

$$\mathrm{Var}\left(\frac{1}{S_X}\right) = \frac{\mathrm{Var}(S_X)}{\overline{S}_X^4}. \tag{16}$$

Though, I am not sure if these latter two equations hold in our case (e.g. it might assume normal distributed random variables). Obtaining $\overline{S}_X$ using equation (13), we are ready to calculate $\mathrm{Var}(g)$ (equation (7)). The following **R** code contains functions to implement the calculation.

```r
# Correction factor J
corrJ <- function(N){
  1-(3/((4*(N-1))-1))
}

# Expectation of S_X
E_Sx <- function(N, sigma){
  num <- gamma(N/2)
  denom <- gamma((N - 1) / 2)
  sqrt(2/N) * (num/denom) * sigma
}

# Expectation of inverse of S_X
# -- Input: expection of Sx and variance of S_X
E_inv_Sx <- function(exp_Sx, variance_Sx){
  (1/exp_Sx) + (1/(exp_Sx)^3) + variance_Sx
}

# Variance of \bar{X}
var_meanX <- function(N, sigma){
  sigma^2/N
}

# Variance of S_X
var_Sx <- function(N, sigma){
  num <- 2 * gamma(N/2)^2
  denom <- gamma((N-1) / 2)^2
  (1/N) * (N - 1 - (num/denom)) * sigma^2
}

# Variance of S_X^{-1}
# -- Input: variance of S_X, expectation of S_X
var_inv_Sx <- function(exp_Sx, variance_Sx){
  variance_Sx / ((exp_Sx)^4)
}
```

Now consider the case where we look at $\mu = 1$, $\sigma = 2$ and $N = 50$. The variance of $g$ in my case would be:

```
mu <- 1
sigma <- 2
N <- 50
expectation_Sx <- E_Sx(N, sigma)
variance_Sx <- var_Sx(N, sigma)
expectation_inv_Sx <- E_inv_Sx(expectation_Sx, variance_Sx)
variance_inv_Sx <- var_inv_Sx(expectation_Sx, variance_Sx)
varG <- corrJ(N)^2 * ((var_meanX(N, sigma) * variance_inv_Sx) +
                var_meanX(N, sigma)*expectation_inv_Sx^2 +
                variance_inv_Sx*mu^2)
results <- data.frame('Parameter' = c('Expectation of S_x',
                      'Expectation of inverse of S_x',
                      'Variance of S_x',
                      'Variance of inverse of S_x',
                      'within study variance of g'),
           'Value' = c(expectation_Sx, expectation_inv_Sx,
            variance_Sx,
            variance_inv_Sx,varG))
print(results, row.names = FALSE)
                     Parameter       Value
            Expectation of S_x 1.969823885
 Expectation of inverse of S_x 0.678286612
               Variance of S_x 0.039793864
    Variance of inverse of S_x 0.002643057
    within study variance of g 0.038449389
```

The result above shows a within study variance of $g$ equal to 0.0384494. Let us compare with a formula found in Hedges (1989). But before doing so, I will create a function to calculate the variance using the calculations from this section.

```
ownVarG <- function(mu, sigma, N){
  varG <- corrJ(N)^2 * ((var_meanX(N, sigma) * variance_inv_Sx) +
                var_meanX(N, sigma)*expectation_inv_Sx^2 +
                variance_inv_Sx*mu^2)
  return(varG)
}
```

### 2.3.2 Hedges (1989)

In Hedges (1989), the variance of $g$ is calculated as:

$$\mathrm{Var}(g) = \frac{1}{N} + \left[1 - \left(\frac{\Gamma((N-2)/2)}{\Gamma((N-1)/2)}\right)^2 \times \frac{(N-3)}{2}\right] \times g \tag{17}$$

Using again $\mu = 1$, $\sigma = 2$ and $N = 50$, we calculate the true value of $g$.

```
true_g <- mu / sigma * corrJ(N = N)
```

In this case $g = 0.4923077$. Using the following function, we are ready to calculate the variance of $g$ with equation (17).

```
# Calculate within study variance of g using Hedges (1989)
varHedge <- function(g,N){
  value <- (1/N) + (1 - (gamma((N - 2) / 2) / gamma((N - 1) / 2))^2 * (N - 3) / 2) * g^2
  return(round(value,7))
}
```

We get $\text{Var}(g) = 0.0225645$.

Close to our expression!

## 2.4 Comparison

Let us generate some values for $g$, $\sigma$ and $N$ and see how the two calculations differ.

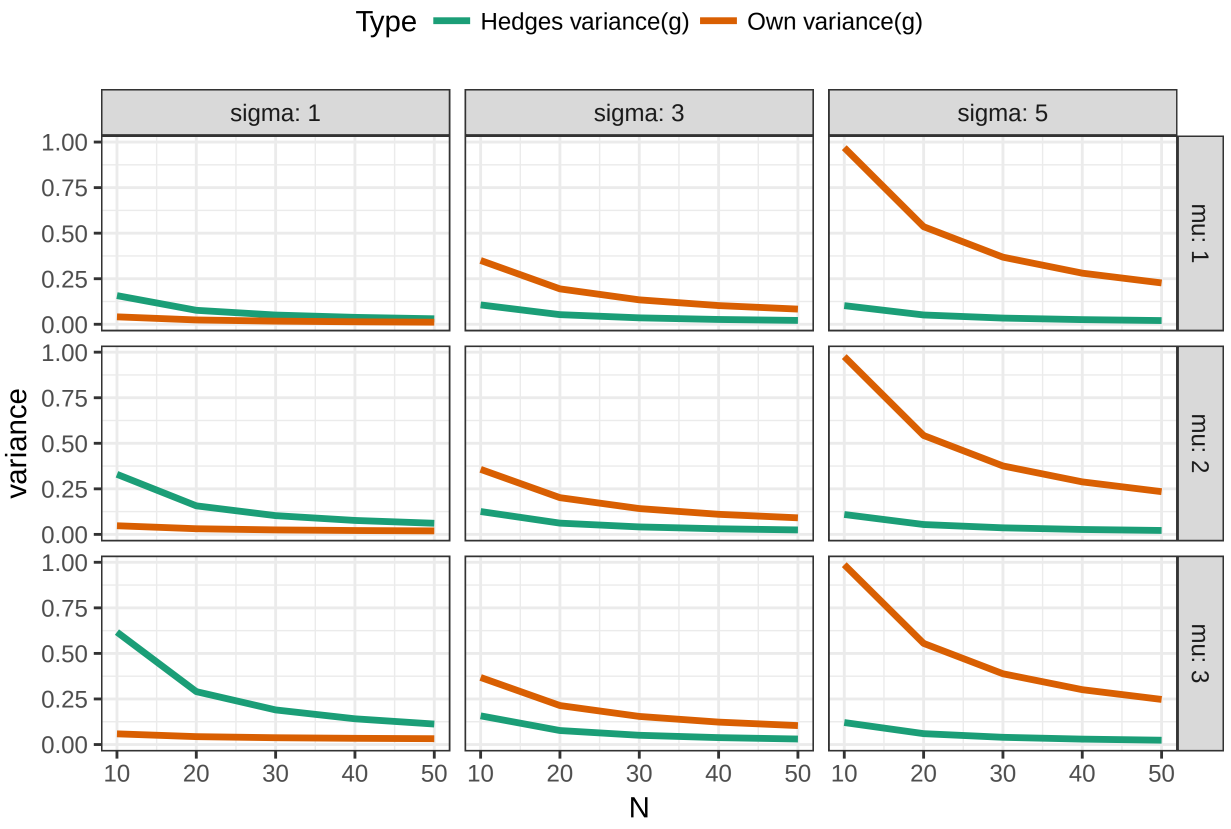
```
mus <- seq(1,3, length.out = 3)
sigmas <- seq(1,5, length.out = 3)
N <- round(seq(10, 50, length.out = 5),0)

# Pairs of combinations
pairs <- expand.grid(mus, sigmas, N)
names(pairs) <- c('mu', 'sigma', 'N')

recValues <- data.frame(matrix(nrow = dim(pairs)[1], ncol = 2))
names(recValues) <- c('ownVarG', 'hedgesVarG')
# For loop over these
for(i in 1:dim(pairs)[1]){
  mu_loop <- pairs[i, 'mu']
  sigma_loop <- pairs[i, 'sigma']
  N_loop <- pairs[i, 'N']

  # Get true g
  true_g_loop <- mu_loop / sigma_loop * corrJ(N_loop)

  ownVar <- ownVarG(mu = mu_loop, sigma = sigma_loop, N_loop)
  hedgeVar <- varHedge(g = true_g_loop, N = N_loop)
  recValues[i,] <- c(ownVar, hedgeVar)
}
compResults <- data.frame(pairs, recValues)
tbl_df(compResults)
```

```
# A tibble: 45 x 5
      mu sigma     N    ownVarG hedgesVarG
   <dbl> <dbl> <dbl>      <dbl>      <dbl>
 1     1     1    10 0.04088864  0.1574411
 2     2     1    10 0.04751678  0.3297646
 3     3     1    10 0.05856368  0.6169703
 4     1     3    10 0.35032274  0.1063823
 5     2     3    10 0.35695088  0.1255294
 6     3     3    10 0.36799778  0.1574411
 7     1     5    10 0.96919094  0.1022976
 8     2     5    10 0.97581908  0.1091906
 9     3     5    10 0.98686598  0.1206788
10     1     1    20 0.02375778  0.0766960
# ... with 35 more rows
```

There are substantial differences!

Not visible in the table, but look at the plot below.

```
compResults %>% gather(key = type, value = variance, 4:5) %>%
  ggplot(., aes(x = N, y = variance, colour = type)) +
  geom_line(aes(colour = type), size = 1.2) +
  facet_grid(mu ~ sigma, labeller = label_both) +
  scale_color_brewer(type = 'qual', palette = 2, name = 'Type',
                     labels = c('Hedges variance(g)',
                                'Own variance(g)')) +
  theme_bw() +
  theme(legend.position = 'top')
```



For the remainder of this report, we use the Hedges' estimator.

## 2.5 Between study variance

The between-study variance is estimated using the DerSimonian and Laird estimator. This estimator is centered around the heterogeneity statistic $Q$. This is given as:

$$Q = \sum_{k=1}^{K} w_k (Y_k - \overline{Y})^2. \tag{18}$$

In this case, $w_k = \sigma_k^{-2}$, $Y_k = g_k$ and $\overline{Y} = \sum_{k=1}^{K} w_k Y_k / \sum_{k=1}^{K} w_k$. Furthermore, $Q \sim \chi^2_{k-1}$. The expected value is given as:

$$\mathrm{E}(Q) = (K-1) + \left( \sum_{k=1}^{K} w_k - \frac{\sum_{k=1}^{K} w_k^2}{\sum_{k=1}^{K} w_k} \right) \tau^2. \tag{19}$$

From here, the DerSimonian and Laird estimator is given as:

$$\hat{\tau}^2 = \max \left( 0, \frac{Q-(K-1)}{\sum_{k=1}^{K} w_k - \frac{\sum_{k=1}^{K} w_k^2}{\sum_{k=1}^{K} w_k}} \right). \tag{20}$$

Truncation is done to prevent a negative variance.

In **R**, we use:

```
# Calculate between-studies variance
tau <- function(Y,W,k){
  C <- sum(W)-(sum(W^2)/sum(W))
  df <- k-1
  Q <- sum(W*Y^2)-sum(W*Y)^2/sum(W)
  if(Q < df){
    T2 <- 0
  }else{
    T2 <- (Q-df)/C
  }
  return (T2)
}

# Calculate weighted mean
wMean <- function(Y,W){
    M <- sum(W*Y)/sum(W)
    return(M)
}
```

## 2.6 Statistical significance testing

Recall that the goal of a meta-analysis is to quantify the population effect. A null and alternative hypothesis can be constructed which take the form of:

$$H_0 : \theta = 0 \tag{21}$$
$$H_a : \theta \neq 0 \tag{22}$$

The null hypothesis can be tested using the Wald-type statistic:

$$\frac{\hat{\theta}}{\sqrt{1/(\sum_{k=1}^{K} W_k)}} \tag{23}$$

$$\Leftrightarrow \frac{M^*}{\sqrt{1/\left(\sum_{k=1}^{K} W_k\right)}}, \tag{24}$$

which has an approximate standard normal distribution. Statistical significance can then be derived under the regular $\alpha$ thresholding levels.

### 2.7 Goal

In this report, I will generate subject data using a normal linear model containing within and between-study variability. We will use the Ordinary Least Squares method to estimate study level parameters. From these, we estimate $g$, the within and between-study variability and then calculate the weighted average.

We should be able to define the true value of the weighted averages. Then we will look at the distribution of the weighted average, the normal approximation, the type I error rate and the observed power.

## 3 Method

We will generate response data for each subject $s$ from study $k$ using the following linear normal model:

$$Y_{sk} = \beta X + \varepsilon_{sk}, \tag{25}$$

where $\varepsilon \sim N(0, \sigma_k^2 + \tau^2)$. In this model, $X$ will only take the value of 1 (for each subject). It is a bit cumbersome to write it like this, but it is done to keep an explicit reference to the GLM models.

The OLS estimate of the study level parameter $\beta$ equals $\overline{Y}$ and its standard error is estimated as $S_y/\sqrt{n}$ where $S_y$ is the unbiased sample standard deviation. The $t$-value equals $\beta/\text{SE}_\beta$.

### 3.1 True values

We will have $K = 20$. The amount of subjects in each study is equally spaced between $10 - 50$. Furthermore, we make pairs between all values of the following parameters:

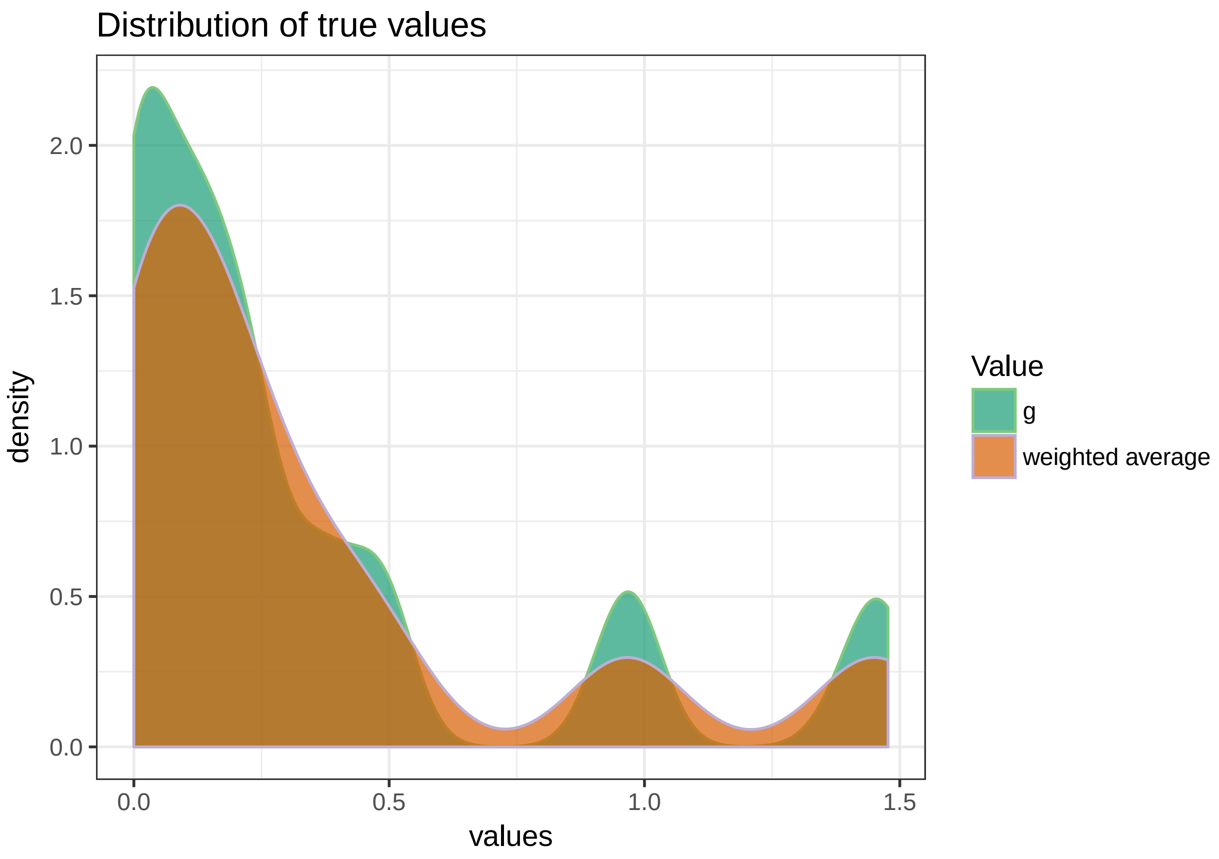
- $\beta \in [0, 0.5, 1, 1.5]$
- $\sigma \in [1, 4, 7]$
- $\tau \in [0, 3.5, 7]$

This results in 36 combinations. For each combination, we calculate the true $t$-value for each study. Then we convert the true $t$-value to Hedges' $g$ per study. We can then calculate the true weighted average. However, as all studies have the same weights($W_k$): $(\sigma^2 + \tau^2)^{-1}$, the weighted average equals an unweighted average.

We thus have:

```r
K <- 20
N <- round(seq(10, 50, length.out = K), 0)
TrueBeta <- 1.5
TrueSigma <- 2
TrueTau <- 1.2
TrueValues <- data.frame(K, N, TrueBeta, TrueSigma, TrueTau) %>%
  mutate(TrueG = (TrueBeta/TrueSigma * corrJ(N = N)))
TrueWMean <- summarise(TrueValues, TrueWMean = mean(TrueG))

print(tbl_df(data.frame('Sample size in study k' = N,
                 'True g' = TrueValues$TrueG)))
TrueWMean
```

```r
# True values for K (number of studies), the sample size per study,
# beta, sigma and tau
K <- 20
N <- round(seq(10, 50, length.out = K), 0)
TrueBeta <- seq(0,1.5, by = .5)
TrueSigma <- seq(1, 7, length.out = 3)
TrueTau <- seq(0, 7, length.out = 3)

# Create combinations of beta, sigma and tau
pairsSim <- expand.grid(TrueBeta, TrueSigma, TrueTau)
  names(pairsSim) <- c('TrueBeta', 'TrueSigma', 'TrueTau')
nPairs <- dim(pairsSim)[1]

# Stack the combinations for K times
stackedPairs <- do.call(rbind, replicate(K, pairsSim, simplify=FALSE))

# Add to data frame
TrueValPair <- data.frame('K' = rep(K, nPairs),
                'N' = rep(N, each = nPairs), stackedPairs) %>% tbl_df()

# Calculate the true value of G and then weighted mean
# Note, it does not make sense to calculate the true
# value of the DerSimonian and Laird estimator. This value is based on
# heterogeneity of the response variable.
# We only have the true values.
TrueValues <- TrueValPair %>%
  mutate(TrueG = (TrueBeta/TrueSigma * corrJ(N = N)),
         TrueWeight = 1/(TrueSigma^2 + TrueTau^2))

# Now calculate true weighted average.
# Note, as the true within study variability is equal for each study,
# all the weights will be equal. Hence weighted average over all studies =
# unweighted average.
TrueWMean <- TrueValues %>% group_by(TrueBeta, TrueSigma, TrueTau) %>%
  summarise(wMean = wMean(Y = TrueG, W = TrueWeight),
            unwMean = mean(TrueG)) %>% ungroup()

# print(tbl_df(data.frame('Sample size in study k' = N,
#                   'True g' = TrueValues$TrueG)))
# TrueWMean

# Create plot
ToPlot_true <- data.frame('values' = c(TrueValues$TrueG,
                          as.numeric(unlist(select(TrueWMean, wMean)))),
           'type' = c(rep('true_g', length(TrueValues$TrueG)),
                      rep('true_wMean', dim(TrueWMean)[1]))) %>% tbl_df()
ggplot(ToPlot_true, aes(x = values, fill = type, colour = type)) +
  geom_density(aes(fill = type, colour = type), alpha = 0.7) +
  scale_color_brewer(name = "Value", labels = c('g', 'weighted average'),
                     type = 'qual', palette = 1) +
  scale_fill_brewer(name = "Value", labels = c('g', 'weighted average'),
                   type = 'qual', palette = 2) +
  ggtitle('Distribution of true values') +
  theme_bw()
```



```
# Summary of true g
filter(ToPlot_true, type == 'true_g') %>% select(values) %>% unlist() %>% as.numeric() %>% summary()
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.00000 0.04898 0.16829 0.33662 0.39121 1.47692
```

```
# Summary of weighted average
filter(ToPlot_true, type == 'true_wMean') %>% select(values) %>% unlist() %>% as.numeric() %>% summary(
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.00000 0.05179 0.17262 0.33662 0.39272 1.45004
```

## 4 Monte Carlo Simulation

We use 1000 Monte Carlo simulations to assess the distribution of $M^*$, the type I error rate and the power. The following **R** code is used to run the simulations.

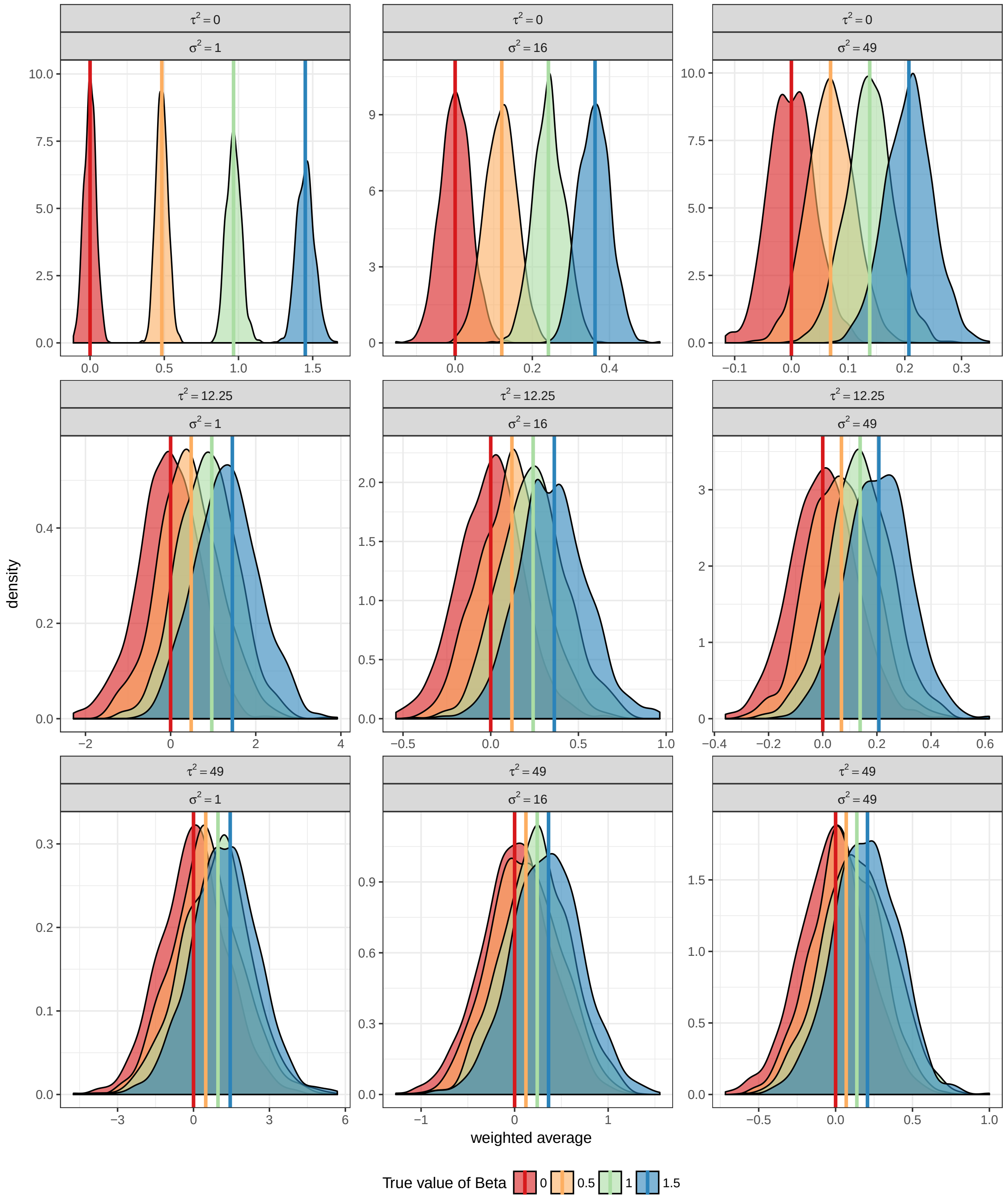
```
# Set the seed
seed <- round(pi, 6)
set.seed(seed)

# Number of simulations
nsim <- 1000

# Stack the combinations for nsim times
```

```r
stackedPairs_nsim <- do.call(rbind,
          replicate(nsim, pairsSim, simplify=FALSE))

# Empty vector for weighted mean and z-statistic
# with the true values for the parameters attached.
sim_Wmean <- data.frame(matrix(nrow = nsim, ncol = 2))
names(sim_Wmean) <- c('wMean', 'Zstat')
simWmean <- cbind(sim_Wmean, stackedPairs_nsim)

# Start Monte Carlo for loop: over simulations and pairs of
# parameters.
for(i in 1:dim(simWmean)[1]){
  # Get the parameters
  sim_TrueBeta <- simWmean[i, 'TrueBeta']
  sim_TrueSigma <- simWmean[i, 'TrueSigma']
  sim_TrueTau <- simWmean[i, 'TrueTau']

  # Empty study vectors
  studG <- NULL
  # Loop over the studies
  for(k in 1:K){
    # Study variance
    StudVar <- rnorm(n = 1, sd = sim_TrueTau)
    # Subject data
    Y <- sim_TrueBeta + rnorm(n = N[k], sd = sim_TrueSigma) + StudVar
    # Study parameters: t- value
    studT <- coef(summary(lm(Y ~ 1)))[3]
    # Study g: using conversion
    studG <- c(studG, (studT / sqrt(N[k]) * corrJ(N[k])))
  }
  # Estimate tau
  estTau <- data.frame(N, studG) %>%
    mutate(withinVar = varHedge(g = studG, N = N)) %>%
      summarise(tau = tau(Y = studG, W = (1/withinVar), k = K))

  # Now calculate weighted average and Z-statistic
  simWmean[i,c(1:2)] <- data.frame(N, studG, estTau) %>%
    mutate(withinVar = varHedge(g = studG, N = N),
           weights = 1/(withinVar + tau)) %>%
    summarise(wMean = wMean(Y = studG, W = weights),
              Zstat = (wMean/sqrt((1/sum(weights)))))
}
```

## 5 Results

### 5.1 Distribution of weighted average

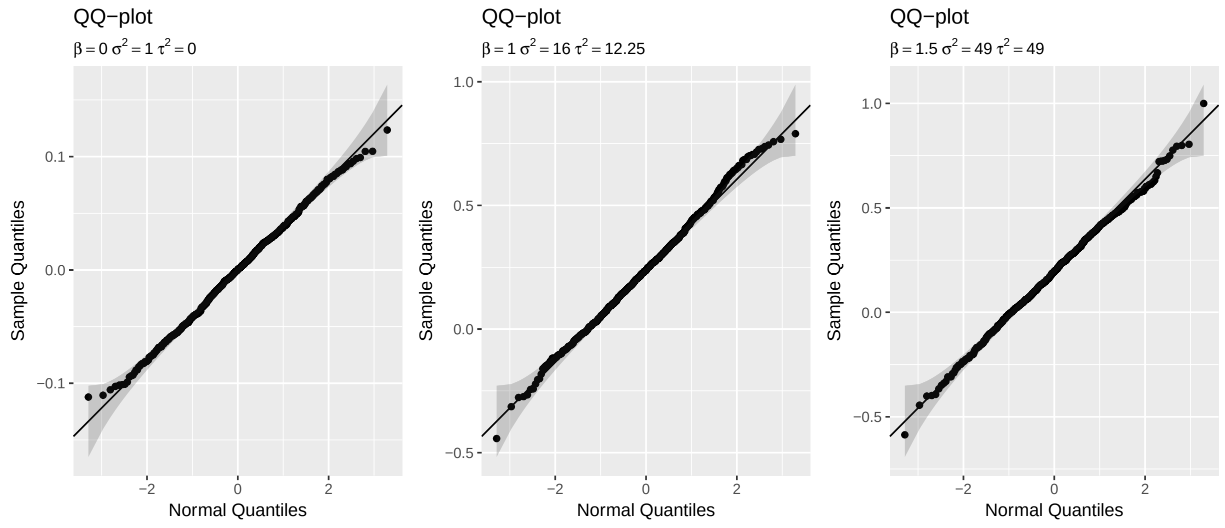
We start by looking at the distributions of the simulated weighted averages. Each panel corresponds to a combination of $\sigma^2$ and $\tau^2$. The vertical lines correspond to the true values of the weighted average for each $\beta$. The distribution is fairly centered around the true value which indicates an unbiassed performance of the estimator.



## 5.2 Normal approximation

We assess the normal approximation of $M^*$ by looking at the normal QQ-plot over the Monte Carlo simulations. We only plot the QQ-plots for some combinations between $\beta$, $\sigma$ and $\tau$. Overal, the performance is decent though we observe some deviations in the tails.

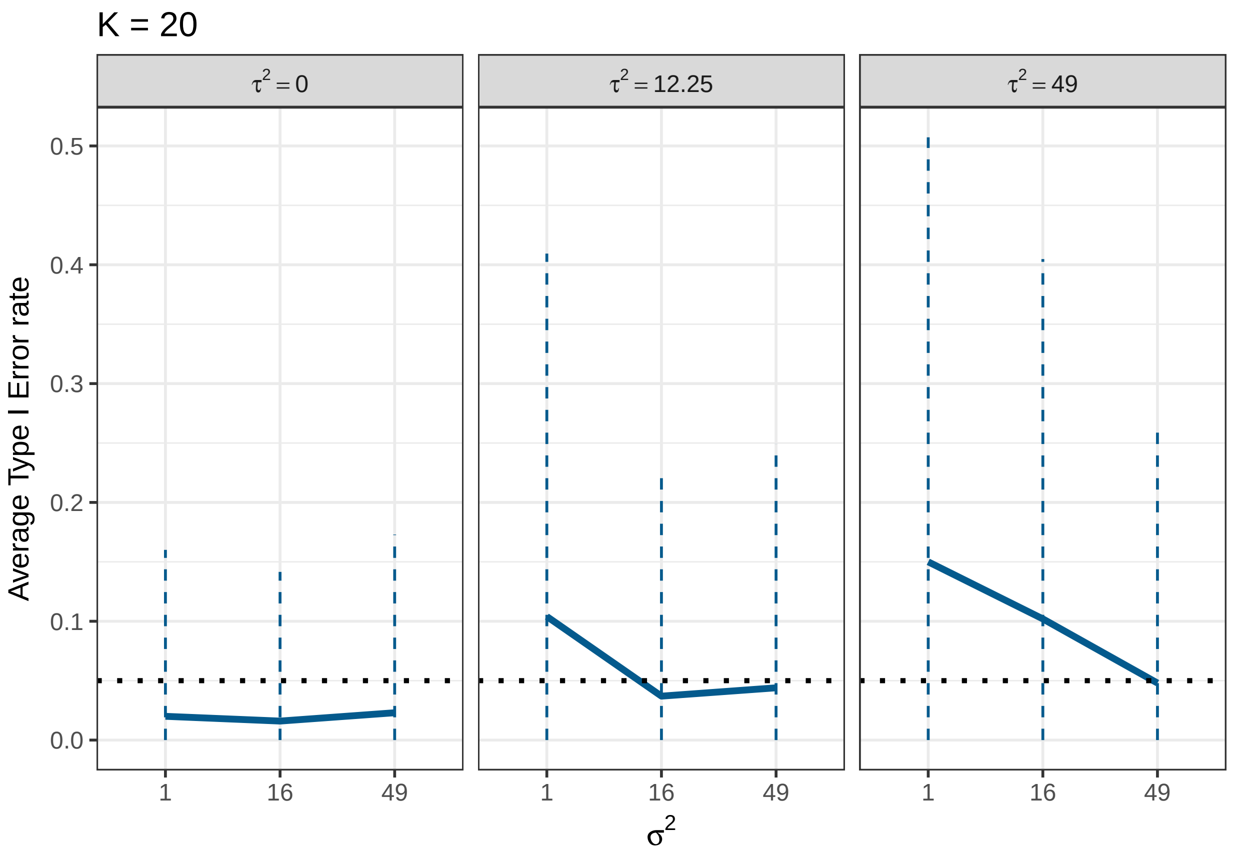
```r
# Just plotting QQ-plots for three parameter values
Selection <- data.frame(beta = TrueBeta[c(1,3,4)],
                        sigma = TrueSigma,
                        tau = TrueTau)
for(i in 1:dim(Selection)[1]){
  simWmean %>% filter(TrueBeta == Selection[i, 'beta'] &
                    TrueSigma == Selection[i, 'sigma'] &
                    TrueTau == Selection[i, 'tau']) %>%
    select(wMean) %>%
    unlist(.) %>% as.numeric() %>%
  gg_qq(x = ., title =
    bquote(beta ==  .(Selection[i, 'beta']) ~
          sigma^2 ==  .(Selection[i, 'sigma']^2) ~
          tau^2 ==  .(Selection[i, 'tau']^2))) %>%
    print()
}
```



## 5.3 Type I error rate

To assess, the type I error rate, we focus on the $\beta = 0$. We test the null hypothesis of no effect under $\alpha = 0.05$. Note that we use a two-sided alternative hypothesis. Interestingly, the average type I error rate is considerably above nominal level of $\alpha = 0.05$ when the between-study variance is much larger than within study variability (see $\sigma^2 = 1$ with $\tau^2 = 49$). However, this improves when within study variance increases again. The same trend, though less pronounced is observed when $\tau^2 = 12.25$.

```r
ToPlot %>% tbl_df() %>%
  filter(TrueBeta == 0) %>%
  mutate(Pval = 2 * pnorm(Zstat, lower.tail = FALSE),
         TypeIer = ifelse(Pval < 0.05, 1, 0)) %>%
  group_by(TrueSigma, TrueTau) %>%
  summarise(AvgTypeIErr = mean(TypeIer),
            SDTypeI = sd(TypeIer),
            yStart = max(0, AvgTypeIErr - SDTypeI),
            yEnd = min(1, AvgTypeIErr + SDTypeI)) %>%
  ggplot(., aes(x = TrueSigma, y = AvgTypeIErr, group = 1)) +
    geom_line(size = 1.2, colour = '#045a8d') +
    geom_hline(yintercept = 0.05, linetype = 'dotted', size = .9)+
    geom_segment(aes(x = TrueSigma,
                 xend = TrueSigma,
                 y = yStart,
                 yend = yEnd),
                 colour = '#045a8d',
                 linetype = 'dashed') +
    facet_wrap( ~ TrueTau, labeller = label_parsed) +
  scale_y_continuous('Average Type I Error rate') +
  scale_x_discrete(expression(sigma^2), labels = TrueSigma^2) +
  ggtitle(paste0('K = ', K)) +
  theme_bw()
```

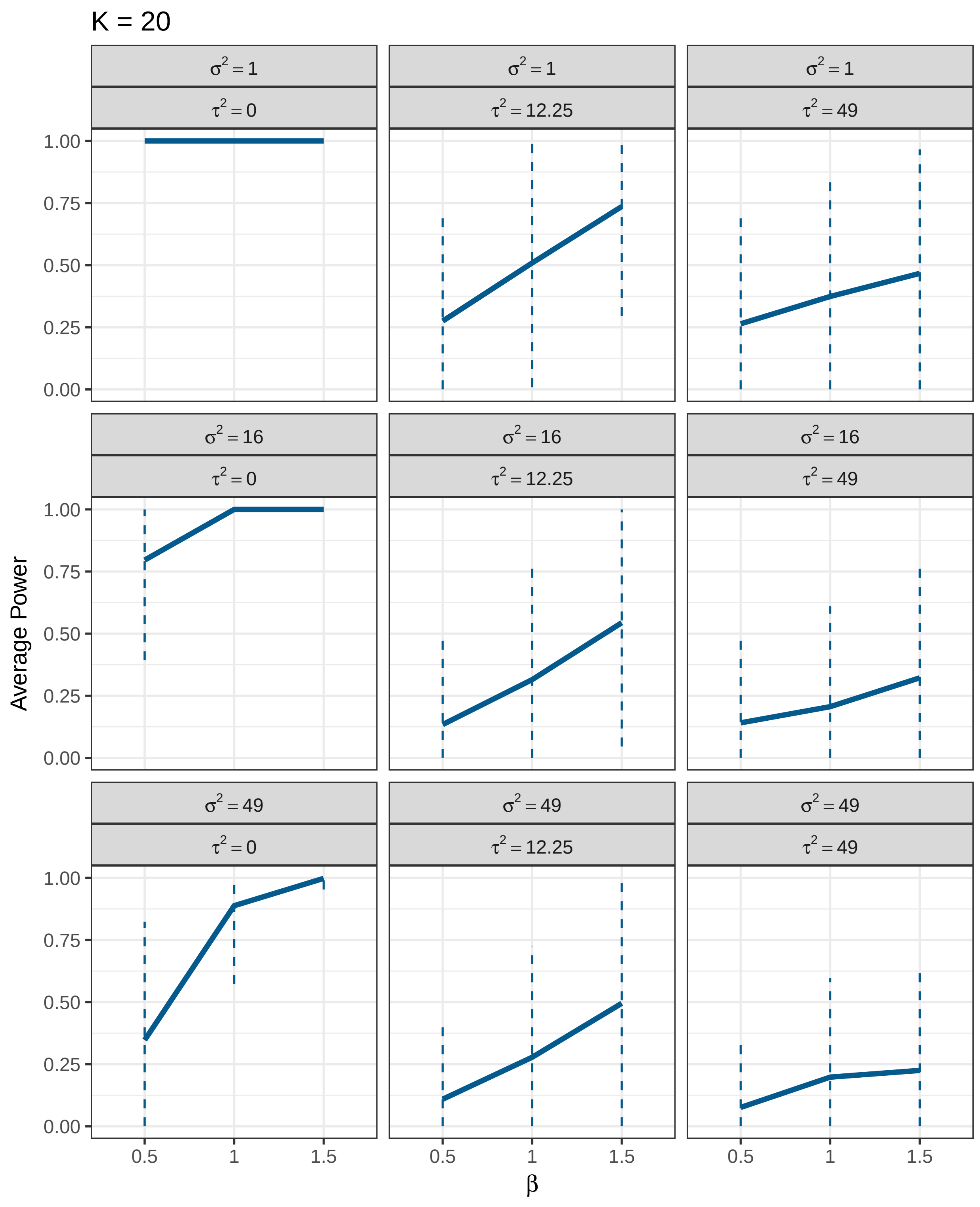


## 5.4 Power

To assess power, we look at $\beta \in [0.5, 1, 1.5]$. As expected, the power decreases when both within and between study variability increases. Power also increases with the true effect size. Interestingly, between within- and between-study variance, the drop in power for higher effect sizes is mostly dominated by increasingly between-study variance. This is shown as the difference between increasing levels of $\sigma^2$ on the observed power is small (except when we look at small effect sizes and $\tau = 0$). However, as $\tau^2$ increases, the observed power decreases considerably for modest and larger true effect sizes.

```
ToPlot %>% tbl_df() %>%
  filter(TrueBeta != 0) %>%
  mutate(Pval = 2 * pnorm(Zstat, lower.tail = FALSE),
         Power = ifelse(Pval < 0.05, 1, 0)) %>%
```

```r
group_by(TrueBeta, TrueSigma, TrueTau) %>%
summarise(AvgPower = mean(Power),
          SDPower = sd(Power),
          yStart = max(0, AvgPower - SDPower),
          yEnd = min(1, AvgPower + SDPower)) %>%
  ggplot(., aes(x = factor(TrueBeta), y = AvgPower, group = 1)) +
  geom_line(size = 1.2, colour = '#045a8d') +
  geom_segment(aes(x = factor(TrueBeta),
                   xend = factor(TrueBeta),
                   y = yStart,
                   yend = yEnd),
               colour = '#045a8d',
               linetype = 'dashed') +
  facet_wrap(TrueSigma ~ TrueTau, labeller = label_parsed) +
scale_y_continuous('Average Power') +
scale_x_discrete(expression(beta), labels = TrueBeta[-1]) +
ggtitle(paste0('K = ', K)) +
theme_bw()
```



## 6 Conclusion

We have not investigated the effect of increasingly study set sizes ($K$, the amount of studies in the meta-analysis).