Supplemenentary code to reproduce the numerical results in Di Caterina and Kosmidis (2017)

Ioannis Kosmidis, Claudia Di Caterina

01 Nov 2017

Workspace preparation

This page provides R (R Core Team 2017) code to reproduce the results in the manuscript 'Location-adjusted Wald statistic for scalar parameters' (Di Caterina and Kosmidis 2017).

This script assumes that the working directory is set to `supplementary_1710-11217`. If you have placed the contents of `supplementary_1710-11217` elsewhere, then change `path` appropriately.

```
path <- "."
code_path <- paste(path, "code", sep = "/")
results_path <- paste(path, "results", sep = "/")
```

The script `corzed.R` provides the `corzed` method for computing location-adjusted Wald statistics and other associated procedures (e.g. confidence intervals) for generalised linear models and beta regression models.

```
source(paste(code_path, "corzed.R", sep = "/"))
```

The following code chunk loads the required packages

```
library("betareg")
library("enrichwith")
library("plyr")
library("dplyr")
library("doMC")
library("lmtest")
library("survival")
library("brglm2")
library("cond")
library("ggplot2")
library("gridExtra")
```

Pre-saved R image files

Some of the code-chunks below load objects from the pre-saved R image files in the results directory. These image files are the outputs of the script `babies_simulation.R`, `brockwell_gordon_simulation.R`, `clotting_simulation.R`, `dyslexia_simulation.R`.

Table 1

```
data("ReadingSkills", package = "betareg")

## maximum likelihood estimates and corresponding 95\% Wald confidence intervals
rs_beta_ml <- betareg(accuracy ~ dyslexia * iq | dyslexia + iq,
```

```
                      data = ReadingSkills, type = "ML")
rs_summary_ml <- coef(summary(rs_beta_ml))
rs_ml_estimates <- do.call("rbind", lapply(rs_summary_ml, function(z) z[, c("Estimate", "Std. Error")]))
rs_ml_cis <- confint(rs_beta_ml)

## bias corrected fit and corresponding 95\% Wald confidence intervals
rs_beta_bc <- update(rs_beta_ml, type = "BC")
rs_summary_bc <- coef(summary(rs_beta_bc))
rs_bc_estimates <- do.call("rbind", lapply(rs_summary_bc, function(z) z[, c("Estimate", "Std. Error")]))
rs_bc_cis <- confint(rs_beta_bc)

## Table 1 in manuscript
table1 <- cbind(rs_ml_estimates, rs_bc_estimates, rs_ml_cis, rs_bc_cis)
round(table1, 3)
#             Estimate Std. Error Estimate Std. Error  2.5 % 97.5 %  2.5 % 97.5 %
# (Intercept)    1.123      0.143    1.104      0.148  0.843  1.403  0.815  1.394
# dyslexia      -0.742      0.143   -0.723      0.148 -1.021 -0.462 -1.013 -0.434
# iq             0.486      0.133    0.477      0.139  0.225  0.747  0.205  0.750
# dyslexia:iq   -0.581      0.133   -0.572      0.138 -0.841 -0.321 -0.843 -0.301
# (Intercept)    3.304      0.223    3.127      0.224  2.868  3.741  2.688  3.566
# dyslexia       1.747      0.262    1.703      0.262  1.232  2.261  1.189  2.217
# iq             1.229      0.267    1.170      0.268  0.705  1.753  0.644  1.696
```

Table 2

`dyslexia_simulation.rda` below is the output of `dyslexia_simulation.R` in ./code, which replicates the simulation study described in Example 1.1 of Di Caterina and Kosmidis (2017)

```
load(paste(results_path, "dyslexia_simulation.rda", sep = "/"))

## typeI error
typeI <- ddply(res, ~ statistic + parameter, function(x) {
    levels <- c(0.1, 1, 2.5, 5, 10)/100
    p_value_2sided <- 2 * pnorm(-abs(x$value))
    p_value_left <- pnorm(x$value)
    p_value_right <- 1 - pnorm(x$value)
    rate_2sided <- sapply(levels, function(alpha) mean(p_value_2sided < alpha))
    rate_left <- sapply(levels, function(alpha) mean(p_value_left < alpha))
    rate_right <- sapply(levels, function(alpha) mean(p_value_right < alpha))
    out <- data.frame(
        test = rep(c("2sided", "left", "right"), each = length(levels)),
        typeI = c(rate_2sided, rate_left, rate_right),
        level = rep(levels, times = 3))
    out
})

## compute coverage probabilities
rs_coverage <- typeI %>%
    filter((statistic %in% c("mle", "cor", "bc")) &
           level %in% c(0.1, 0.05, 0.01) &
           test == "2sided" &
           parameter %in% c(2, 3, 4, 6, 7)) %>%
```

```
    select(-test) %>%
    mutate(coverage = round(100 * (1 - typeI), 1)) %>%
    mutate(level = 100 * (1 - level)) %>%
    select(-typeI) %>%
    arrange(level) %>%
    reshape(idvar = c("level", "parameter"), v.names = "coverage",
            timevar = "statistic",
            direction = "wide")

## Table 2 in the manuscript
table2 <- rs_coverage %>% select(-coverage.cor)
table2
#    parameter level coverage.bc coverage.mle
# 1          2    90        88.2         86.9
# 2          3    90        86.9         84.8
# 3          4    90        87.0         85.0
# 4          6    90        83.4         82.4
# 5          7    90        81.7         79.1
# 16         2    95        93.4         92.4
# 17         3    95        92.7         91.0
# 18         4    95        92.7         91.2
# 19         6    95        89.9         89.1
# 20         7    95        88.5         86.0
# 31         2    99        98.2         97.7
# 32         3    99        97.9         97.1
# 33         4    99        97.9         97.2
# 34         6    99        96.5         96.1
# 35         7    99        95.7         94.4
```

Table 3

```
## The clotting data set
clotting <- data.frame(
  conc = c(118,58,42,35,27,25,21,19,18,69,35,26,21,18,16,13,12,12),
  u = c(5,10,15,20,30,40,60,80,100, 5,10,15,20,30,40,60,80,100),
  lot = factor(c(rep(1, 9), rep(2, 9))))

## The maximum likelihood fit of the gamma regression model
clotting_ml <- glm(conc ~ log(u)*lot, data = clotting, family = Gamma(link = "log"))

## Maximum likelihood estimates and Wald statistics using maximum likelihood estimator of the dispersio
dispersion_ml <- MASS::gamma.dispersion(clotting_ml)
clotting_summary_ml <- summary(clotting_ml, dispersion = dispersion_ml)
clotting_ml_estimates <- coef(clotting_summary_ml)[, c("Estimate", "z value")]

## Maximum likelihood estimates and Wald statistics using the moment-based estimator of the dispersion
clotting_summary_mom <-  summary(clotting_ml)
dispersion_mom <- clotting_summary_mom$dispersion
clotting_mom_estimates <- coef(clotting_summary_mom)[, c("Estimate", "t value")]

## Location-adjusted Wald statistic
```

```r
clotting_corzed <- corzed(clotting_ml, null = 0, correction = TRUE)

## Table 3
table3 <- cbind(c(clotting_ml_estimates[, 1], dispersion_ml, dispersion_mom),
                c(clotting_ml_estimates[, 2], NA, NA),
                c(clotting_mom_estimates[, 2], NA, NA),
                c(clotting_corzed, NA, NA))
round(table3, 3)
#                [,1]    [,2]    [,3]    [,4]
# (Intercept)  5.503  34.124  29.282  28.953
# log(u)      -0.602 -12.842 -11.020 -10.896
# lot2        -0.584  -2.563  -2.199  -2.173
# log(u):lot2  0.034   0.520   0.446   0.441
#              0.017      NA      NA      NA
#              0.024      NA      NA      NA
```

Figure 2

`clotting_simulation.rda` below is the output of `clotting_simulation.R` in ./code, which replicates the simulation study described in Section 3.3 of Di Caterina and Kosmidis (2017)

```r
load(paste(results_path, "clotting_simulation.rda", sep = "/"))

## Compute type I error rates
typeI <- ddply(res, ~ statistic + parameter, function(x) {
    ## empirical <- pnorm(quantile(x$value, c(0, 1, 2.5, 5, 1)/100))
    levels <- c(0.1, 1, 2.5, 5)/100
    p_value_2sided <- 2 * pnorm(-abs(x$value))
    p_value_left <- pnorm(x$value)
    p_value_right <- 1 - pnorm(x$value)
    rate_2sided <- sapply(levels, function(alpha) mean(p_value_2sided < alpha))
    rate_left <- sapply(levels, function(alpha) mean(p_value_left < alpha))
    rate_right <- sapply(levels, function(alpha) mean(p_value_right < alpha))
    out <- data.frame(
        test = rep(c("2sided", "left", "right"), each = length(levels)),
        typeI = c(rate_2sided, rate_left, rate_right),
        level = rep(levels, times = 3))
    out
})
typeI <- typeI %>%
    filter(test != "right") %>%
    mutate(test = recode(test,
                         "2sided" = "beta[italic(j)] != beta[paste(italic(j), 0)]",
                         "left" = "beta[italic(j)] < beta[paste(italic(j), 0)]",
                         "right" = "beta[italic(j)] > beta[paste(italic(j), 0)]"),
           level_chr = paste(level*100, "~symbol('\045')"),
           upper = typeI - qnorm(1 - 0.01/2)*sqrt(typeI*(1-typeI)/nsimu),
           lower = typeI + qnorm(1 - 0.01/2)*sqrt(typeI*(1-typeI)/nsimu))

## Figure 2 in the manuscript
ggplot(typeI %>% filter(parameter != 1)) +
    geom_point(aes(parameter, typeI, pch = statistic), alpha = 0.7) +
```

```
    geom_hline(aes(yintercept = level), col = "grey", lty = 2) +
    facet_grid(test ~ level_chr, labeller = label_parsed, scales = "free") +
    scale_x_continuous(name = element_blank(),
                       breaks = c(2, 3, 4),
                       limits = c(1.8, 4.2),
                       labels = c(
                           expression(beta[2]),
                           expression(beta[3]),
                           expression(beta[4]))) +
    scale_y_continuous(name = expression(paste("Empirical rejection probability (", symbol('\045'), ")")
                       labels = function (x) {
                           if (length(x) == 0)
                               return(character())
                           x <- round_any(x, scales:::precision(x)/100)
                           scales:::comma(x * 100)
                       }) +
    theme_bw() +
    theme(legend.position = "none",
          panel.grid.major.y = element_blank(),
          panel.grid.minor.y = element_blank(),
          panel.grid.minor.x = element_blank(),
          strip.background = element_blank())
```

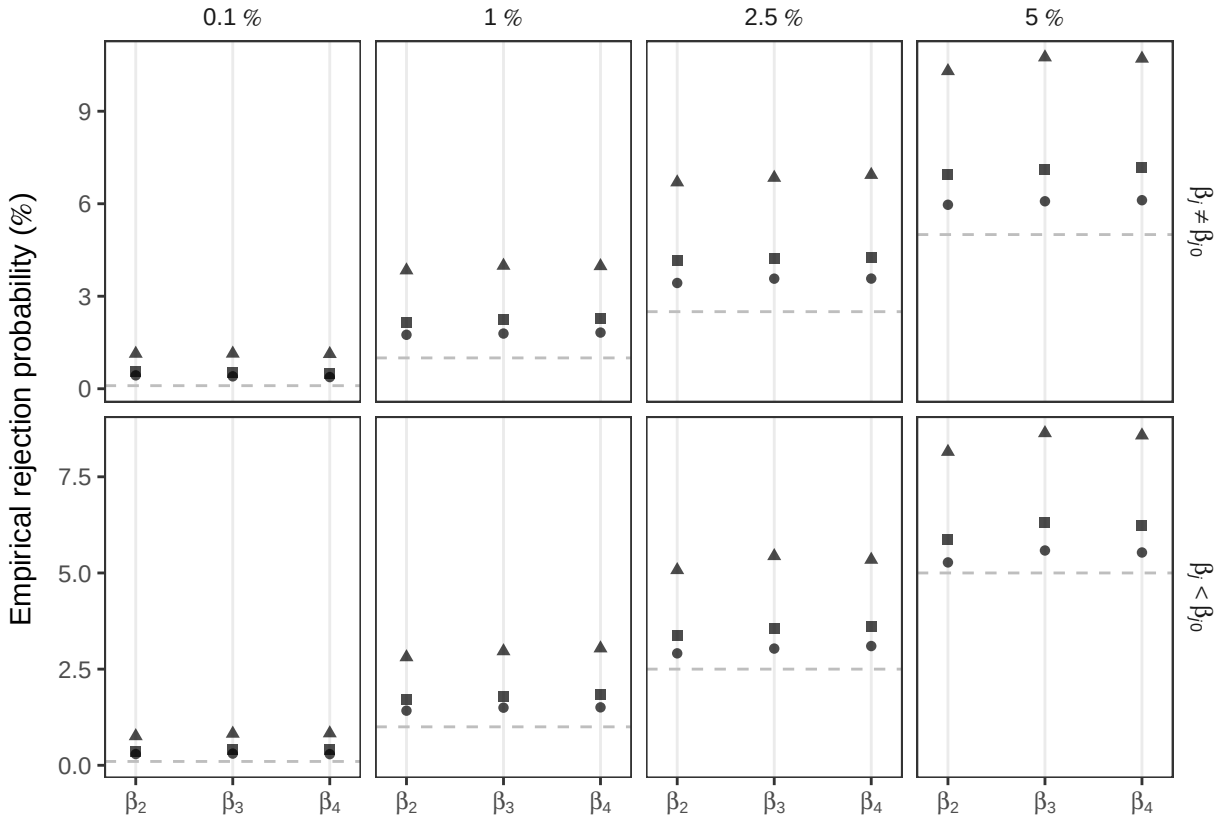

Table 4

```
data("babies", package = "cond")

## clogit understands only 0-1 so expand
babies_expand <- ddply(babies, ~ lull + day, function(z) {
    data.frame(y = rep(c(0, 1), c(z$r2, z$r1)))
})

## Maximum likelihood fit
babies_ml <- glm(formula = y ~ day + lull - 1,
                 family = binomial, data = babies_expand)

## Maximum conditional likelihood fit
babies_cond <- clogit(y ~ strata(day) + lull, data = babies_expand)

ml <- coef(summary(babies_ml))["lullyes", ]
mcl <- coef(summary(babies_cond))["lullyes", ]
r <- lrtest(update(babies_ml, . ~ . - lull),
            babies_ml)
rc <- summary(babies_cond)$logtest[1]
scorec <- summary(babies_cond)$sctest[1]
out1 <- c(
    ml = unname(ml["Estimate"]),
    mcl = unname(mcl["coef"]),
    wald_ml = unname(ml["z value"]),
    wald_mcl = unname(mcl["z"]),
    r = unname(sign(ml["Estimate"]) * sqrt(r$Chisq[2])),
    rc = unname(sign(mcl["coef"]) * sqrt(rc)),
    wald_adjusted = unname(corzed(babies_ml, what = 19)))
out2 <- c(
    ml_se = unname(ml["Std. Error"]),
    mcl_se = unname(mcl["se(coef)"]),
    ml_p = ml["Pr(>|z|)"],
    mcl_p = mcl["Pr(>|z|)"],
    r_p = 2 * pnorm(-abs(out1["r"])),
    rc_p = 2 * pnorm(-abs(out1["rc"])),
    cor_p = 2 * pnorm(-abs(out1["wald_adjusted"])))

## Table 4 in the manuscript
table4 <- matrix(c(out1, out2), ncol = 7, byrow = TRUE,
                 dimnames = list(NULL, c("mle", "mcle", "wald_ml", "wald_mlc", "r", "rc", "wald_adjuste
round(table4, 4)
#         mle   mcle wald_ml wald_mlc      r     rc wald_adjusted
# [1,] 1.4324 1.2561  1.9511   1.8307 2.1596 2.0214        1.9257
# [2,] 0.7341 0.6861  0.0510   0.0671 0.0308 0.0432        0.0541
```

Figure 3

`babies_simulation.rda` below is the output of `babies_simulation.R` in ./code, which replicates the simulation study described in Section 3.4 of Di Caterina and Kosmidis (2017)

```r
load(paste(results_path, "babies_simulation.rda", sep = "/"))

## Compute pvalues from the various statistics
pval <- ddply(res %>% filter(!infinite & !is.na(value)),
              ~ statistic,
              function(data) {
                  p2 <- 2*pnorm(-abs(data$value))
                  pl <- pnorm(data$value)
                  pr <- 1 - pl
                  data.frame(sample = c(p2, pl, pr),
                             test = rep(c("2sided", "left", "right"), each = length(p2))) })
pval <- pval %>%
    filter(statistic != "scorec" & test != "right") %>%
    mutate(test = dplyr::recode(test,
                                "2sided" = "gamma != 0",
                                "left" = "gamma < 0",
                                "right" = "gamma > 0"),
           statistic = factor(statistic,
                              levels = c("mle", "r", "cond", "scorec", "rc", "cor"), ordered = TRUE)) %
    mutate(statistic = dplyr::recode(statistic,
                                     "mle" = "italic(t)",
                                     "r" = "italic(r)",
                                     "cond" = "italic(t)[c]",
                                     "scorec" = "italic(s)[c]",
                                     "rc" = "italic(r)[c]",
                                     "cor" = "italic(t)^'*'"))

## Figure 3 in the manuscript
ggplot(pval) +
    geom_segment(aes(x = 0, xend = 1, y = 1, yend = 1)) +
    geom_histogram(aes(x = sample, y = ..density..), bins = 20, fill = "darkgray", col = "white", alpha
    facet_grid(test ~ statistic, labeller = label_parsed) +
    theme_bw() +
    theme(legend.position = "top",
          panel.grid.major.y = element_blank(),
          panel.grid.minor.y = element_blank(),
          panel.grid.minor.x = element_blank(),
          panel.grid.major.x = element_blank(),
          strip.background = element_blank(),
          axis.text.x = element_text(angle = 45, hjust = 1)) +
    coord_cartesian(ylim = c(0.4 , 1.25)) +
    labs(x = expression(paste("Empirical ", italic(p), "-value ")), y = "Density")
```

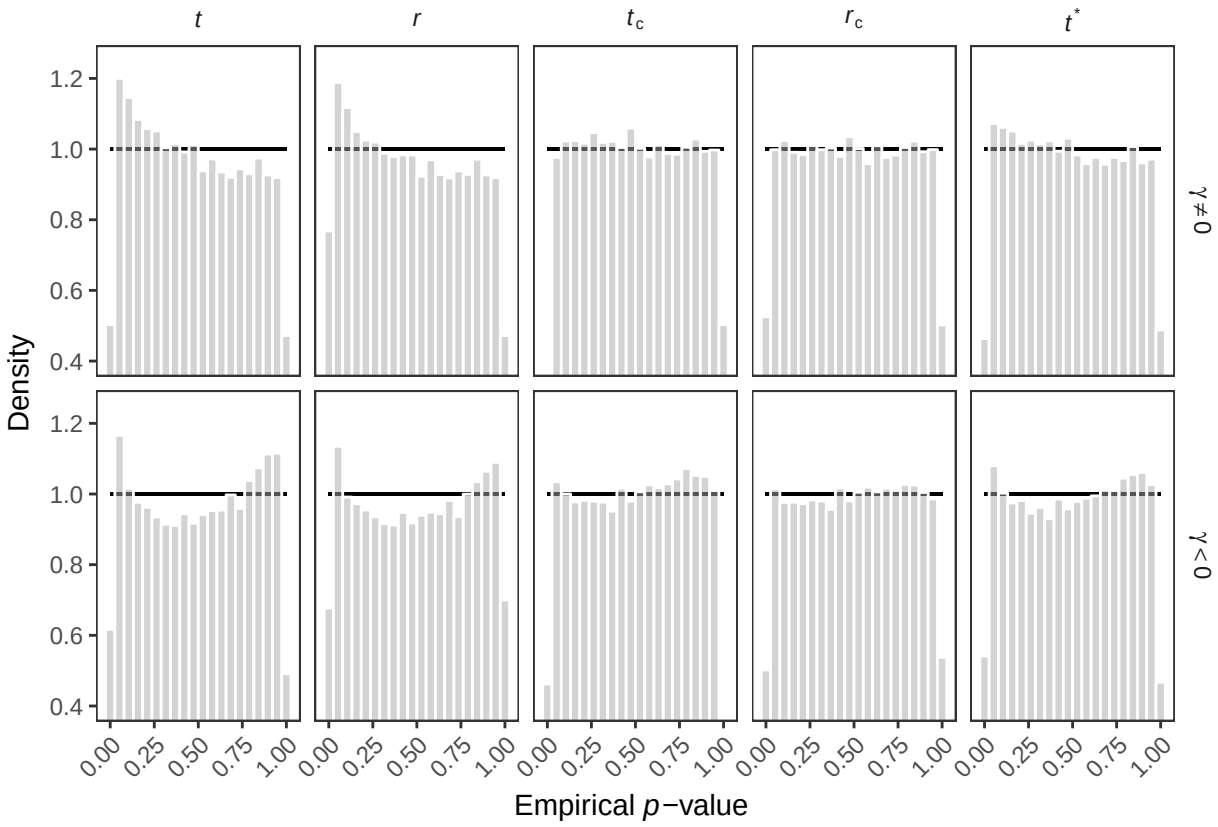

Figure 4

`brockwell_gordon_simulation.rda` below is the output of `brockwell_gordon_simulation.R` in ./code, which replicates the simulation study described in Section 4.3 of Di Caterina and Kosmidis (2017)

```
load(paste(results_path, "brockwell_gordon_simulation.rda", sep = "/"))

## Coverage plots: Coverage versus psi
n.val <- c(unname(Ks)[2], unname(Ks)[4])
j.val <- truepsis
p <- length(n.val)
df <- expand.grid(n = n.val, method = c("t", "tstar", "DL", "pr", "r"), j = j.val)
df$setting <- paste0("setting", seq_len(nrow(df)))
df <- ddply(df, ~ setting, function(dfc) {
  alpha <- dfc$alpha
  n <- dfc$n
  j <- dfc$j
  method <- dfc$method
  covt <- switch(as.character(method),
                 "t" = (1 - sizeML[2, which(truepsis==j), which(unname(Ks)==n)])*100,
                 "tstar" = (1 - sizeBC[2, which(truepsis==j), which(unname(Ks)==n)])*100,
                 "DL" = (1 - sizeDL[2, which(truepsis==j), which(unname(Ks)==n)])*100,
                 "pr" = (1 - sizeplr[2, which(truepsis==j), which(unname(Ks)==n)])*100,
                 "r" = (1 - sizer[2, which(truepsis==j), which(unname(Ks)==n)])*100
  )
  data.frame(n = n, j = j, method = method, covt = covt)
```

```
})
df$method <- factor(df$method, levels = c("t", "tstar", "DL", "pr", "r"), ordered = TRUE)
df$nstring <- paste0("K = ", df$n)
df$nstring <- factor(df$nstring, levels = c("K = 10", "K = 20"),
                     labels = c("italic(K) == 10", "italic(K) == 20"), ordered = TRUE)
plot_meta_lines1 <- ggplot(df) +
  geom_hline(aes(yintercept = 95), color = "grey") +
  geom_line(aes(j, covt, group = method, lty = method), alpha = 1) +
  facet_wrap(~ nstring, labeller = label_parsed, ncol = length(n.val)) +
  labs(y = "Coverage probability (%)", x = expression(psi)) +
  lims(y = c(85, 100)) +
    scale_x_continuous(name = expression(psi), breaks = seq(0, 0.1, length = 6),
                       labels = c("0.00", "0.02", "0.04", "0.06", "0.08", "0.10")) +
    scale_linetype_manual(name = "", values = c(2, 1, 3, 4, 5),
                          labels = c(expression(italic(t)), expression(italic(t)^list("*")), "DL",
                                     expression(italic(r)[p]), expression(italic(r)))) +
    theme_bw() +
    theme(text=element_text(size = 11)) +
    theme(legend.position = "none", panel.grid.major.y = element_blank(), panel.grid.minor.y = element_
        panel.grid.minor.x = element_blank(), strip.background = element_blank())

## Coverage plots: Coverage versus K
n.val <- unname(Ks)
j.val <- c(truepsis[4], truepsis[8])
p <- length(j.val)
df <- expand.grid(n = n.val, method = c("t", "tstar", "DL", "pr", "r"), j = j.val)
df$setting <- paste0("setting", seq_len(nrow(df)))
df <- ddply(df, ~ setting, function(dfc) {
  alpha <- dfc$alpha
  n <- dfc$n
  j <- dfc$j
  method <- dfc$method
  covt <- switch(as.character(method),
                 "t" = (1 - sizeML[2, which(truepsis==j), which(unname(Ks)==n)])*100,
                 "tstar" = (1 - sizeBC[2, which(truepsis==j), which(unname(Ks)==n)])*100,
                 "DL" = (1 - sizeDL[2, which(truepsis==j), which(unname(Ks)==n)])*100,
                 "pr" = (1 - sizeplr[2, which(truepsis==j), which(unname(Ks)==n)])*100,
                 "r" = (1 - sizer[2, which(truepsis==j), which(unname(Ks)==n)])*100
  )
  data.frame(n = n, j = j, method = method, covt = covt)
})
df$method <- factor(df$method, levels = c("t", "tstar", "DL", "pr", "r"), ordered = TRUE)
df$jstring <- paste0("psi = ", df$j)
df$jstring <- factor(df$jstring, levels = c("psi = 0.03", "psi = 0.07"),
                     labels = c("psi == 0.03", "psi == 0.07"), ordered = TRUE)
plot_meta_lines2 <- ggplot(df) +
  geom_hline(aes(yintercept = 95), color = "grey") +
  geom_line(aes(log(n), covt, group = method, lty = method), alpha = 1) +
  facet_wrap(~ jstring, labeller = label_parsed, ncol = length(j.val)) +
  labs(y = "Coverage probability (%)", x = expression(italic(K))) +
  lims(y = c(85, 100)) +
    scale_x_continuous(name = expression(italic(K)),
                       breaks = c(log(5), log(10), log(25), log(50), log(100), log(200)),
```

```
                        labels = c("5", "10", "25", "50", "100", "200")) +
    scale_linetype_manual(name = "", values = c(2, 1, 3, 4, 5),
                          labels = c(expression(italic(t)), expression(italic(t)^list("*")), "DL",
                                     expression(italic(r)[p]), expression(italic(r)^list("*")))) +
    theme_bw() +
    theme(text=element_text(size = 11)) +
    theme(legend.position = "none", panel.grid.major.y = element_blank(), panel.grid.minor.y = element_
          panel.grid.minor.x = element_blank(), strip.background = element_blank())

## Figure 4 in manuscript
grid.arrange(plot_meta_lines1, plot_meta_lines2, ncol = 1, nrow = 2)
```

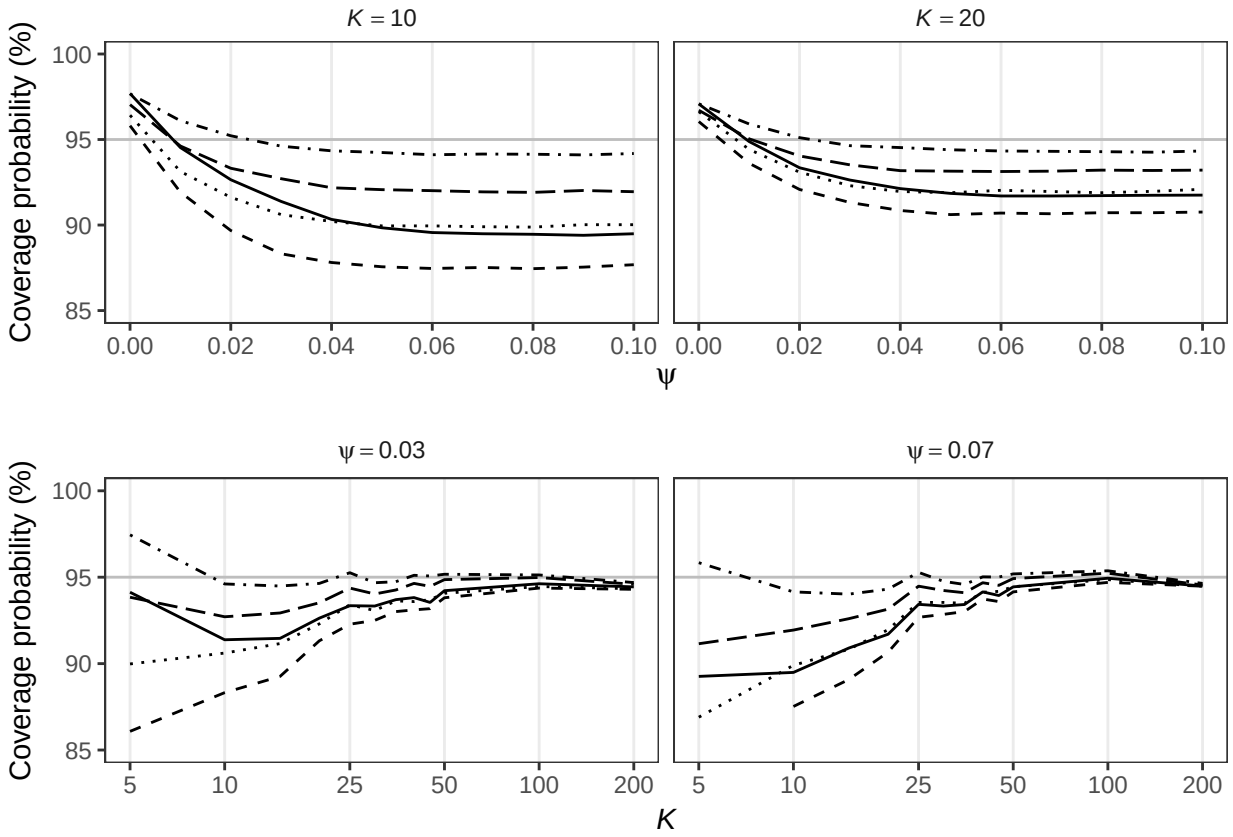

Table 5

```
## Compute confidence intervals by inversion of the location-adjusted Wald statistic
corzed_cis <- corzed_ci(rs_beta_ml, level = 0.95, correction = TRUE)

## Table 5 in the manuscript
table5 <- cbind(corzed_cis[-c(1, 5),],
                rs_coverage %>% select(parameter, level, coverage.cor) %>%
                reshape(idvar = c("parameter"), v.names = "coverage.cor", timevar = "level", direction =
                select(-parameter))
round(table5, 3)
#                2.5 % 97.5 % coverage.cor.90 coverage.cor.95 coverage.cor.99
# dyslexia      -1.019 -0.435            88.5            93.7            98.4
```

```
# iq              0.204  0.752          87.1          92.8          98.0
# dyslexia:iq    -0.845 -0.299          87.2          92.8          98.0
# (phi)_dyslexia  1.186  2.214          83.5          90.0          96.6
# (phi)_iq        0.639  1.691          81.8          88.6          95.7
```

References

Di Caterina, Claudia, and Ioannis Kosmidis. 2017. "Location-Adjusted Wald Statistic for Scalar Parameters." *ArXiv E-Prints*. https://arxiv.org/abs/1710.11217.

R Core Team. 2017. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.