Electronic Supplementary Material for
Renkewitz, F., & Keiner, M. (2019). How to detect publication bias in psychological research: A comparative evaluation of six statistical methods. ***Zeitschrift für Psychologie*****. https://doi.org/10.1027/2151-2604/a000386**

Contents

List of Figures

1 Effect sizes and their variance

The effect size Cohen's *d* for each study can be computed from its *t*-value:

$$d = t\sqrt{\frac{n_1 + n_2}{n_1 n_2}} \approx t\frac{2t}{n_1 + n_2}.$$

where n_1 and n_2 are the sample sizes in each group.

We calculated Hedges' g for each study from its Cohen's *d*:

$$g = d \cdot J.$$

J is a correction factor used to obtain unbiased estimates in small samples (Schmidt & Hunter, 2014). *J* is given by:

$$J = 1 - \frac{3}{4(n_1 + n_2 - 2) - 1}.$$

The variance of g was calculated as

$$V_g = \frac{n_1 + n_2}{n_1 n_2} + \frac{d^2}{2(n_1 + n_2)} \cdot J^2.$$

Begg's rank correlation and PET were additionally performed after a variance-stabilizing transformation on Hedges' g values (Hedges & Olkin, 1985). This transformation is given by:

$$h(g) = \sqrt{2}\, sinh^{-1}\left(\frac{g}{2\sqrt{2}}\right) \; or \; h(g) = \sqrt{2} * log\left(\frac{g}{2\sqrt{2}} + \sqrt{\left(\frac{g}{2\sqrt{2}}\right)^2 + 1}\right).$$

The variance of the effect size *h* is:

$$V_h = \frac{1}{n_1 + n_2}.$$

2 Technical details on the detection methods and their implementation

The following section provides information on the functional principles and the implementation of the detection methods.

2.1 Funnel plot

Several methods for the detection of publication bias are based on the funnel plot. A funnel plot is a simple scatter plot of effect sizes on the *x*-axis against some measure of sample sizes on the *y*-axis (such as precision, variance or standard error). In Figure S1 the standard error is shown on an inverted *y*-axis. Thus, large studies that estimate the true effect precisely appear in the upper part of this plot; in contrast, less precise and smaller studies are located at the base. If there is no bias in the dataset, the distribution of observed effect sizes is symmetrically centred around the true population effect (and, hence, the emerging plot looks like a funnel). However, if there is publication bias and nonsignificant results are suppressed, the shape of the distribution changes. Small studies (with large standard errors) that observe small effect sizes will be missing. Thus, there is a lack of studies in the lower left-hand corner of the distribution, which will now appear asymmetric. This also causes an association between effect sizes and their standard errors.

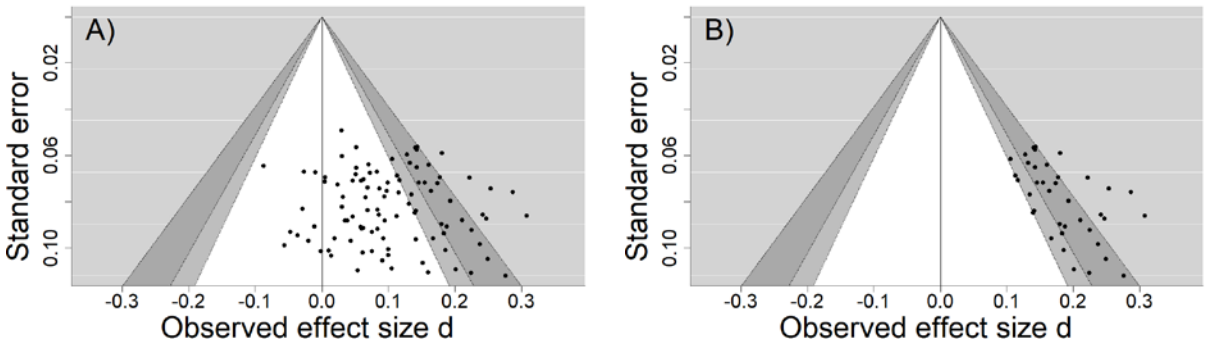

Figure S1. Funnel plots in the absence of publication bias (A) and after the exclusion of nonsignificant results (B). The contours in both plots represent two-tailed significance thresholds of 10%, 5% and 1%.

2.2 Begg's rank correlation

Begg's rank correlation (Begg & Mazumdar, 1994) uses Kendall's tau to measure the correlation between standardized effect sizes and their variances. The standardized effect size of study *i* (T_i^*) is defined as:

$$T_i^* = \frac{T_i - \bar{T}_\bullet}{\sqrt{\tilde{v}_i^*}},$$

where $\bar{T}_\bullet = \frac{\sum_i^k \frac{T_i}{SE_i^2}}{\sum_i^k \frac{1}{SE_i^2}}$, T_i is the observed effect size of study *i*, SE_i is the standard error of the observed effect size, and $\tilde{v}_i^* = SE_i^2 - \left(\sum_i^k \frac{1}{SE_i^2}\right)^{-1}$.

Bias is indicated when Kendall's tau is statistically significant. Due to power considerations several authors recommend using a liberal significance criterion of α = 0.1 for this test (Egger et al., 1997; Borenstein et al., 2009; Sterne & Egger, 2005). We kept with the usual threshold of α = 0.05 (one-tailed) to ensure comparability with the detection rates of all other investigated methods. Furthermore, in each simulated meta-analysis, we conducted Begg's rank correlation with two different effect size metrics: Hedges' *g* and *h* (see section 1 on effect sizes). Kromrey & Rendina-Gobioff (2006) and Pustejovsky & Rodgers (2018) observed inflated Type I error rates for Begg's rank correlation with Cohen's *d* and Hedges' *g* when the true effect size was different from zero. Hence, the *h* values were used as input to test whether the method keeps the nominal α-level when variances of effect sizes are stabilized. To perform Begg's rank correlation we employed the function "ranktest" from the metafor package (Viechtbauer, 2010).

2.3 PET & PEESE

The precision effect test (PET; Stanley & Doucouliagos, 2014), is a weighted regression of effect sizes on their standard errors:

$$T_i = b_0 + b_1 \cdot se_i + e_i, \quad \text{weighted by } \frac{1}{se_i^2}.$$

In the absence of bias, observed effect sizes and their standard errors are expected to be uncorrelated (see Figure S1) and the regression slope b_1 should be zero. Accordingly, a statistically significant slope indicates bias.

Additionally, to correct for publication bias, the intercept b_0 can be used as an adjusted estimate of the true effect size. However, Stanley & Doucouliagos (2014) have shown that b_0 systematically underestimates the true effect size if the true effect is non-zero. Therefore, they suggested the precision effect estimate with standard error (PEESE) as an alternative method for the correction of bias. With this method, effect sizes are regressed not on their standard errors but their variances:

$$T_i = b_0 + b_1 \cdot se_i^2 + e_i, \quad \text{weighted by } \frac{1}{se_i^2}.$$

With regard to the test for publication bias, however, both methods yielded virtually identical results. Therefore, we report only about the Type I error rate and power of PET. To estimate PET we employed the function “lm”. We tested the regression slope b_1 at $\alpha = 0.05$ (one-tailed). Furthermore, as with Begg’s rank correlation, we assessed the performance of PET with Cohen’s *d* and *h* scores. With Cohen’s *d* and Hedges’ *g* the method is known to produce inflated Type I error rates (Sterne, J., Gavaghan, D., & Egger, M., 2000; Pustejovsky & Rodgers, 2018). This occurs as estimates of the variance of *d* and *g* depend on the estimate of the effect sizes themselves (see section 1).

2.4 Trim and fill

The trim and fill method, pioneered by Duval and Tweedie (2000a), assesses asymmetry in the univariate frequency distribution of effect sizes. The method relies on the assumption that the *j* studies with the smallest observed effect sizes are excluded. Based on this assumption, the method estimates the number *j* of missing studies. Duval and Tweedie (2000a) originally suggested three non-parametric estimators (R_0, L_0 or Q_0), but later (Duval, 2005) warned against the use of Q_0. Furthermore, a formal test of the null hypothesis that $j = 0$ was suggested only for the estimator R_0. We used the function "trimfill" from the metafor package (Viechtbauer, 2010) to estimate *j* with R_0. The null hypothesis is rejected if $j > 3$ independently of the number of studies included in a meta-analysis (Duval and Tweedie, 2000b).

2.5 P-uniform

P-uniform (van Assen, van Aert, and Wicherts, 2015) uses only *p*-values of significant primary studies as input. When the true effect size is zero, the distribution of these *p*-values is uniform. With increasing true effect size (and increasing power of primary studies) this distribution becomes increasingly right-skewed. However, *p*-values that are computed conditional on the true effect size θ are generally uniformly distributed. In the bias test of *p*-uniform, $p_i^{\hat{\theta}}$ values are determined that are conditional on the meta-analytic estimate $\hat{\theta}$ of the true effect size θ (using a fixed-effect model). $p_i^{\hat{\theta}}$ represents the probability of observing effect $\hat{\theta}_i$ or larger in study *i* when the true effect size is $\hat{\theta}$ and a significant result is obtained. If $\hat{\theta} = \theta$ the distribution of the $p_i^{\hat{\theta}}$ values can be expected to be uniform.

To test this null hypothesis $p_i^{\hat{\theta}}$ values are combined using Fisher's method:

$$L^{\hat{\theta}} = -\sum_{i=1}^{k} \ln(p_i^{\hat{\theta}}),$$

where *k* is the number of studies (several alternative estimators in *p*-uniform are described by van Assen et al., 2015, and van Aert et al., 2016). The test statistic $L^{\hat{\theta}}$ is gamma distributed with *k* and 1 degrees of freedom, denoted by $\Gamma(k, 1)$. We tested for left-skewness in the distribution of $p_i^{\hat{\theta}}$ values (one-tailed, on the left-hand side of the Γ-distribution) at $\alpha = 0.05$.

To perform *p*-uniform we employed the function "puniform" from the puniform package (van Aert, 2017). By default, the puniform function assumes that two-tailed tests are applied to primary studies. We conducted one-tailed tests on our simulated studies with effect sizes in the right-tail of the distribution. Thus, we used a significance level of 0.1 with the 'right' side command.

2.6 Test of insufficient variance

The test of insufficient variance (TIVA) was developed by Schimmack (2014). In a first step, *p*-values of primary studies are converted into *z*-scores using the standard normal distribution. Observed *z*-scores should follow a normal distribution with a variance of 1 around their non-centrality parameter. For a given study, the non-centrality parameter depends on its sample size and the true effect size. Thus, when all studies included in a meta-analysis investigate the same true effect and have the same sample size the variance of *z*-scores can be expected to be 1. A variance larger than 1 can and will arise when sample sizes vary or when true effect sizes differ (i.e., when there is heterogeneity, $\tau > 0$). Thus, an unbiased set of *z*-scores should have a minimum variance of 1. A variance smaller than 1 can occur if *p*-values are selectively reported or primary studies are affected by *p*-hacking. For a test of the null hypothesis that the variance of observed *z*-scores is 1 Schimmack (2014) provides the following test statistic:

$$\chi^2 = observed\ variance(z) \cdot (k-1) \qquad \text{with} \quad df = k-1,$$

where *k* is the number of *z*-scores (or *p*-values) included. Again, we tested for insufficient variance (one-tailed, on the left-hand side of the χ^2-distribution) at $\alpha = 0.05$.

2.7 Test for an excess of significant findings

The test for an excess of significant findings (TES) was originally suggested by Ioannidis and Trikalinos (2007). The test compares the observed number of significant results in a set of studies to the expected number of significant results. The expected number is given by the average power of the studies. Obviously, to estimate power an estimate of the true effect size is required. Francis (2013) suggested three different ways to obtain this estimate. 1.) Using the meta-analytic

effect size estimate across all available studies. 2.) Using each study's reported effect size to calculate post-hoc power of all individual studies. 3.) Using a random effects model to estimate power. In this case, the calculation of power is again based on a pooled effect size estimate but, additionally, takes variability in true effect sizes into account. Hence, in this case an estimate of τ is required.

In a simulation study Francis (2013) found that TES produces inflated Type I error rates when post-hoc power of individual studies is used. We are not aware of an application of TES that employed a random effects model to estimate power. Hence, we based power calculations on the meta-analytic effect size estimate (expressed as Hedges' g) across all available studies and a fixed effect model.

Following a suggestion of Francis (2013), we compared the observed and expected number of significant studies with Fisher's exact test when the meta-analytic study set was small ($k \leq 10$). The probability of a particular pattern of significant and nonsignificant studies included in a meta-analysis can be computed from the product of the power and the probability β of Type II errors of individual studies:

$$Prob(a) = \prod_{i=1}^{M} (1 - \beta_i)^{a(i)} \ \beta_i^{(1-a(i))}$$

where $a(i) = 1$ if study *i* is significant and $a(i) = 0$ if it is not significant.

For a given number of rejections of the null hypothesis in a set of *M* studies there are different possible combinations. Therefore, the probability of *M* studies including an observed number *O* of significant studies or more is given by:

$$P_C = Prob(\geq O \ signficant \ studies) = \sum_{k=O}^{M} \sum_{j=1}^{MC_K} Prob(a_j)$$

where MC_K is the number of different possible combinations of a number of significant studies and j is the index of different numbers of significant results.

In larger study sets a χ^2test was used for the comparison of the observed and expected number of significant results:

$$\chi^2(1) = \frac{(O - E)^2}{E} + \frac{(O - E)^2}{M - E}$$

where E is the expected number of significant studies. The expected number can be calculated from the different power values of each study:

$$E = \sum_{i=1}^{M} (1 - \beta_i)$$

Both procedures were implemented as one-tailed tests with $\alpha = 0.05$.

3 Figures

3.1 Percentage of significant results among published studies

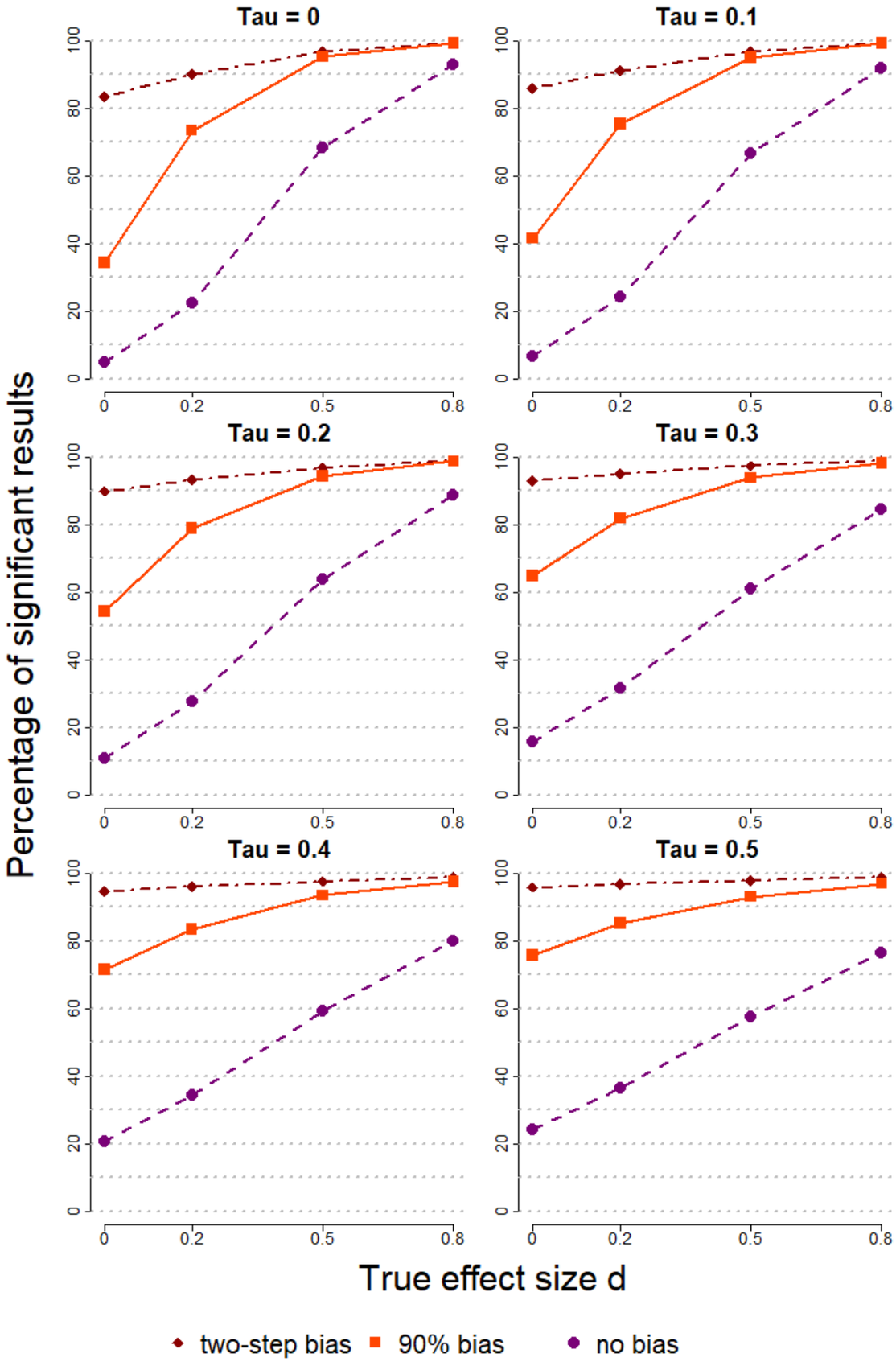

Figure S2. Percentage of significant results among published studies in three selection conditions under different levels of heterogeneity. In the no bias condition these percentages correspond to the average power of simulated primary studies.

3.2 Metaanalytic estimates

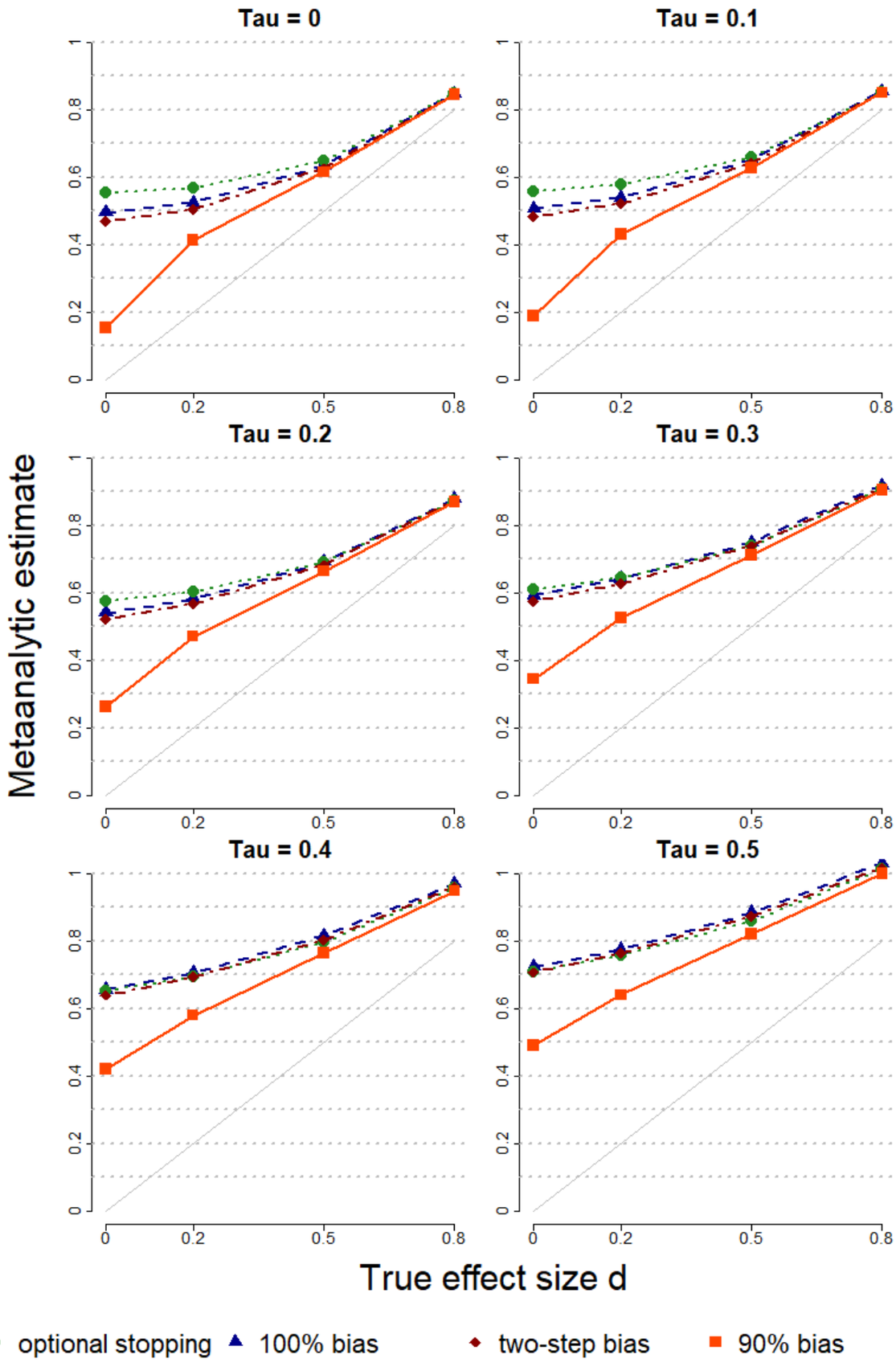

Figure S3. Metaanalytic effect size estimates in four selection conditions under different levels of heterogeneity. The dotted black lines indicate optimal estimates that are almost exactly matched when there is no publication bias (not shown).

3.3 Type I error rates

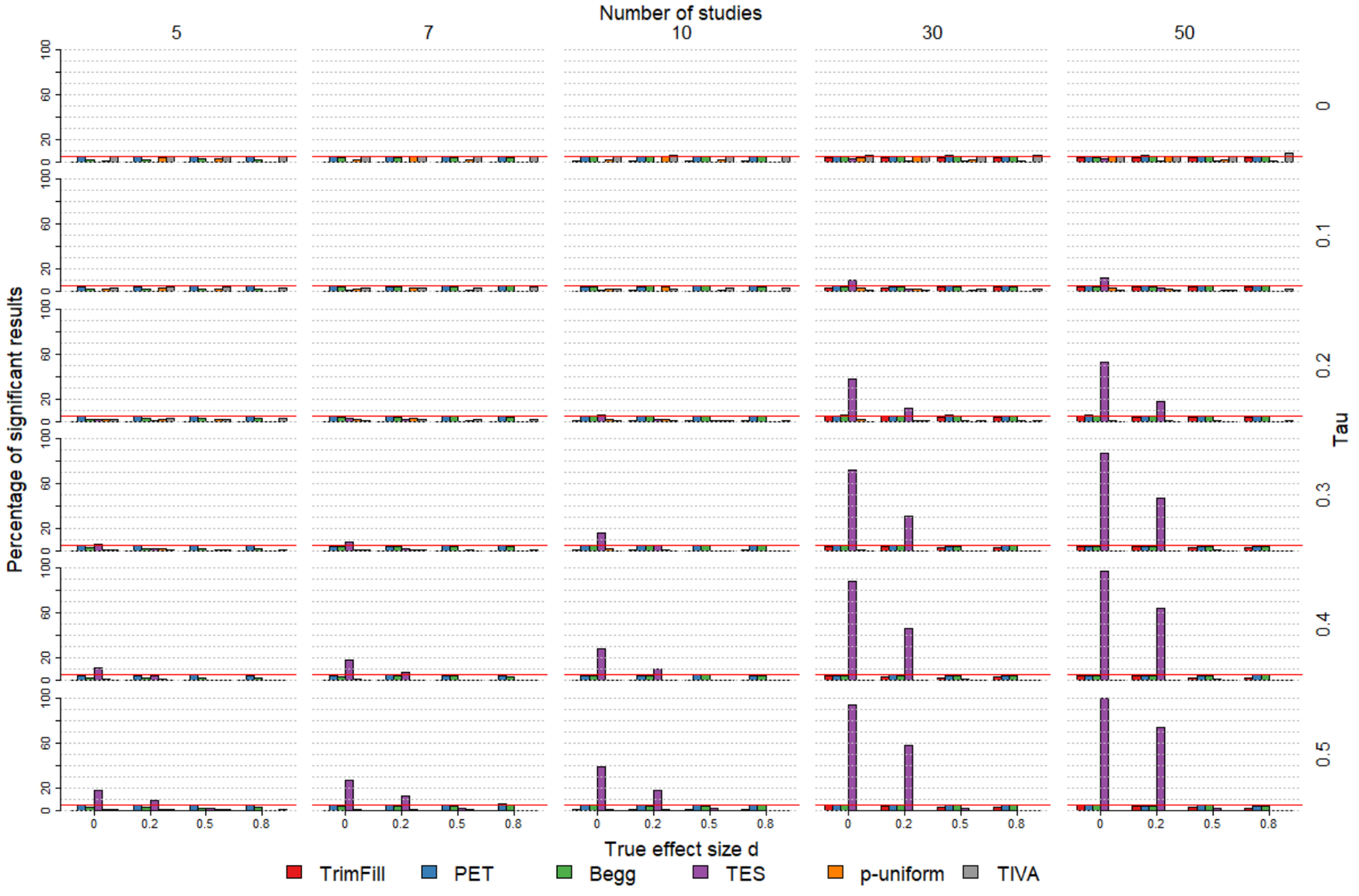

Figure S4. Type I error rates of six detection methods as a function of true effect size, heterogeneity, and number of studies.

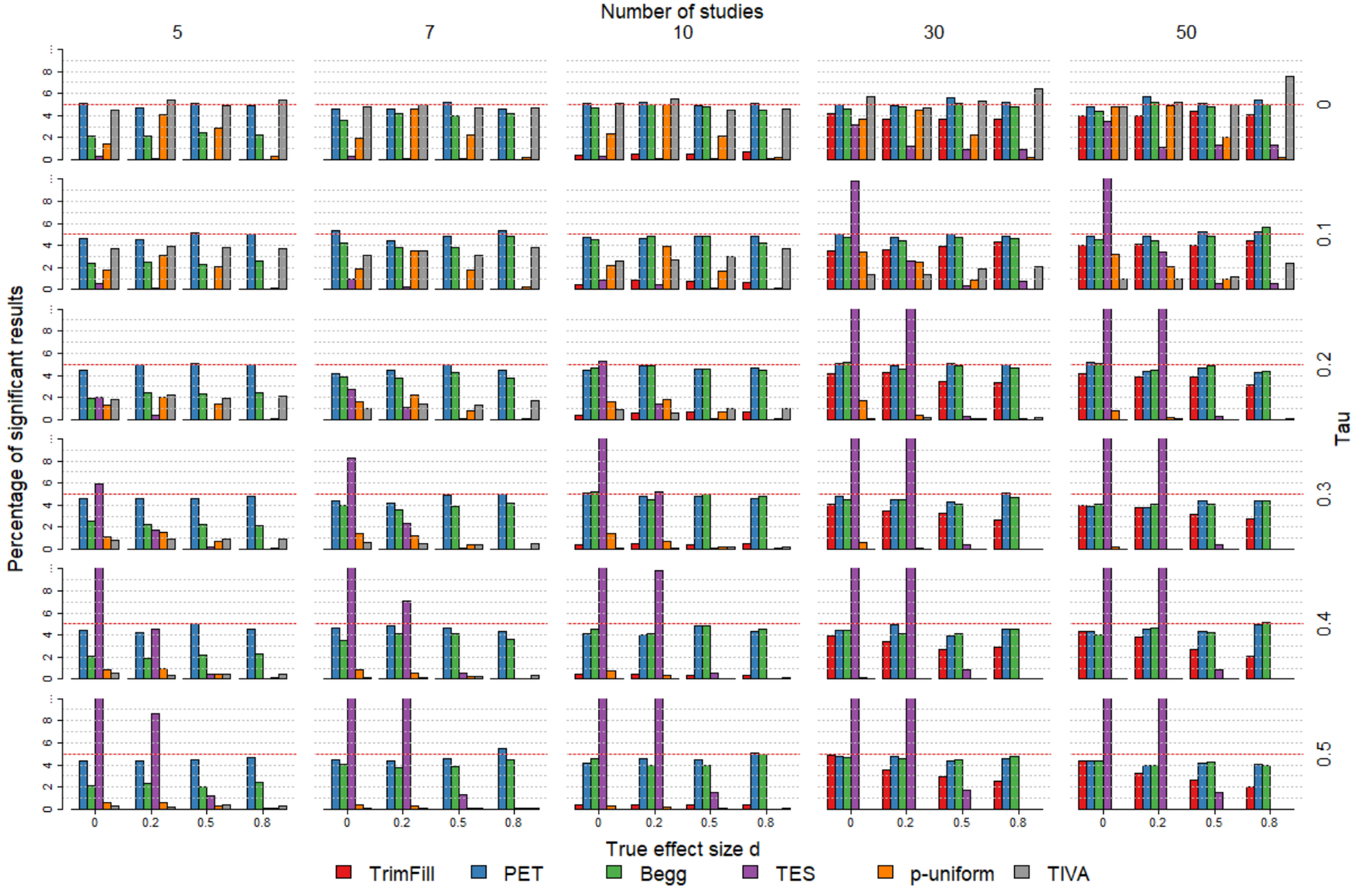

Figure S5. Type I error rates of six detection methods as a function of true effect size, heterogeneity, and number of studies. Error rates of TES exceeding 10% are truncated.

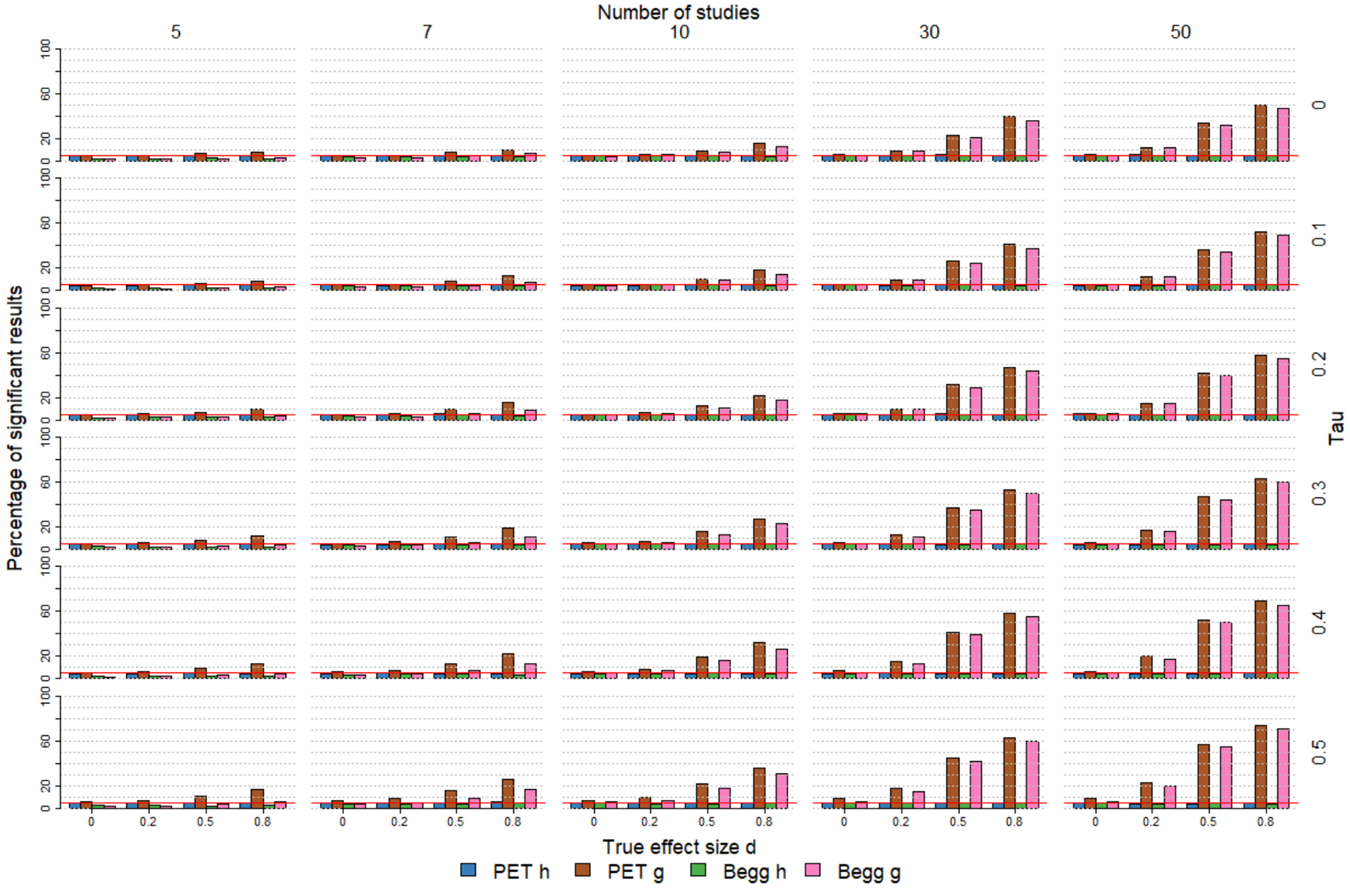

Figure S6. Type I error rates for Begg's rank correlation and PET with different effect size measures as input (Hedges' *g* or *h*)

3.4 *Power at different levels of heterogeneity (τ)*

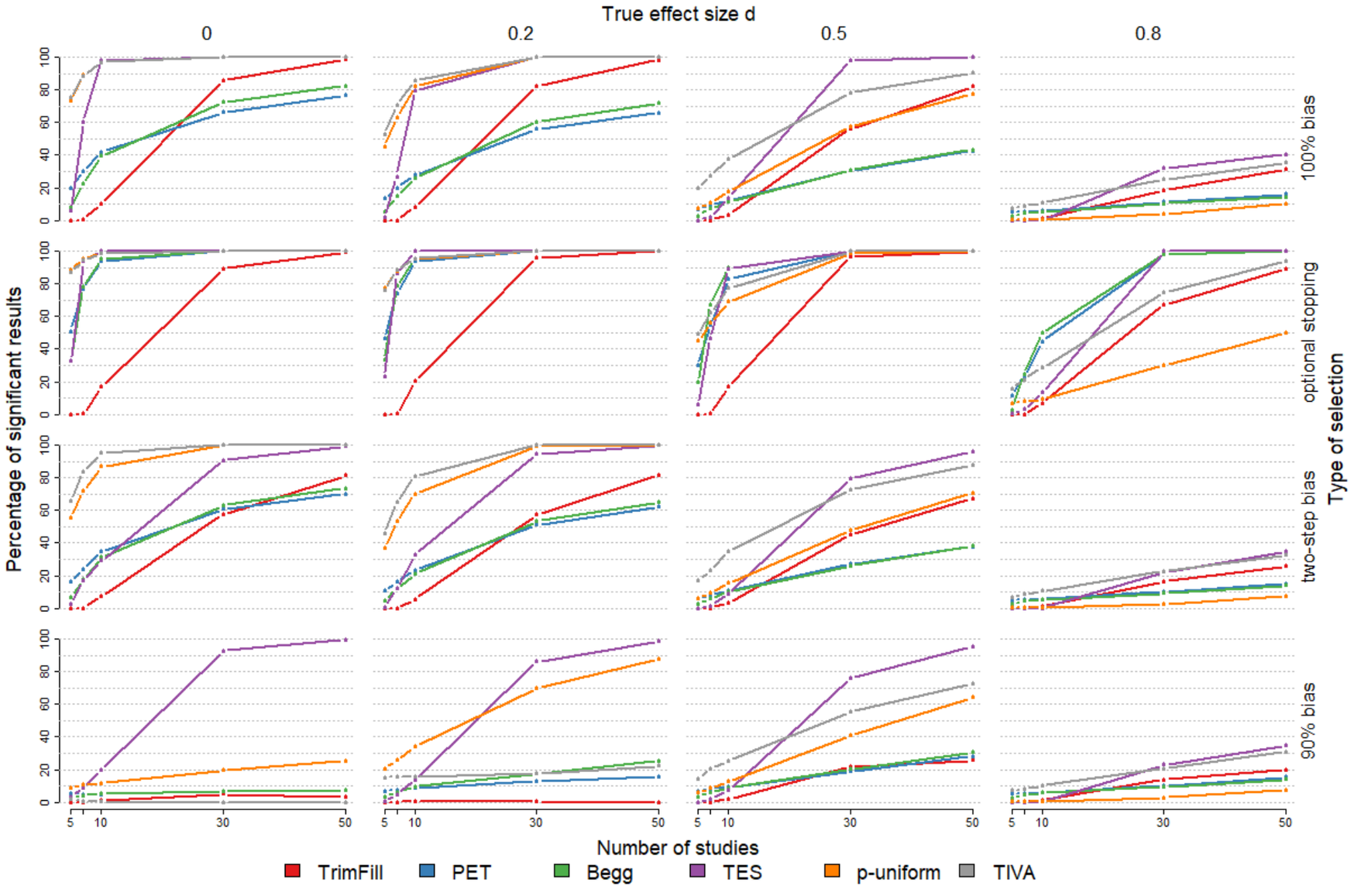

Figure S7. Power of six detection methods at **τ = 0** as a function of selection condition, true effect size and number of studies.

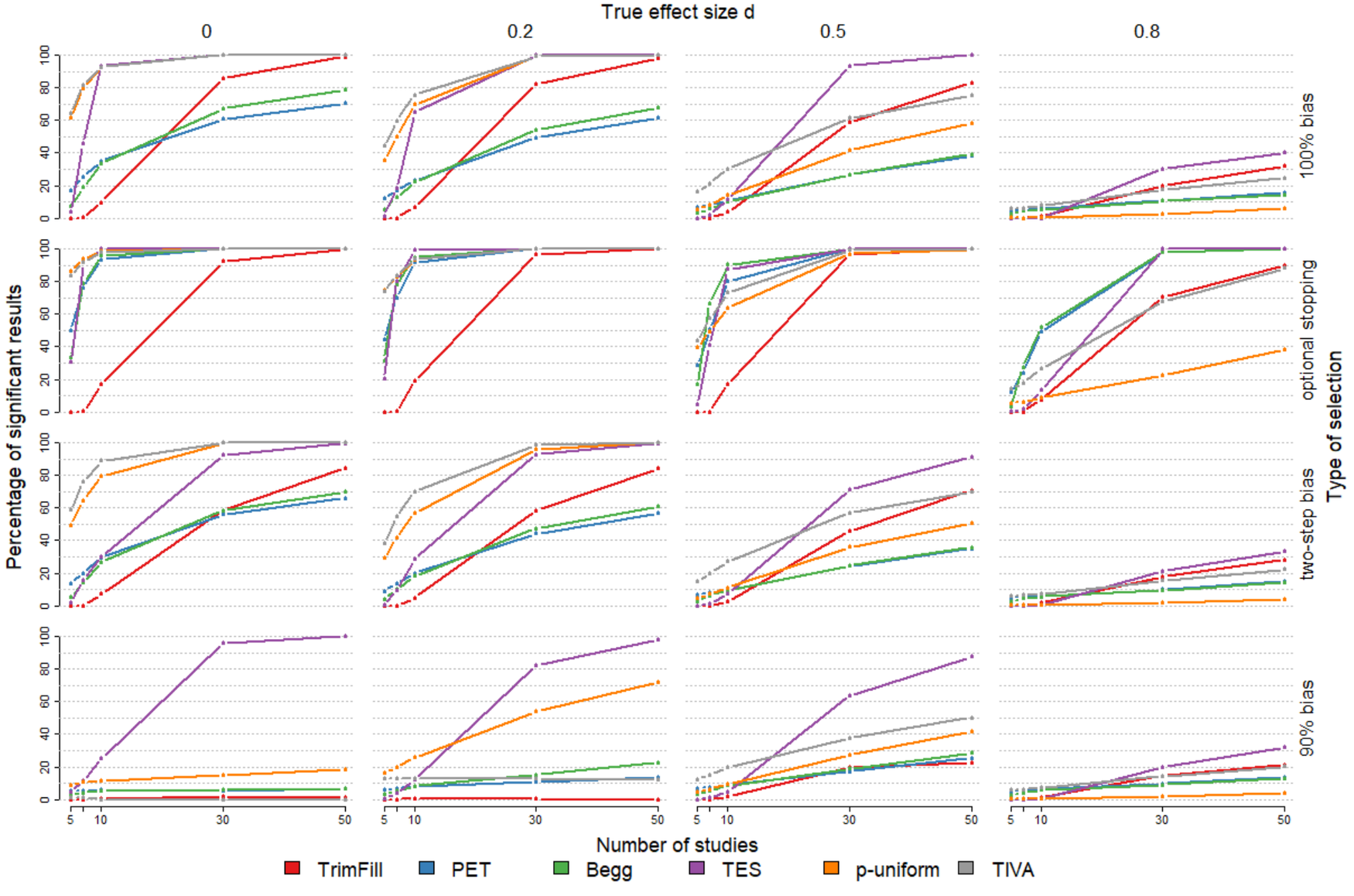

Figure S8. Power of six detection methods at **τ = 0.1** as a function of selection condition, true effect size and number of studies.

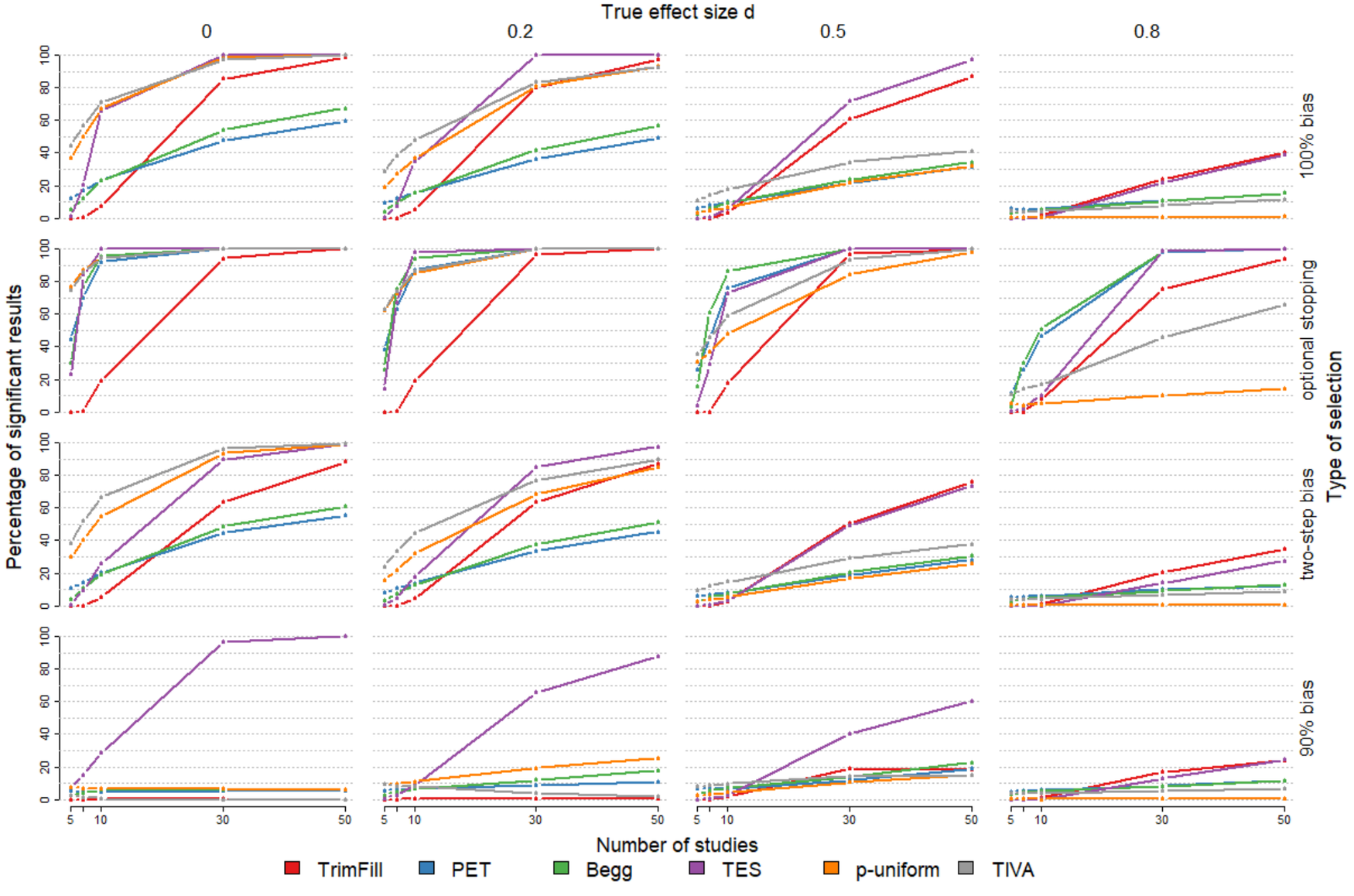

Figure S9. Power of six detection methods at **τ = 0.2** as a function of selection condition, true effect size and number of studies.

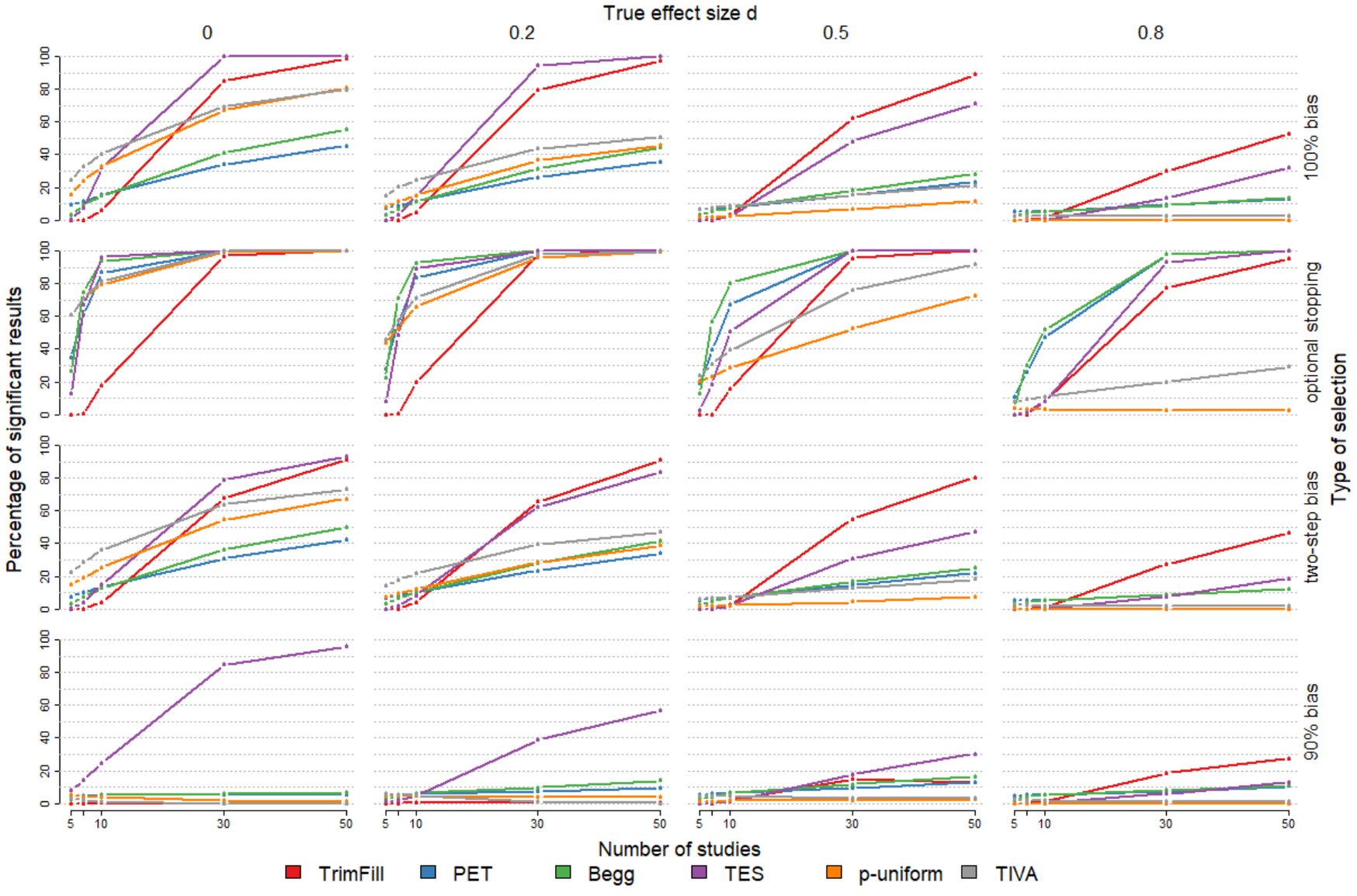

Figure S10. Power of six detection methods at **τ = 0.3** as a function of selection condition, true effect size and number of studies.

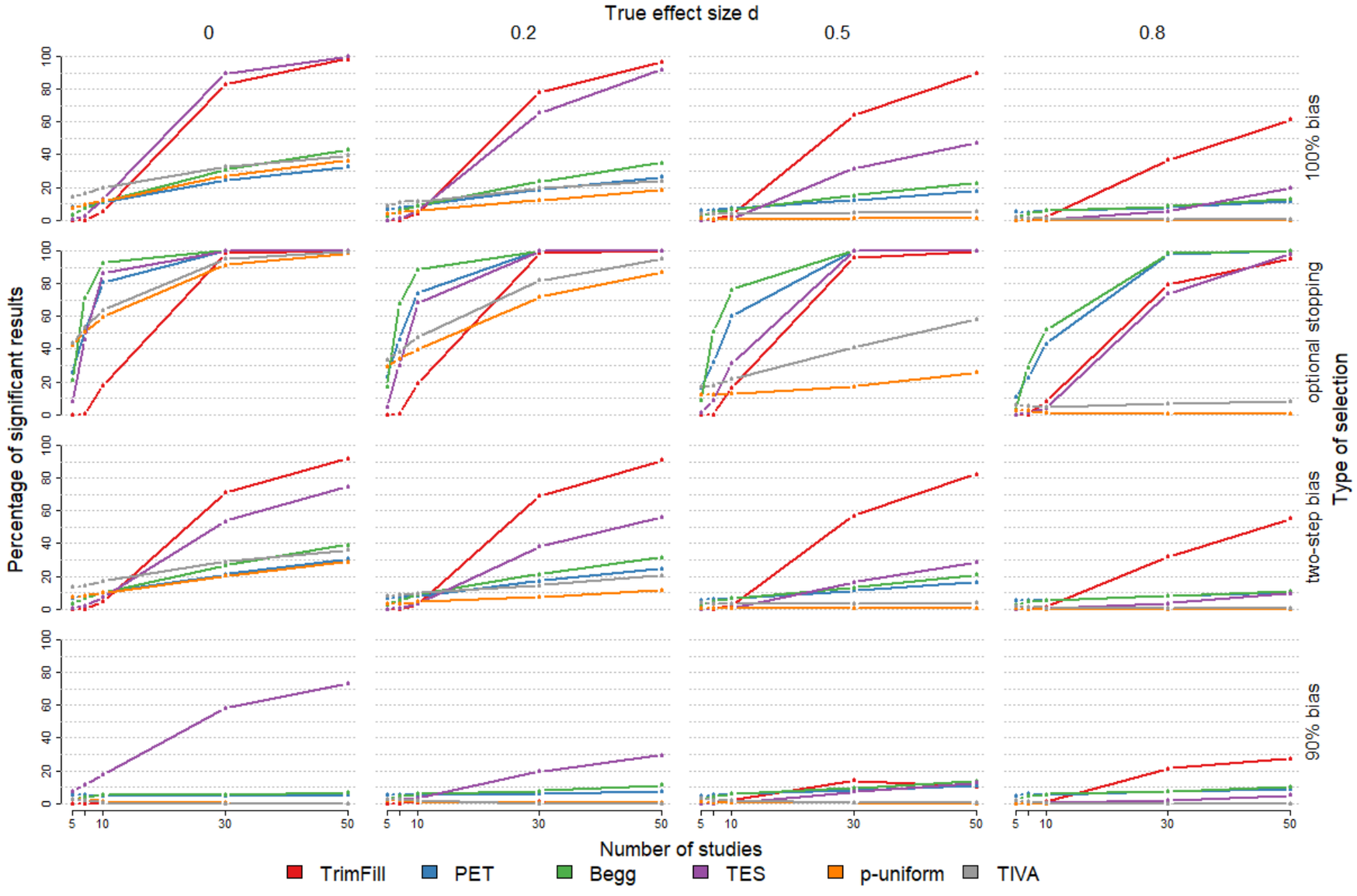

Figure S11. Power of six detection methods at **τ = 0.4** as a function of selection condition, true effect size and number of studies.

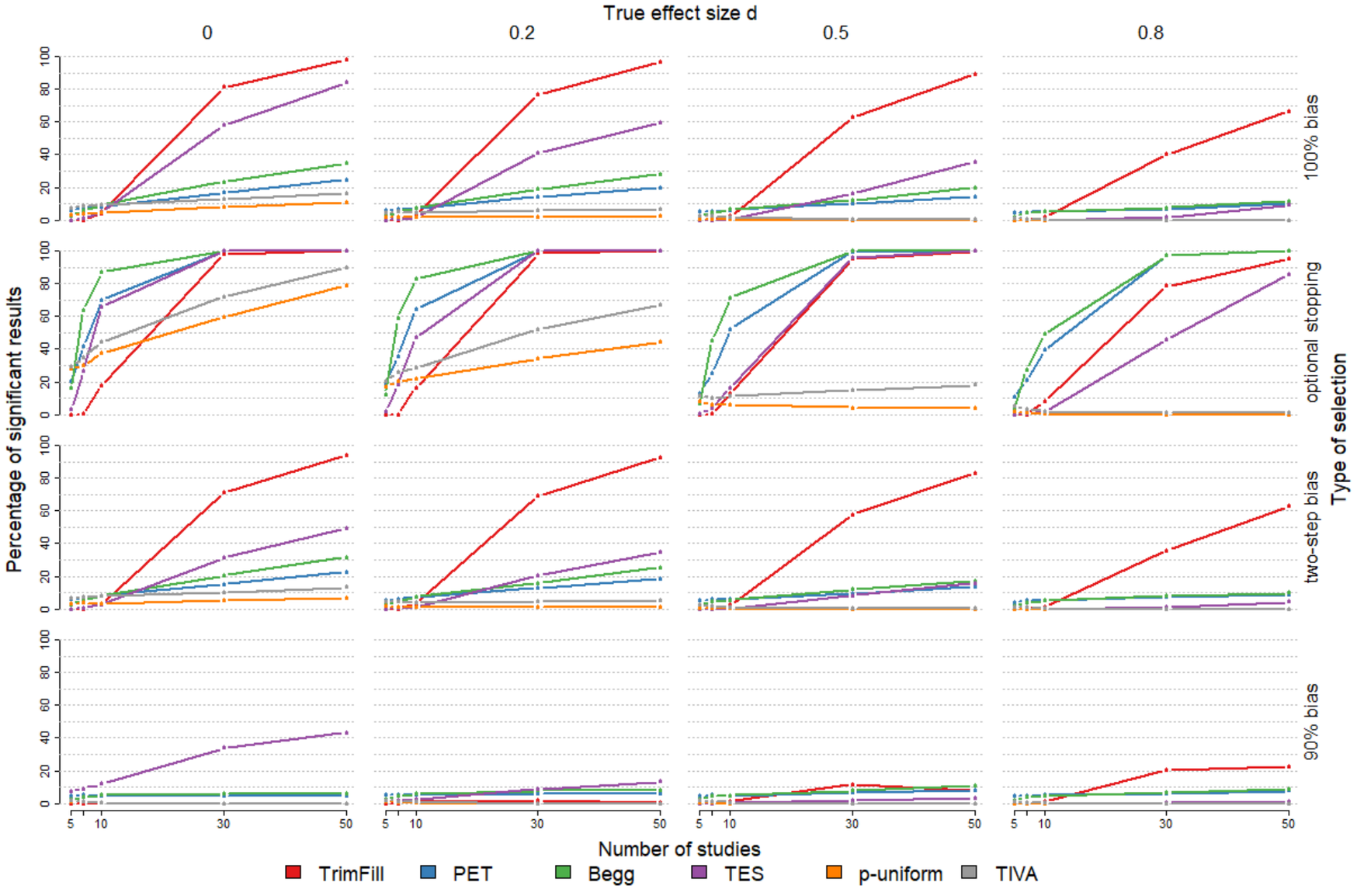

Figure S12. Power of six detection methods at **τ = 0.5** as a function of selection condition, true effect size and number of studies.

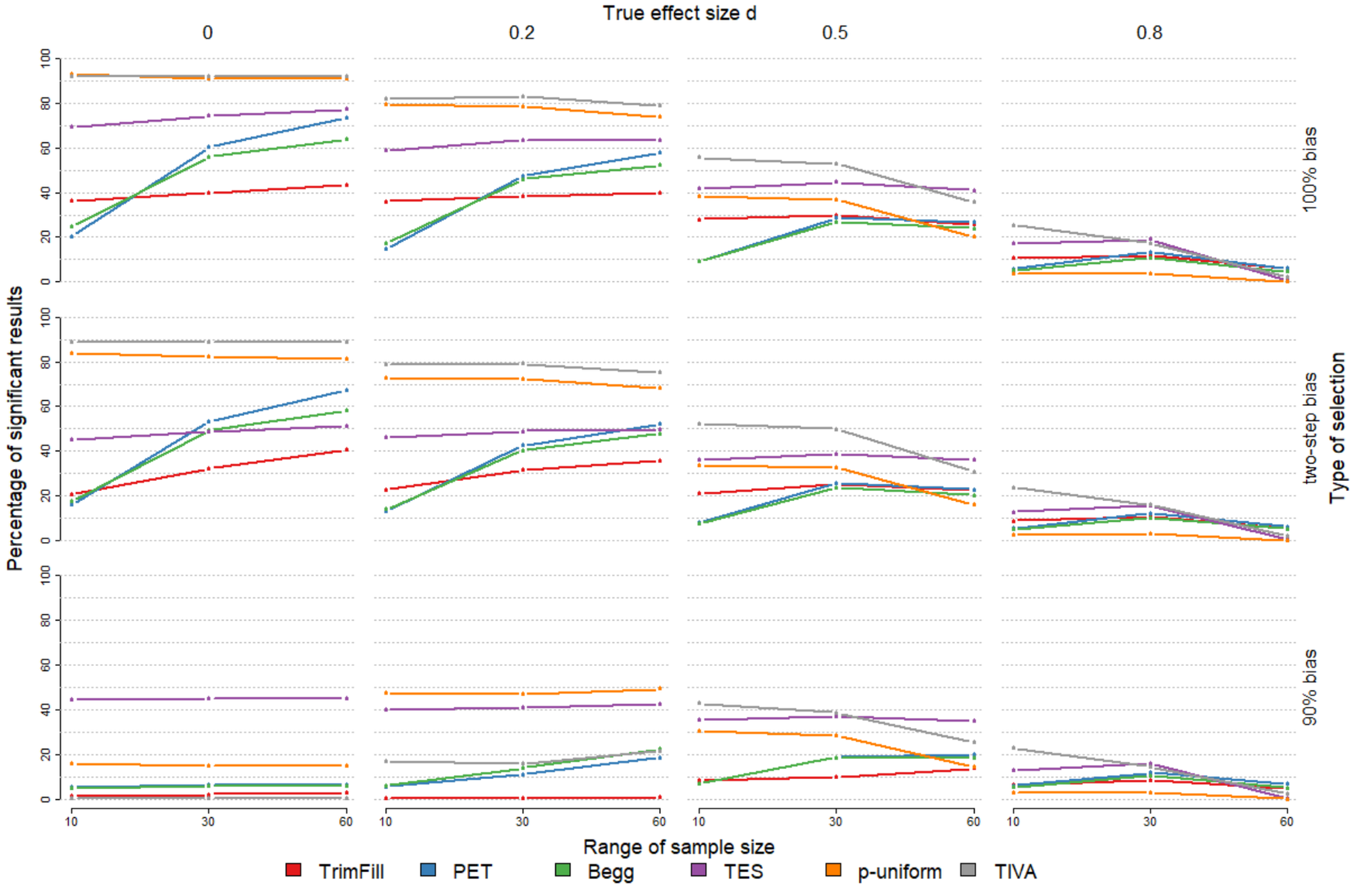

Figure S13. Power of six detection methods at $\boldsymbol{\tau = 0}$ as a function of selection condition, true effect size and range in sample size.

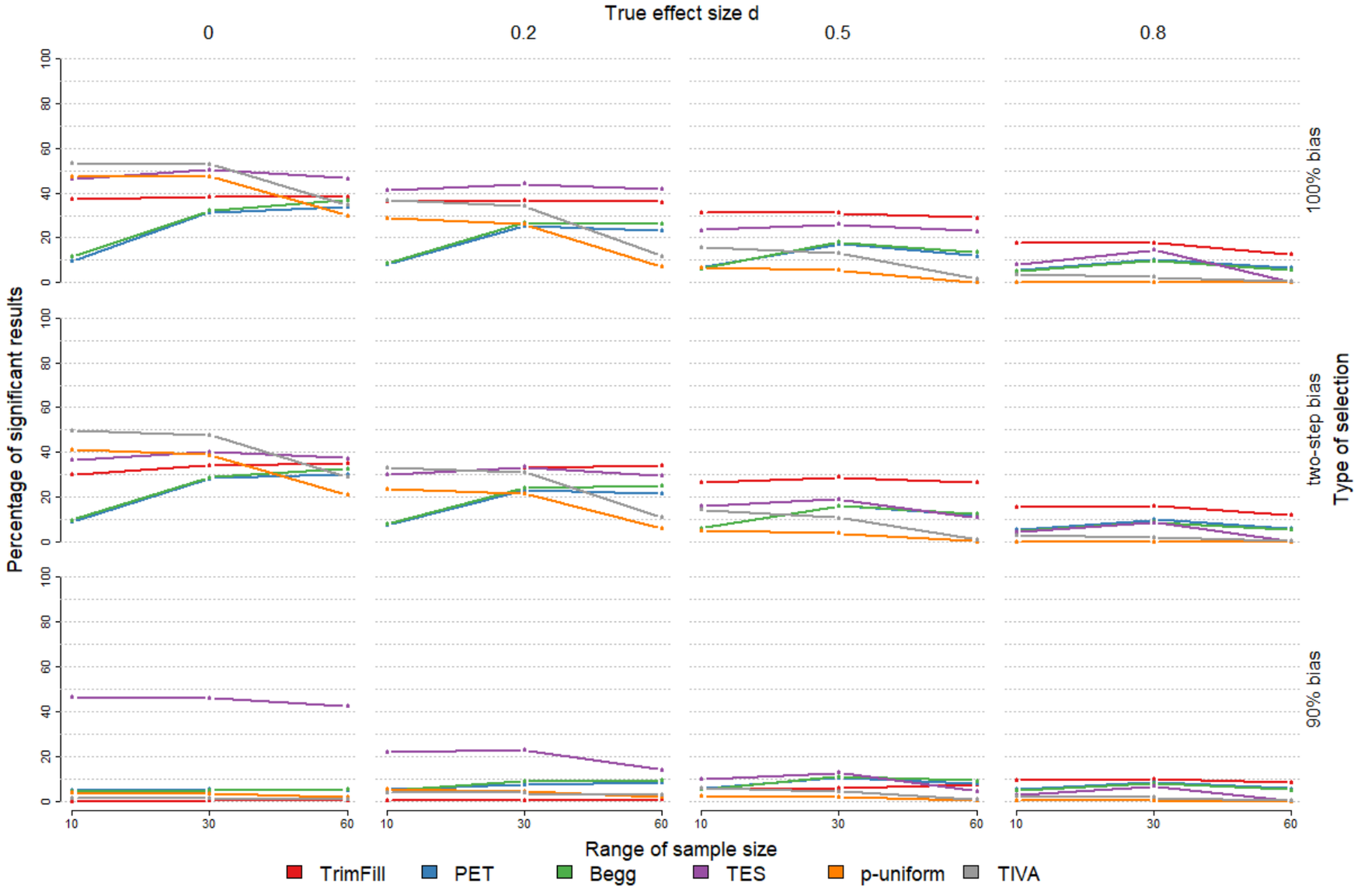

Figure S14. Power of six detection methods at **τ = 0.3** as a function of selection condition, true effect size and range in sample size.

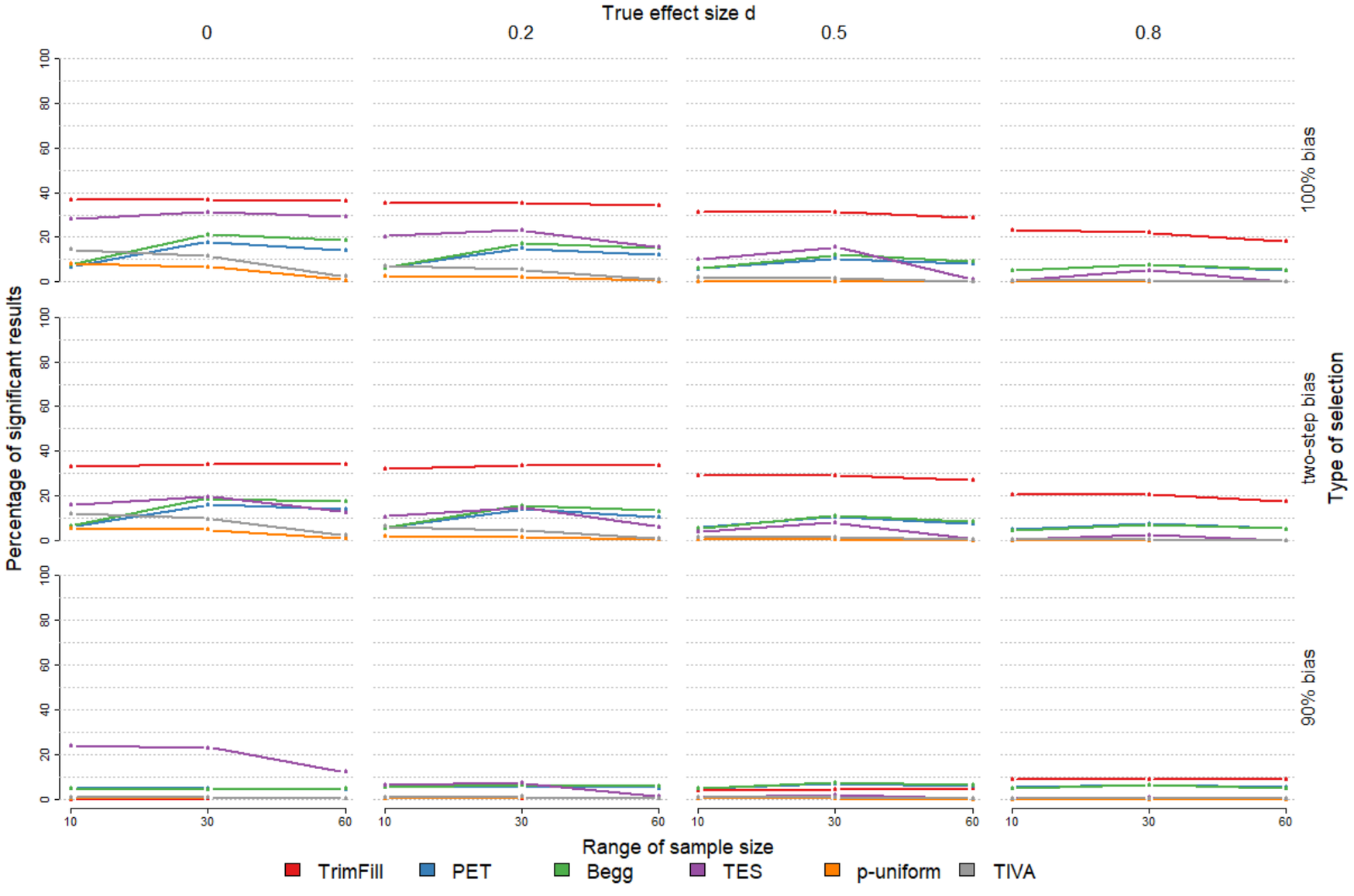

Figure S15. Power of six detection methods at **τ = 0.5** as a function of selection condition, true effect size and range in sample size.

3.5 Heterogeneity estimates

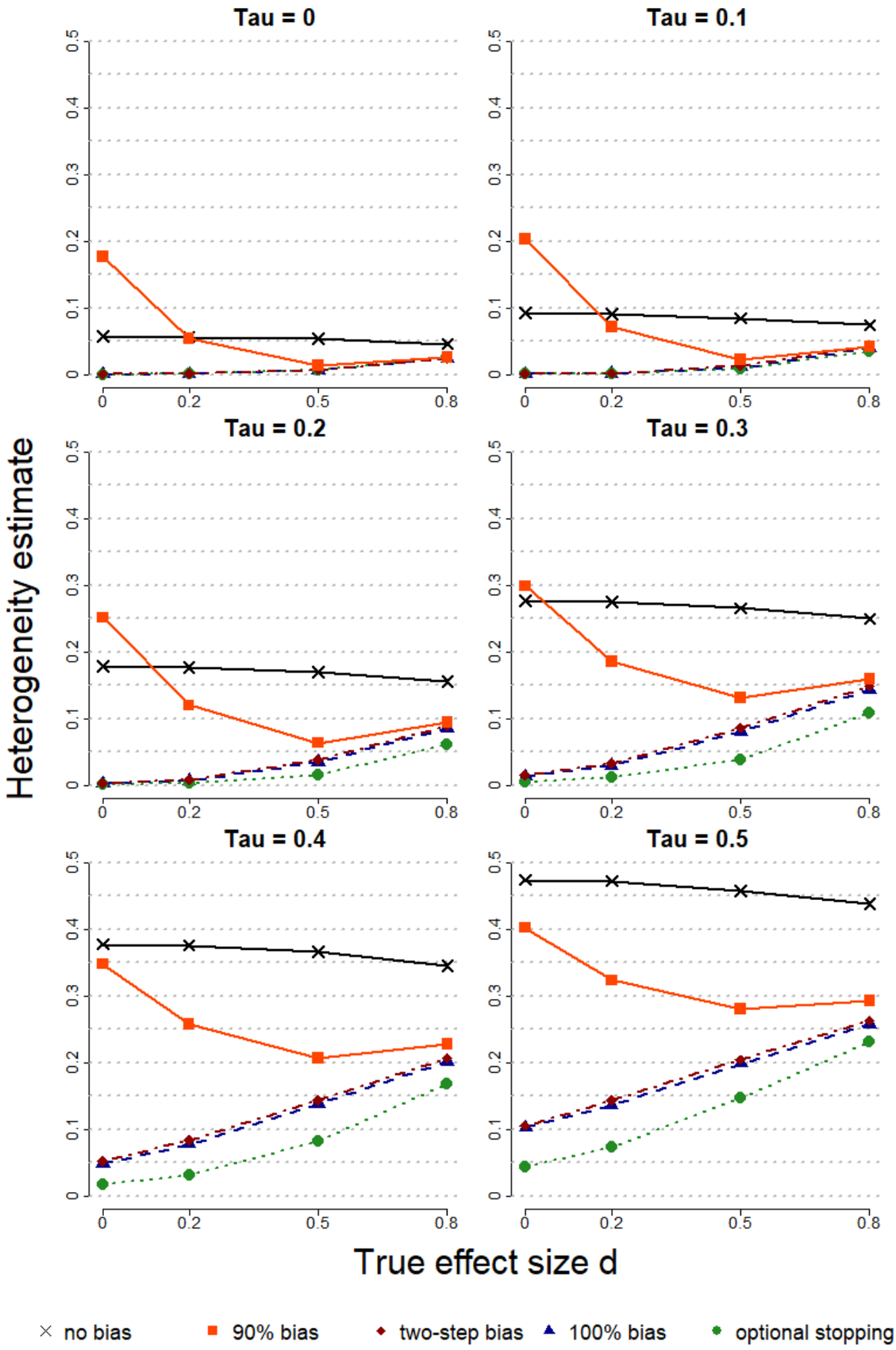

Figure S16. Heterogeneity estimates for six different levels of **τ** in four selection conditions.

4 R code for the simulations

4.1 Simulation 1 (no bias)

```
##############################################################################################
####################                    Simulation 1                    ######################
##############################################################################################

####################        Lookup Table for Parameter Settings         ######################

Para <- read.csv("~", sep = ";", strip.white = TRUE, header = TRUE, na.strings = "NA")

setwd("~")

for (Look in 1 : 600)

{

  ####################                    Parameter                    ######################

  seed  <- Para[Look, 3]
  set.seed(seed)

  Start_time <- Sys.time()

  m     <- Para[Look, 12]     # number of metas
  k     <- Para[Look, 4]      # number of studies
  n_min <- Para[Look, 6]      # min sample size
  n_max <- Para[Look, 7]      # max sample size
  m1g   <- Para[Look, 8]      # mean EG
  Tau   <- Para[Look, 17]     # heterogeneity
  sd1   <- Para[Look, 10]     # standard deviation EG
  m2    <- Para[Look, 9]      # mean CG
  sd2   <- Para[Look, 11]     # standard deviation CG

  ####################                    Simulation                   ######################

  Matrix <- matrix(nrow = (k*m), ncol = 22)
  count = 0

  for (b in 1:m)
  {
    i = 1

    while (i<=k)
    {
      a = i
      count = count+1

      n   <- round((runif(1, min = n_min, max = n_max)), digits = 0)
      m1  <- rnorm(1, m1g, Tau)
      V1  <- rnorm(n, m1, sd1)
      V2  <- rnorm(n, m2, sd2)
      MW1 <- mean(V1)
      SD1 <- sd(V1)
      MW2 <- mean(V2)
      SD2 <- sd(V2)
      N   <- n+n

      t.Wert        <- t.test(V1, V2, var.equal = TRUE) $statistic
      p.Wert_2tail  <- t.test(V1, V2, var.equal = TRUE) $p.value
      p.Wert_1tail  <- t.test(V1, V2, var.equal = TRUE, alternative = "greater")$p.value
      df            <- t.test(V1, V2, var.equal = TRUE) $parameter
      r             <- sqrt((t.Wert^2)/((t.Wert^2)+df))
```

```
      if(MW2>MW1){r <- ((-1)*r)}

      fisherz    <- (0.5*(log((1+r)/(1-r))))
      sefisherz  <- sqrt(1/(N-3))
      varfisherz <- (1/(N-3))
      d          <- (t.Wert*(sqrt((2*n)/(n*n))))
      var_d      <- ((2*n/(n*n)) + ((d^2)/(2*(n+n))))
      J          <- (1-(3/((4*(2*n-2))-1)))
      g          <- d*J
      var_g      <- ((J^2)*var_d)

      Vector_Sa  <- c(b, a, n, N, MW1, SD1, MW2, SD2, t.Wert, p.Wert_2tail, p.Wert_1tail,
                      df, r, fisherz, sefisherz, varfisherz, d, var_d, g, var_g, count, m1)

      c <- (b-1)*k+a
      Matrix[c,] <- Vector_Sa
      i <- i+1
      print(b)
    }
  }

  ####################                    Storage                    ##################

  End_time <- Sys.time()
  time     <- round((End_time - Start_time), digits = 3)

  yyy  <- as.character(Para[Look, 16])
  tt   <- m*k
  Name <- (c(yyy, (Para[Look,5]), Tau, m, k, n_min, n_max, m1g, sd1, m2, sd2, tt, seed,".rda"))

  save(Matrix, file = paste(Name, collapse = "-"))

  print(count)
  print(time)
  Look = Look+1
}

#############################################################################################
```

4.2 Simulation 2 (100% bias)

```
#########################################################################################################
####################                    Simulation 2                    ####################
#########################################################################################################

####################        Lookup Table for Parameter Settings         ####################

Para <- read.csv("~", sep = ";", strip.white = TRUE, header = TRUE, na.strings = "NA")

setwd("~")

for (Look in 601 : 1200)

{

  ####################                    Parameter                    ##################

  seed  <- Para[Look, 3]
  set.seed(seed)

  Start_time <- Sys.time()

  m     <- Para[Look, 12]     # number of metas
  k     <- Para[Look, 4]      # number of studies
  n_min <- Para[Look, 6]      # min sample size
  n_max <- Para[Look, 7]      # max sample size
  m1g   <- Para[Look, 8]      # mean EG
  Tau   <- Para[Look, 17]     # heterogeneity
  sd1   <- Para[Look, 10]     # standard deviation EG
  m2    <- Para[Look, 9]      # mean CG
  sd2   <- Para[Look, 11]     # standard deviation CG
  sigS  <- 0.05               # significance limit, selection sig. studies

  ####################                    Simulation                   ##################

  Matrix <- matrix(nrow = (k*m), ncol = 22)
  count = 0

  for (b in 1:m)
  {
    i = 1

    while (i<=k)
    {
      a = i
      count = count+1

      n   <- round((runif(1, min = n_min, max = n_max)), digits = 0)
      m1  <- rnorm(1, m1g, Tau)
      V1  <- rnorm(n, m1, sd1)
      V2  <- rnorm(n, m2, sd2)

      p.Wert_1tail  <- t.test(V1, V2, var.equal = TRUE, alternative = "greater")$p.value

      if (sigS < p.Wert_1tail) {i = i-1}
      else
        {
          MW1 <- mean(V1)
          SD1 <- sd(V1)
          MW2 <- mean(V2)
          SD2 <- sd(V2)
          N   <- n+n

          t.Wert        <- t.test(V1, V2, var.equal = TRUE) $statistic
```

```
          p.Wert_2tail  <- t.test(V1, V2, var.equal = TRUE) $p.value
          df            <- t.test(V1, V2, var.equal = TRUE) $parameter
          r             <- sqrt((t.Wert^2)/((t.Wert^2)+df))

          fisherz    <- (0.5*(log((1+r)/(1-r))))
          sefisherz  <- sqrt(1/(N-3))
          varfisherz <- (1/(N-3))
          d          <- (t.Wert*(sqrt((2*n)/(n*n))))
          var_d      <- ((2*n/(n*n)) + ((d^2)/(2*(n+n))))
          J          <- (1-(3/((4*(2*n-2))-1)))
          g          <- d*J
          var_g      <- ((J^2)*var_d)

          Vector_Sa  <- c(b, a, n, N, MW1, SD1, MW2, SD2, t.Wert, p.Wert_2tail, p.Wert_1tail,
                          df, r, fisherz, sefisherz, varfisherz, d, var_d, g, var_g, count, m1)

          c <- (b-1)*k+a
          Matrix[c,] <- Vector_Sa
          count = 0
        }

      i = i+1
      print(b)
    }
  }

  ####################                    Storage                    ###################

  End_time <- Sys.time()
  time     <- round((End_time - Start_time), digits = 3)

  yyy  <- as.character(Para[Look, 16])
  tt   <- m*k
  Name <- (c(yyy, (Para[Look,5]), Tau, m, k, n_min, n_max, m1g, sd1, m2, sd2, tt, seed,".rda"))

  save(Matrix, file = paste(Name, collapse = "-"))


  print(count)
  print(time)
  Look = Look+1
}

#############################################################################################
```

4.3 Simulation 3 (optional stopping)

```
#############################################################################################
####################                    Simulation 3                    ####################
#############################################################################################

####################        Lookup Table for Parameter Settings         ####################

Para <- read.csv("~", sep = ";", strip.white = TRUE, header = TRUE, na.strings = "NA")

setwd("~")

for (Look in 1201 : 1440)

{

  ####################                    Parameter                     ##################

  seed  <- Para[Look, 3]
  set.seed(seed)

  Start_time <- Sys.time()

  m     <- Para[Look, 12]     # number of metas
  k     <- Para[Look, 4]      # number of studies
  n     <- Para[Look, 6]      # min sample size
  nn    <- Para[Look, 20]     # increase sample size
  n_max <- Para[Look, 7]      # max sample size
  m1g   <- Para[Look, 8]      # mean EG
  Tau   <- Para[Look, 17]     # heterogeneity
  sd1   <- Para[Look, 10]     # standard deviation EG
  m2    <- Para[Look, 9]      # mean CG
  sd2   <- Para[Look, 11]     # standard deviation CG
  sigS  <- 0.05               # significance limit, selection sig. studies

  ####################                    Simulation                    ##################

  Matrix <- matrix(nrow = (k*m), ncol = 22)
  count = 0


  for (b in 1:m)
  {
    i = 1

    while (i<=k)
    {
      a = i
      count = count+1

      z = n

      m1  <- rnorm(1, m1g, Tau)
      V1  <- rnorm(n, m1, sd1)
      V2  <- rnorm(n, m2, sd2)

      p.Wert_1tail  <- t.test(V1, V2, var.equal = TRUE, alternative = "greater")$p.value

      if (sigS<=p.Wert_1tail)
      {

        while(sigS<p.Wert_1tail)
        {
          if (n<n_max)
```

```
            {
              V1<-c(V1, (rnorm(nn, m1, sd1)))
              V2<-c(V2, (rnorm(nn, m2, sd2)))
              p.Wert_1tail  <- t.test(V1, V2, var.equal = TRUE,
                                      alternative = "greater")$p.value
              n = n+nn
            }
          else
            {
              p.Wert_1tail = 0
              i = i-1
            }
        }
      }

      if (p.Wert_1tail>0)
      {
        MW1 <- mean(V1)
        SD1 <- sd(V1)
        MW2 <- mean(V2)
        SD2 <- sd(V2)
        N   <- n+n

        t.Wert        <- t.test(V1, V2, var.equal = TRUE) $statistic
        p.Wert_2tail  <- t.test(V1, V2, var.equal = TRUE) $p.value
        df            <- t.test(V1, V2, var.equal = TRUE) $parameter
        r             <- sqrt((t.Wert^2)/((t.Wert^2)+df))

        fisherz    <- (0.5*(log((1+r)/(1-r))))
        sefisherz  <- sqrt(1/(N-3))
        varfisherz <- (1/(N-3))
        d          <- (t.Wert*(sqrt((2*n)/(n*n))))
        var_d      <- ((2*n/(n*n)) + ((d^2)/(2*(n+n))))
        J          <- (1-(3/((4*(2*n-2))-1)))
        g          <- d*J
        var_g      <- ((J^2)*var_d)

        Vector_Sa  <- c(b, a, n, N, MW1, SD1, MW2, SD2, t.Wert, p.Wert_2tail, p.Wert_1tail,
                        df, r, fisherz, sefisherz, varfisherz, d, var_d, g, var_g, count, m1)

        c <- (b-1)*k+a
        Matrix[c,] <- Vector_Sa
        count = 0
      }
      i <- i+1
      n = z
    }
    print(b)
  }
  ####################                    Storage                    ##################

  End_time <- Sys.time()
  time     <- round((End_time - Start_time), digits = 3)

  yyy  <- as.character(Para[Look, 16])
  tt   <- m*k
  Name <- (c(yyy, (Para[Look,5]), Tau, m, k,nn, n, n_max, m1g, sd1, m2, sd2, tt, seed,".rda"))

  save(Matrix, file = paste(Name, collapse = "-"))

  print(count)
  print(time)
  Look = Look+1
}

#################################################################################################
```

4.4 Simulation 4 (two-step bias)

```
#############################################################################################
####################                    Simulation 4                    ####################
#############################################################################################

####################        Lookup Table for Parameter Settings         ####################

Para <- read.csv("~", sep = ";", strip.white = TRUE, header = TRUE, na.strings = "NA")

setwd("~")

for (Look in 1441 : 2040)

{

  ####################                    Parameter                    #################

  seed  <- Para[Look, 3]
  set.seed(seed)

  Start_time <- Sys.time()

  m         <- Para[Look, 12]     # number of metas
  k         <- Para[Look, 4]      # number of studies
  n_min     <- Para[Look, 6]      # min sample size
  n_max     <- Para[Look, 7]      # max sample size
  m1g       <- Para[Look, 8]      # mean EG
  Tau       <- Para[Look, 17]     # heterogeneity
  sd1       <- Para[Look, 10]     # standard deviation EG
  m2        <- Para[Look, 9]      # mean CG
  sd2       <- Para[Look, 11]     # standard deviation CG
  sigS      <- 0.05               # significance limit selection sig. studies
  AsigS     <- 1                  # selection criterion, sig. studies p.value <=
                                  # sigS (1 = 100% selection)
  nsigS     <- 0.1                # nonsignificant studies, subdivision of
                                  # the nonsignificant area
  AnsigS_1  <- 0.2                # AnsigS_1 % studies sigS<p.value<=nsigS
  AnsigS_2  <- 0                  # Ansigs_2 % studies nsigS<p.value

  ####################                    Simulation                   ##################

  Matrix <- matrix(nrow = (k*m), ncol = 22)
  count = 0

  for (b in 1:m)
  {
    i = 1

    while (i<=k)
    {

      a = i
      count = count+1

      n   <- round((runif(1, min = n_min, max = n_max)), digits = 0)
      m1  <- rnorm(1, m1g, Tau)
      V1  <- rnorm(n, m1, sd1)
      V2  <- rnorm(n, m2, sd2)
      MW1 <- mean(V1)
      SD1 <- sd(V1)
      MW2 <- mean(V2)
      SD2 <- sd(V2)
```

```
      N   <- n+n

      t.Wert        <- t.test(V1, V2, var.equal = TRUE) $statistic
      p.Wert_2tail  <- t.test(V1, V2, var.equal = TRUE) $p.value
      p.Wert_1tail  <- t.test(V1, V2, var.equal = TRUE, alternative = "greater")$p.value
      df            <- t.test(V1, V2, var.equal = TRUE) $parameter
      r             <- sqrt((t.Wert^2)/((t.Wert^2)+df))

      if(MW2>MW1){r <- ((-1)*r)}

      fisherz    <- (0.5*(log((1+r)/(1-r))))
      sefisherz  <- sqrt(1/(N-3))
      varfisherz <- (1/(N-3))
      d          <- (t.Wert*(sqrt((2*n)/(n*n))))
      var_d      <- ((2*n/(n*n)) + ((d^2)/(2*(n+n))))
      J          <- (1-(3/((4*(2*n-2))-1)))
      g          <- d*J
      var_g      <- ((J^2)*var_d)

      if (p.Wert_1tail < sigS){sel_p = AsigS} else
      {if(p.Wert_1tail >= nsigS){sel_p = AnsigS_2} else
          {sel_p = AnsigS_1}}

      Cha <- runif(1, min = 0,max = 1)

      if(sel_p>Cha) {selected = 1} else {selected = 0}

      Vector_Sa  <- c(b, a, n, N, MW1, SD1, MW2, SD2, t.Wert, p.Wert_2tail, p.Wert_1tail,
                      df, r, fisherz, sefisherz, varfisherz, d, var_d, g, var_g, count, m1)

      c <- (b-1)*k+a
      Matrix[c,] <- Vector_Sa

      if(selected == 1){i <- i+1}
    }
    print(b)
  }

  ####################                    Storage                    ##################

  End_time <- Sys.time()
  time     <- round((End_time - Start_time), digits = 3)

  yyy  <- as.character(Para[Look, 16])
  tt   <- m*k
  Name <- (c(yyy, (Para[Look,5]), Tau, m, k, n_min, n_max, m1g, sd1, m2, sd2, tt, seed,".rda"))

  save(Matrix, file = paste(Name, collapse = "-"))

  print(count)
  print(time)
  Look = Look+1
}

#############################################################################################
```

4.5 Simulation 5 (90% bias)

```
#############################################################################################
####################                    Simulation 5                    ####################
#############################################################################################

####################        Lookup Table for Parameter Settings         ####################

Para <- read.csv("~", sep = ";", strip.white = TRUE, header = TRUE, na.strings = "NA")

setwd("~")

for (Look in 2041 : 2640)

{

  ####################                    Parameter                    ##################

  seed  <- Para[Look, 3]
  set.seed(seed)

  Start_time <- Sys.time()

  m     <- Para[Look, 12]     # number of metas
  k     <- Para[Look, 4]      # number of studies
  n_min <- Para[Look, 6]      # min sample size
  n_max <- Para[Look, 7]      # max sample size
  m1g   <- Para[Look, 8]      # mean EG
  Tau   <- Para[Look, 17]     # heterogeneity
  sd1   <- Para[Look, 10]     # standard deviation EG
  m2    <- Para[Look, 9]      # mean CG
  sd2   <- Para[Look, 11]     # standard deviation CG

  sigS  <- 0.05               # significance limit, selection sig. studies
  Asig  <- 0.9

  ####################                    Simulation                   ##################

  Matrix <- matrix(nrow = (k*m), ncol = 23)
  count = 0

  for (b in 1:m)
  {
    i = 1

    while (i<=k)
    {
       Cha <- runif(1, min= 0, max = 1)

       if (Cha<Asig)
      {
        a = i

        n   <- round((runif(1, min = n_min, max = n_max)), digits = 0)
        m1  <- rnorm(1, m1g, Tau)
        V1  <- rnorm(n, m1, sd1)
        V2  <- rnorm(n, m2, sd2)

        p.Wert_1tail  <- t.test(V1, V2, var.equal = TRUE, alternative = "greater")$p.value

        if (sigS<p.Wert_1tail) {i = i-1}
        else
        {
          MW1 <- mean(V1)
          SD1 <- sd(V1)
```

```
      MW2 <- mean(V2)
      SD2 <- sd(V2)
      N   <- n+n

      p.Wert_2tail  <- t.test(V1, V2, var.equal = TRUE) $p.value
      t.Wert        <- t.test(V1, V2, var.equal = TRUE) $statistic
      df            <- t.test(V1, V2, var.equal = TRUE) $parameter
      r             <- sqrt((t.Wert^2)/((t.Wert^2)+df))


      fisherz    <- (0.5*(log((1+r)/(1-r))))
      sefisherz  <- sqrt(1/(N-3))
      varfisherz <- (1/(N-3))
      d          <- (t.Wert*(sqrt((2*n)/(n*n))))
      var_d      <- ((2*n/(n*n)) + ((d^2)/(2*(n+n))))
      J          <- (1-(3/((4*(2*n-2))-1)))
      g          <- d*J
      var_g      <- ((J^2)*var_d)

      Bias = 1

      Vector_Sa  <- c(b, a, n, N, MW1, SD1, MW2, SD2, t.Wert, p.Wert_2tail, p.Wert_1tail,
                      df, r, fisherz, sefisherz, varfisherz, d, var_d, g, var_g, m1,
                      Bias, Cha)

      c <- (b-1)*k+a
      Matrix[c,] <- Vector_Sa

    }
     i <- i+1
     print(b)
}
 else
    {
      a = i

      n   <- round((runif(1, min = n_min, max = n_max)), digits = 0)
      m1  <- rnorm(1, m1g, Tau)
      V1  <- rnorm(n, m1, sd1)
      V2  <- rnorm(n, m2, sd2)

      MW1 <- mean(V1)
      SD1 <- sd(V1)
      MW2 <- mean(V2)
      SD2 <- sd(V2)
      N   <- n+n

      p.Wert_1tail  <- t.test(V1, V2, var.equal = TRUE, alternative = "greater")$p.value
      p.Wert_2tail  <- t.test(V1, V2, var.equal = TRUE) $p.value
      t.Wert        <- t.test(V1, V2, var.equal = TRUE) $statistic
      df            <- t.test(V1, V2, var.equal = TRUE) $parameter
      r             <- sqrt((t.Wert^2)/((t.Wert^2)+df))

      if(MW2>MW1){r <- ((-1)*r)}

      fisherz    <- (0.5*(log((1+r)/(1-r))))
      sefisherz  <- sqrt(1/(N-3))
      varfisherz <- (1/(N-3))
      d          <- (t.Wert*(sqrt((2*n)/(n*n))))
      var_d      <- ((2*n/(n*n)) + ((d^2)/(2*(n+n))))
      J          <- (1-(3/((4*(2*n-2))-1)))
      g          <- d*J
      var_g      <- ((J^2)*var_d)

      Bias = 0
```

```
          Vector_Sa  <- c(b, a, n, N, MW1, SD1, MW2, SD2, t.Wert, p.Wert_2tail, p.Wert_1tail,
                          df, r, fisherz, sefisherz, varfisherz, d, var_d, g, var_g, m1,
                          Bias, Cha)

          c = (b-1)*k+a
          Matrix[c,] <- Vector_Sa

          i <- i+1
          print(b)
        }
      }
    }

    ####################                    Storage                    ##################

    End_time <- Sys.time()
    time     <- round((End_time - Start_time), digits = 3)

    yyy  <- as.character(Para[Look, 16])
    tt   <- m*k
    Name <- (c(yyy, (Para[Look,5]), Tau, m, k, n_min, n_max, m1g, sd1, m2, sd2, tt, seed,".rda"))

    save(Matrix, file = paste(Name, collapse = "-"))

    print(count)
    print(time)
    Look = Look+1
}

#############################################################################################
```

5 R code for the evaluation

```
#############################################################################################
####################                 Evaluation Script               ########################
#############################################################################################

####################              Installation of packages           ########################

'
ipak <- function(pkg){
  new.pkg <- pkg[!(pkg %in% installed.packages()[, "Package"])]
  if (length(new.pkg))
    install.packages(new.pkg, dependencies = TRUE)
  sapply(pkg, require, character.only = TRUE)
}

packages <- c("pwr", "metafor", "dplyr", "poibin", "MBESS", "puniform",
              "stringr", "puniform", "doParallel")
ipak(packages)
'
####################                 Load of Packages                ########################


packages <- c("pwr", "metafor", "dplyr", "poibin", "MBESS", "puniform","stringr","doParallel")
lapply(packages, library, character.only = TRUE)

####################                   Functions                     ########################

# Power TES

    posthoc.power <- function(n, d) pwr.t.test(n = n, d = d, type = "two.sample",
                                               alternative = "greater",
                                               sig.level = 0.05)$power

# TES Function

    TES_Fisher <- function(Pow_value, Obs, k)
    {
      if(Obs<(k/2))
      {
        iiii = 0
        P_TES_fi = 0
        while (iiii<Obs)
        {
          r <- expand.grid(rep(list(0:1), k))
          Mat_komb <- r[rowSums(r) == iiii, ]
          tt <- choose(k, iiii)
          j = 1
          while (j<=tt)
          {
            jj = 1
            Produkt = 1
            while (jj<=k)
            {
              if (Mat_komb[j, jj] == 1)
              {
                Mat_komb[j, jj] = Pow_value[jj]
              }
              else
              {
                Mat_komb[j, jj] = (1-Pow_value[jj])
              }
              Produkt = Produkt*Mat_komb[j, jj]
              jj = jj+1
            }
```

```
            Mat_komb[j, (k+1)] = Produkt
            j = j+1
          }
          summe <- sum(Mat_komb[, (k+1)])
          P_TES_fi <- P_TES_fi+summe
          iiii = iiii+1
        }
        P_TES_fi <- 1-P_TES_fi
        return(P_TES_fi)
      }
      else
      {
        iiii = k
       P_TES_fi = 0
       while (Obs<=iiii)
       {
         r <- expand.grid(rep(list(0:1), k))
         Mat_komb <- r[rowSums(r) == iiii, ]
         tt <- choose(k, iiii)
         j = 1
         while (j<=tt)
         {
           jj = 1
           Produkt = 1
           while (jj<=k)
           {
             if (Mat_komb[j, jj] == 1)
             {
               Mat_komb[j, jj] <- Pow_value[jj]
             }
             else
             {
               Mat_komb[j, jj] <- (1-Pow_value[jj])
             }
              Produkt <- Produkt*Mat_komb[j, jj]
              jj = jj+1
             }
              Mat_komb[j, (k+1)] = Produkt
            j = j+1
          }
         summe <- sum(Mat_komb[, (k+1)])
         P_TES_fi <- P_TES_fi+summe
         iiii = iiii-1
      }
      return(P_TES_fi)
      }
    }

d_trans <- function (x)
    {
      h <- x/(2*sqrt(2))
      y <- log(h + sqrt(h ^ 2 + 1))
      d <- sqrt(2)* y
      return(d)
     }
```

```
####################              Set Working Directory              ####################

SWD          <- setwd("~")
Para_WD      <- as.character("/Parameter.csv")
Name_Para_WD <- (c(SWD, Para_WD))
Name_Para_WD <- paste(Name_Para_WD, collapse = "")

####################        Lookup Table for Parameter Settings         ####################

Para <- read.csv(Name_Para_WD, sep = ";", strip.white = TRUE, header = TRUE, na.strings = "NA")

options(digits = 4)

cl <- makeCluster(3)
registerDoParallel(cl)


foreach (Look = 1 : 2640, .export = c('rma', 'ppoibin', 'puniform', 'pwr.t.test'),
                          .packages = c('metafor', 'poibin', 'puniform', 'pwr')) %dopar%  {

  Nams <- c(SWD, (as.character(Para[Look, 15])))
  Nams <- paste(Nams, collapse = "")
  load(Nams)

#############################################################################################
####################              Start of the Evaluation              ####################
#############################################################################################

  Start_time <- Sys.time()

  seed  <- Para[Look, 3]
  m     <- Para[Look, 12]
  k     <- Para[Look, 4]
  Tau   <- Para[Look, 17]
  MSP   <- Para[Look, 18]
  SDSP  <- Para[Look, 19]
  d     <- Para[Look, 5]
  n_min <- Para[Look, 6]
  n_max <- Para[Look, 7]
  m1    <- Para[Look, 8]
  sd1   <- Para[Look, 10]
  m2    <- Para[Look, 9]
  sd2   <- Para[Look, 11]

  r     <- round((d/(sqrt((d^2)+4))), digits = 2)

  Matrix_Par <- matrix(nrow = 1000, ncol = 46)

#############################################################################################

  vv <- 1
  while(vv<=m)
  {
    aa <- (((vv-1)*k)+1)
    bb <- (((vv-1)*k)+k)

    Num_sig <- (length(subset((Matrix[(aa:bb), 11]), (Matrix[(aa:bb), 11])<0.05)))

    g_trans     <- d_trans(Matrix[(aa:bb), 19])
    Var_g_trans <- 1/(2*(Matrix[(aa:bb), 3]))
    se_g_trans  <- sqrt(1/(2*(Matrix[(aa:bb), 3])))
```

```
#################################################################################################
#### (0) Metaanalysis (W. Viechtbauer) ##########################################################
#################################################################################################

    Meta_SIM <- rma(yi = (Matrix[(aa:bb), 14]), sei = (Matrix[(aa:bb), 15]),
                    ni = (Matrix[(aa:bb), 4]),  data = "Matrix", method = "DL")

    Meta_CON <- confint(Meta_SIM, digits = 4)

    Meta_k     <- Meta_SIM$k
    Meta_est   <- Meta_SIM$b
    Meta_es    <- Meta_SIM$se
    Meta_p     <- Meta_SIM$pval

    # Inverse Transformation
    r_Rueck <- (((exp(2*Meta_est))-1)/((exp(2*Meta_est))+1))
    d_Rueck <- ((2*r_Rueck)/(sqrt(1-(r_Rueck*r_Rueck))))

    Meta_tau        <- Meta_CON$random[2]


#################################################################################################
#### (1) Trim & Fill (Duval & Tweedie, 2000) ####################################################
#################################################################################################


### L0

  TR_LO_k      <- (trimfill(Meta_SIM, side = "left", estimator = "L0")$k)

### R0

  TR_RO_k      <- (trimfill(Meta_SIM, side = "left", estimator = "R0")$k)

#################################################################################################
#### (2) p Uniform (van Assen & van Aert & Wicherts, 2014 & 2016 & 2017) ########################
#################################################################################################

  if ((length((subset((Matrix[(aa:bb), 11]), (Matrix[(aa:bb), 11])<0.05))))>0)
    {
      LNP           <- puniform(n1i = (Matrix[(aa:bb), 3]), n2i = (Matrix[(aa:bb), 3]),
                                tobs = (Matrix[(aa:bb), 9]), alpha = 0.1, side = "right",
                                method = "LNP")

      LNP_PB_Test   <- LNP$L.pb
      LNP_PB_Test_p <- LNP$pval.pb

      LN1MINP       <- puniform(n1i = (Matrix[(aa:bb), 3]), n2i = (Matrix[(aa:bb), 3]),
                                tobs = (Matrix[(aa:bb), 9]), alpha = 0.1, side = "right",
                                method = "LN1MINP")

      LN1MINP_PB_Test   <- LN1MINP$L.pb
      LN1MINP_PB_Test_p <- LN1MINP$pval.pb

      P             <- puniform(n1i = (Matrix[(aa:bb), 3]), n2i = (Matrix[(aa:bb), 3]),
                                tobs = (Matrix[(aa:bb), 9]), alpha = 0.1, side = "right",
                                method = "P")

      P_PB_Test   <- P$L.pb
      P_PB_Test_p <- P$pval.pb


    }else
```

```
    {
      LNP_PB_Test   <- NA
      LNP_PB_Test_p <- NA

      LN1MINP_PB_Test   <- NA
      LN1MINP_PB_Test_p <- NA

     P_PB_Test   <- NA
     P_PB_Test_p <- NA

  }


#################################################################################################
#### (3) TES (Francis, 2013) ####################################################################
#################################################################################################

  g.pooled         <- (rma(yi = (Matrix[(aa:bb), 19]), vi = (Matrix[(aa:bb), 20]),
                           method = "DL")$b)
  phpower_g.pooled <- mapply(posthoc.power, n = (Matrix[(aa:bb), 3]), d = g.pooled)

  phpower_g.Ind    <- mapply(posthoc.power, n = (Matrix[(aa:bb), 3]),
                             d = (Matrix[(aa:bb), 19]))

#################################################################################################

  # Chi_Test TES k>15 ###

  Obs_Sig <- length(subset((Matrix[(aa:bb), 3]), ((Matrix[(aa:bb), 11]) <= 0.05)))

  A_Test  <-  function (O, E, M)
              {(((O-E)^2)/E)+(((O-E)^2)/(M-E)))}

  if (k>15)
    {
      # Chi + Post hoc Power, g pooled k>15
      Expe_g.pooled           <- sum(phpower_g.pooled)
      A_g.pooled              <- A_Test(O = Obs_Sig, E = Expe_g.pooled, M = k)
      p_TES_Fish_Chi_g.pooled <- (1-(pchisq(A_g.pooled, 1)))

      # Chi + Post hoc Power, hedges g k>15
      Expe_g           <- sum(phpower_g.Ind)
      A_g              <- A_Test(O = Obs_Sig, E = Expe_g, M = k)
      p_TES_Fish_Chi_g <- (1-(pchisq(A_g, 1)))
    } else
      {
        Expe_g.pooled <- NA
        Expe_g        <- NA

  # Fisher-Exact-Test, bei k<15

        #Fisher-Exact-Test, Post hoc Power, g pooled k<15 ####
        p_TES_Fish_Chi_g.pooled <- TES_Fisher(Pow_value = phpower_g.pooled,
                                              Obs = Obs_Sig, k = k)

        #Fisher-Exact-Test, Post hoc Power, hedges g k<15
        p_TES_Fish_Chi_g        <- TES_Fisher(Pow_value = phpower_g.Ind,
                                              Obs = Obs_Sig, k = k)
    }
```

```
#############################################################################################
#### (4) TIVA, Test of Insufficient Variance (Schimmack, 2014) ##############################
#############################################################################################

    Calc_z.Val       <- function(p.Val){qnorm(1-(p.Val))}

    z.Values         <- mapply(Calc_z.Val, p.Val = (Matrix[(aa:bb), 11]))
    Obs.Var_z.Values <- var(z.Values)
    Chi.square       <- Obs.Var_z.Values*(k-1)

    # TIVA-Wert
    p.TIVA           <- pchisq(Chi.square, df = (k-1), lower.tail = TRUE)

#############################################################################################
#### (5) PET_PEESE (Stanley & Doucouliagos, 2012 & 2014) ####################################
#############################################################################################


    SE_d  <- sqrt(Matrix[(aa:bb), 18])
    PET_d <- lm((Matrix[(aa:bb), 17])~(SE_d), weights = (1/(Matrix[(aa:bb), 18])))

    PET_Slo_d    <- summary(PET_d)$coefficient[2:2]
    PET_Slo_p_d  <- summary(PET_d)$coefficient[8:8]


    PEESE_d <- lm((Matrix[(aa:bb), 17])~(Matrix[(aa:bb), 18]),
                  weights = (1/(Matrix[(aa:bb), 18])))

    PEESE_Slo_d    <- summary(PEESE_d)$coefficient[2:2]
    PEESE_Slo_p_d  <- summary(PEESE_d)$coefficient[8:8]


    PET_r <- lm((Matrix[(aa:bb), 14])~(Matrix[(aa:bb), 15]),
                weights = (1/(Matrix[(aa:bb), 16])))

    PET_Slo_r    <- summary(PET_r)$coefficient[2:2]
    PET_Slo_p_r  <- summary(PET_r)$coefficient[8:8]


    PEESE_r <- lm((Matrix[(aa:bb), 14])~(Matrix[(aa:bb), 16]),
                  weights = (1/(Matrix[(aa:bb), 16])))

    PEESE_Slo_r    <- summary(PEESE_r)$coefficient[2:2]
    PEESE_Slo_p_r  <- summary(PEESE_r)$coefficient[8:8]

    PET_h <- lm((g_trans)~(se_g_trans),
                weights = (1/(Var_g_trans)))

    PET_Slo_h    <- summary(PET_h)$coefficient[2:2]
    PET_Slo_p_h  <- summary(PET_h)$coefficient[8:8]


    PEESE_h <- lm((g_trans)~(Var_g_trans),
                weights = (1/(Var_g_trans)))

    PEESE_Slo_h    <- summary(PEESE_h)$coefficient[2:2]
    PEESE_Slo_p_h  <- summary(PEESE_h)$coefficient[8:8]
```

```
#############################################################################################
#### (6) Rank-Correlation-Test (Begg & Mazumdar, 1994) ######################################
#############################################################################################

    Rankkor_d     <- ranktest((Matrix[(aa:bb), 17]), (Matrix[(aa:bb), 18]))
    tau.Rankkor_d <- as.numeric(Rankkor_d$tau)
    p.Rankkor_d   <- as.numeric(Rankkor_d$pval)

    Rankkor_r     <- ranktest((Matrix[(aa:bb), 14]), sei = (Matrix[(aa:bb), 15]))
    tau.Rankkor_r <- as.numeric(Rankkor_r$tau)
    p.Rankkor_r   <- as.numeric(Rankkor_r$pval)

    Rankkor_h     <- ranktest((g_trans), sei = (Var_g_trans))
    tau.Rankkor_h <- as.numeric(Rankkor_h$tau)
    p.Rankkor_h   <- as.numeric(Rankkor_h$pval)

#############################################################################################
####################               End of the Evaluation               ####################
#############################################################################################

    Vektor_Ausg_Par <- round((c(

      (d), (r), (Tau), (m), (k), (MSP), (SDSP),
      (Num_sig), (d_Rueck), (r_Rueck),
      (Meta_k), (Meta_est), (Meta_es), (Meta_p), (Meta_tau),
      (TR_LO_k), (TR_RO_k),
      (LNP_PB_Test), (LNP_PB_Test_p), (LN1MINP_PB_Test), (LN1MINP_PB_Test_p),
      (P_PB_Test), (P_PB_Test_p),
      (p_TES_Fish_Chi_g.pooled), (p_TES_Fish_Chi_g), (Expe_g.pooled), (Expe_g),
      (p.TIVA)
      (PET_Slo_d), (PET_Slo_p_d),
      (PEESE_Slo_d), (PEESE_Slo_p_d),
      (PET_Slo_r), (PET_Slo_p_r),
      (PEESE_Slo_r), (PEESE_Slo_p_r),
      (PET_Slo_h), (PET_Slo_p_h),
      (PEESE_Slo_h), (PEESE_Slo_p_h),
      (tau.Rankkor_d), (p.Rankkor_d),
      (tau.Rankkor_r), (p.Rankkor_r),
      (tau.Rankkor_h), (p.Rankkor_h),)), digits = 4)

  Matrix_Par[(vv), ] <- Vektor_Ausg_Par

  print(vv)
  vv <- (vv+1)
  }

#############################################################################################

  colnames(Matrix_Par) <- c(

    ("d"), ("r"), ("Tau"), ("m"), ("k"), ("MSP"), ("SDSP"),
    ("Num_sig"), ("d_Rueck"), ("r_Rueck"),
    ("Meta_k"), ("Meta_est"), ("Meta_es"), ("Meta_p"), ("Meta_tau"),
    ("TR_LO_k"), ("TR_RO_k"),
    ("LNP_PB_Test"), ("LNP_PB_Test_p"), ("LN1MINP_PB_Test"), ("LN1MINP_PB_Test_p"),
    ("P_PB_Test"), ("P_PB_Test_p"),
    ("p_TES_Fish_Chi_g.pooled"), ("p_TES_Fish_Chi_g"), ("Expe_g.pooled"), ("Expe_g"),
    ("p.TIVA"),
    ("PET_Slo_d"), ("PET_Slo_p_d"),
    ("PEESE_Slo_d"), ("PEESE_Slo_p_d"),
    ("PET_Slo_r"), ("PET_Slo_p_r"),
    ("PEESE_Slo_r"), ("PEESE_Slo_p_r"),
    ("PET_Slo_h"), ("PET_Slo_p_h"),
    ("PEESE_Slo_h"), ("PEESE_Slo_p_h"),
```

```
    ("tau.Rankkor_d"), ("p.Rankkor_d"),
    ("tau.Rankkor_r"), ("p.Rankkor_r"),
    ("tau.Rankkor_h"), ("p.Rankkor_h"))

####################                    Storage                    ##################

  End_time <- Sys.time()
  time     <- round((End_time - Start_time), digits = 3)

  yyy  <- as.character(Para[Look, 2])
  Name <- (c("M_Par",yyy,".rda"))
  save(Matrix_Par, file = paste(Name, collapse = "-"))

}

stopCluster(cl)

#############################################################################################
```

6 MIT license

MIT License

References

Begg, C., & Mazumdar, M. (1994). Operating characteristics of a rank correlation test for publication. *Biometrics, 50*(4), 1088-1101. https://doi.org/10.2307/2533446

Borenstein, M., Hedges, L. V., Higgins, J. T., & Rothstein, H. R. (2009). Introduction to meta-analysis. Chichester, UK: Wiley.

Duval, S. (2005). The Trim and Fill Method. In H. R. Rothstein, A. J. Sutton, & M. Borenstein, Publication bias in meta analysis: Prevention, assessment and adjustments (pp. 127-144). Chichester: Wiley.

Duval, S., & Tweedie, R. (2000a). A Nonparametric "Trim and Fill" Method of Accounting for Publication Bias in Meta-Analysis. *Journal of the American Statistical Association, 95*(449), 89-98. https://doi.org/10.2307/2669529

Duval, S., & Tweedie, R. (2000b). Trim and Fill: A Simple Funnel-Plot-Based Method of Testing and Adjusting for Publication Bias in Meta-Analysis. *Biometrics, 56*(2), 455-463. https://doi.org/10.1111/j.0006-341X.2000.00455.x

Egger, M., Smith, G. D., Schneider, M., & Minder, C. (1997). Bias in meta-analysis detected by a simple, graphical test. *British medical journal, 315*, 629–634. https://doi.org/10.1136/bmj.315.7109.629

Francis, G. (2013). Replication, statistical consistency, and publication bias. *Journal of Mathematical Psychology, 57*(5), 153-169. https://doi.org/10.1016/j.jmp.2013.02.003

Hedges, L. V., Olkin, I. (1985). *Statistical methods for meta-analysis.* San Diego, CA: Academic Press.

Ioannidis, J., & Trikalinos, T. (2007). An exploratory test for an excess of significant findings. *Clinical Trials, 4*(3), 245-253. https://doi.org/10.1177/1740774507079441

Kromrey, J., & Rendina-Gobioff, G. (2006). On knowing what we do not know. An Empirical Comparison of Methods to Detect Publication Bias in Meta-Analysis. *Educational and Psychological Measurement, 66*(3), 357-373. https://doi.org/10.1177/0013164405278585

Pustejovsky, J. E., Rodgers, M., A. (2018). Testing for funnel plot asymmetry of standardized mean differences. Research Synthesis Methods, 10(1), 57-71. https://doi.org/10.1002/jrsm.1332

Schimmack, U. (2018, November 29). The Test of Insufficient Variance (TIVA): A New Tool for the Detection of Questionable Research Practices [Blog Post]. Retrieved from https://replicationindex.wordpress.com/2014/12/30/the-test-of-insufficient-variance-tiva-a-new-tool-for-the-detection-of-questionable-research-practices/

Schmidt, F., & Hunter, J. (2014). Methods of Meta-Analysis: Correcting Error and Bias in Research Findings. London: Sage.

Stanley, T. D., & Doucouliagos, H. (2014). Meta-Regression approximations to reduce publication selection bias. *Research Synthesis Methods, 5*(1), 60-78. https://doi.org/10.1002/jrsm.1095

Sterne, J., & Egger, M. (2005). Regression methods to detect publication and other bias in meta-analysis. In H. Rothstein, A. Sutton, & M. Borenstein, Publication bias in meta-analysis: Prevention, assessment and adjustments (pp. 99–110). Chichester, UK: Wiley.

Sterne, J., Gavaghan, D., & Egger, M. (2000). Publication and related bias in meta-analysis: power of statistical tests and prevalence in the literature. *Journal of Clinical Epidemiology*, *53*(11), 1119–1129. https://doi.org/10.1016/S0895-4356(00)00242-0

van Aert, R. (2017). puniform. R package version 0.0.3. Retrieved from https://CRAN.R-project.org/package=puniform

van Aert, R. C. (2018, November 29). Web application p-uniform. Retrieved from https://rvanaert.shinyapps.io/p-uniform/

van Aert, R. C. M., Wicherts, J. M., & van Assen, M. A. L. M. (2016). Conducting meta-analyses based on p values: Reservations and recommendations for applying p-uniform and p-curve. *Perspectives on Psychological Science*, 11(5), 713-729. https://doi.org/10.1177/1745691616650874

van Assen, M., van Aert, R., & Wicherts, J. (2015). Meta-analysis using effect size distributions of only statistically significant studies. *Psychological Methods*, *20*(3), 293-309. https://doi.org/10.1037/met0000025

Viechtbauer, W. (2010). Conducting meta-analyses in R with the metafor package. *Journal of Statistical Software*, *36*(3), 1-18. https://doi.org/10.18637/jss.v036.i03