Supplemental file for "Meta-analysis of proportions using generalized linear mixed models"

eAppendix A. Illustration of SAS code for meta-analyses of proportions via GLMMs

The following SAS code illustrates the implementation of meta-analyses of proportions via PROC NLMIXED based on the real data example on chorioamnionitis. The log, logit, probit, cauchit, and cloglog links are considered. Alternative procedures such as PROC GLIMMIX may be also used. Of note, by default PROC NLMIXED uses adaptive Gauss–Hermite quadrature to approximate the integrated likelihood; the produced results may be different from the default approximation methods implemented in PROC GLIMMIX or certain R functions (see, e.g., Zhang et al., https://doi.org/10.1002/sim.4265). As a result, the standard errors (and thus the confidence intervals) may be different. The following code was implemented in SAS version 9.4.

```
data chor;
  input study e n;
  datalines;
  1       1      165
  2       1      143
  3  134413 10458616
  4   92622  5338995
  5  110747  6018504
  6    3625   190810
  7      35     1607
  8   64695  2504824
  9   19428   471821
  10    221     5158
  11    913    15027
  12     13      205
  13   5710    86371
  14    121     1785
  15   1339    10661
  16   1851    14406
  17    637     4048
  18     20      102
  19      7      916
  20     10     1079
  21   2508   242715
  ;
run;

/* log link */
proc nlmixed data = chor qpoints = 200;
  parms mu = -1 tau = 1;
  eta = mu + u;
  p = exp(eta);
  model e ~ binomial(n, p);
  random u ~ normal(0, tau*tau) subject = study;
  ods output ParameterEstimates = chor_log;
run;
data chor_log_prop;
  set chor_log(obs = 1);
  prop_est = exp(Estimate) * 100;
  prop_lb = exp(Lower) * 100;
  prop_ub = exp(Upper) * 100;
  keep prop_est prop_lb prop_ub;
run;
proc print data = chor_log_prop; run;
```

```
/* logit link */
proc nlmixed data = chor qpoints = 200;
  parms mu = 0 tau = 1;
  eta = mu + u;
  p = exp(eta) / (1 + exp(eta));
  model e ~ binomial(n, p);
  random u ~ normal(0, tau*tau) subject = study;
  ods output ParameterEstimates = chor_logit;
run;
data chor_logit_prop;
  set chor_logit(obs = 1);
  prop_est = exp(Estimate) / (1 + exp(Estimate)) * 100;
  prop_lb = exp(Lower) / (1 + exp(Lower)) * 100;
  prop_ub = exp(Upper) / (1 + exp(Upper)) * 100;
  keep prop_est prop_lb prop_ub;
run;
proc print data = chor_logit_prop; run;

/* probit link */
proc nlmixed data = chor qpoints = 200;
  parms mu = 0 tau = 1;
  eta = mu + u;
  p = probnorm(eta);
  model e ~ binomial(n, p);
  random u ~ normal(0, tau*tau) subject = study;
  ods output ParameterEstimates = chor_probit;
run;
data chor_probit_prop;
  set chor_probit(obs = 1);
  prop_est = probnorm(Estimate) * 100;
  prop_lb = probnorm(Lower) * 100;
  prop_ub = probnorm(Upper) * 100;
  keep prop_est prop_lb prop_ub;
run;
proc print data = chor_probit_prop; run;

/* cauchit link */
proc nlmixed data = chor qpoints = 200;
  parms mu = 0 tau = 1;
  eta = mu + u;
  p = 0.5 + atan(eta)/constant("pi");
  model e ~ binomial(n, p);
  random u ~ normal(0, tau*tau) subject = study;
  ods output ParameterEstimates = chor_cauchit;
run;
data chor_cauchit_prop;
  set chor_cauchit(obs = 1);
  prop_est = (0.5 + atan(Estimate)/constant("pi")) * 100;
  prop_lb = (0.5 + atan(Lower)/constant("pi")) * 100;
  prop_ub = (0.5 + atan(Upper)/constant("pi")) * 100;
  keep prop_est prop_lb prop_ub;
run;
proc print data = chor_cauchit_prop; run;

/* cloglog link */
proc nlmixed data = chor qpoints = 200;
  parms mu = 0 tau = 1;
  eta = mu + u;
  p = 1 - exp(-exp(eta));
  model e ~ binomial(n, p);
  random u ~ normal(0, tau*tau) subject = study;
  ods output ParameterEstimates = chor_cloglog;
```

```
run;
data chor_cloglog_prop;
  set chor_cloglog(obs = 1);
  prop_est = (1 - exp(-exp(Estimate))) * 100;
  prop_lb = (1 - exp(-exp(Lower))) * 100;
  prop_ub = (1 - exp(-exp(Upper))) * 100;
  keep prop_est prop_lb prop_ub;
run;
proc print data = chor_cloglog_prop; run;
```

eAppendix B. Illustration of R code for meta-analyses of proportions via GLMMs

To illustrate the implementation of meta-analyses of proportions via the GLMM approach, we use three popular R packages, i.e., "lme4" (version 1.1-23), "metafor" (version 2.4-0), and "meta" (version 4.11-0). These versions are up-to-date as of April 8, 2020. The first package is designed for general purposes of implementing GLMs and GLMMs; the latter two are specially built for meta-analyses. The involved functions include glmer() in "lme4," rma.glmm() in "metafor," and metaprop() in "meta." The function metaprop() in "meta" is built on the function rma.glmm() in "metafor," and rma.glmm() calls glmer() from "lme4." Various links can be used for GLMMs when using the function glmer() in "lme4," but the functions rma.glmm() in "metafor" and metaprop() in "meta" mainly support the logit link for proportions.

Of note, the earlier versions of the "metafor" and "meta" packages may produce warnings about the convergence for estimating GLMM parameters. Nevertheless, such warnings can be often safely ignored, as they do not have much impact on the primary results about proportions.

The following R code consists of two parts: 1) loading the two illustrative datasets; 2) illustrating the implementation of meta-analyses of proportions via GLMMs with the focus on the logit link. We use the dataset on chorioamnionitis. Various other links are considered in our primary analyses, whose complete code is given in eAppendix F.

```
## Install R packages
#install.packages("lme4")
#install.packages("metafor")
#install.packages("meta")

library("lme4")
library("metafor")
library("meta")

############################################################
## Two illustrative datasets
############################################################

## Chorioamnionitis data from
## Woodd et al. (2019, PLOS Medicine)
## https://doi.org/10.1371/journal.pmed.1002984
chor <- data.frame(e = c(1, 1, 134413, 92622, 110747,
    3625, 35, 64695, 19428, 221, 913, 13, 5710, 121,
    1339, 1851, 637, 20, 7, 10, 2508),
  n = c(165, 143, 10458616, 5338995, 6018504, 190810,
    1607, 2504824, 471821, 5158, 15027, 205, 86371, 1785,
    10661, 14406, 4048, 102, 916, 1079, 242715))

## Depression data from
## Rotenstein et al. (2016, JAMA)
## https://doi.org/10.1001/jama.2016.17324
## Beck Depression Inventory Score >= 17
beck17 <- data.frame(e = c(15, 95, 24, 77, 36, 232),
  n = c(128, 164, 202, 352, 354, 668))
```

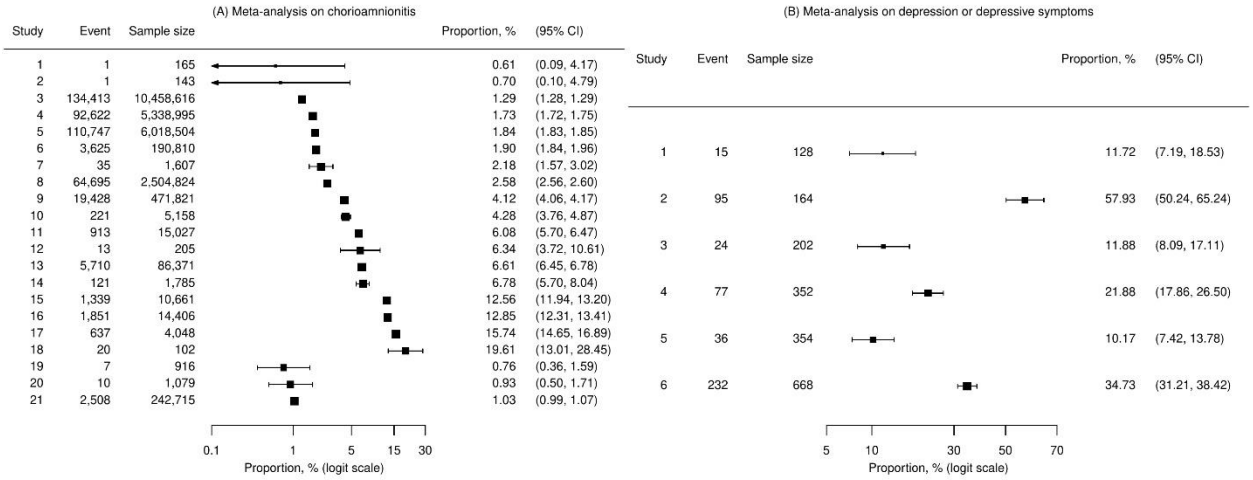
```
############################################################
## Illustration of implementing meta-analysis of
##   proportions via generalized linear mixed models
##   based on the chorioamnionitis data
############################################################

## Using the function glmer() in the package "lme4"

# link specifies the link function, which can be
#   "log", "logit", "probit", "cauchit", "cloglog";
#   here, we use the logit link for illustration
rslt1 <- glmer(cbind(e, n - e) ~ 1 + (1 | sid),
  data = cbind(sid = 1:length(chor$e), chor),
  family = binomial(link = "logit"))
rslt1

glmer(cbind(e, n - e) ~ 1 + (1 | sid),
  data = cbind(sid = 1:length(chor$e), chor),
  family = binomial(link = "log"))

#Generalized linear mixed model fit by maximum likelihood (Laplace
#  Approximation) [glmerMod]
# Family: binomial  ( logit )
#Formula: cbind(e, n - e) ~ 1 + (1 | sid)
#   Data: cbind(sid = 1:length(chor$e), chor)
#      AIC       BIC    logLik  deviance  df.resid
# 330.3775  332.4665 -163.1887  326.3775        19
#Random effects:
# Groups Name        Std.Dev.
# sid    (Intercept) 1.049
#Number of obs: 21, groups:  sid, 21
#Fixed Effects:
#(Intercept)
#      -3.37
mu.est1 <- summary(rslt1)$coefficients[1]
mu.se1 <- summary(rslt1)$coefficients[2]
mu.ci1 <- c(mu.est1 - qnorm(0.975) * mu.se1,
  mu.est1 + qnorm(0.975) * mu.se1)
back.trans <- function(x) 1/(1 + exp(-x))
back.trans(c(mu.est1, mu.ci1)) * 100 # overall proportion (%) with 95% CI
#[1] 3.325553 2.125086 5.168358


## Using the function rma.glmm() in the package "metafor"

# measure = "PLO" specifies the logit transformed proportion
rslt2 <- rma.glmm(xi = e, ni = n, data = chor,
  measure = "PLO", method = "ML")
rslt2
#Random-Effects Model (k = 21; tau^2 estimator: ML)
#
#tau^2 (estimated amount of total heterogeneity): 1.1028
#tau (square root of estimated tau^2 value):      1.0501
#I^2 (total heterogeneity / total variability):   99.9945%
#H^2 (total variability / sampling variability):  18316.1939
#
#Tests for Heterogeneity:
#Wld(df = 20) = 64965.6328, p-val < .0001
#LRT(df = 20) = 53174.0499, p-val < .0001
#
#Model Results:
#
#estimate      se      zval    pval    ci.lb    ci.ub
```

```
# -3.3696  0.2362  -14.2633  <.0001  -3.8327  -2.9066  ***
#
#---
#Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
back.trans(c(rslt2$b, rslt2$ci.lb, rslt2$ci.ub)) * 100
#[1] 3.325807 2.119294 5.182817


## Using the function metaprop() in the package "meta"

rslt3 <- metaprop(event = e, n = n, data = chor,
  method = "GLMM", sm = "PLOGIT")
rslt3
#   proportion           95%-CI
#1      0.0061 [0.0002; 0.0333]
#2      0.0070 [0.0002; 0.0383]
#3      0.0129 [0.0128; 0.0129]
#4      0.0173 [0.0172; 0.0175]
#5      0.0184 [0.0183; 0.0185]
#6      0.0190 [0.0184; 0.0196]
#7      0.0218 [0.0152; 0.0302]
#8      0.0258 [0.0256; 0.0260]
#9      0.0412 [0.0406; 0.0417]
#10     0.0428 [0.0375; 0.0487]
#11     0.0608 [0.0570; 0.0647]
#12     0.0634 [0.0342; 0.1060]
#13     0.0661 [0.0645; 0.0678]
#14     0.0678 [0.0566; 0.0805]
#15     0.1256 [0.1194; 0.1320]
#16     0.1285 [0.1231; 0.1341]
#17     0.1574 [0.1463; 0.1689]
#18     0.1961 [0.1241; 0.2865]
#19     0.0076 [0.0031; 0.0157]
#20     0.0093 [0.0045; 0.0170]
#21     0.0103 [0.0099; 0.0107]
#
#Number of studies combined: k = 21
#
#                     proportion           95%-CI
#Fixed effect model       0.0173 [0.0173; 0.0174]
#Random effects model     0.0333 [0.0212; 0.0518]
#
#Quantifying heterogeneity:
# tau^2 = 1.1028; tau = 1.0501; I^2 = 100.0%; H = 135.34
#
#Test of heterogeneity:
#        Q d.f. p-value             Test
# 64965.63   20       0        Wald-type
# 53174.05   20       0 Likelihood-Ratio
#
#Details on meta-analytical method:
#- Random intercept logistic regression model
#- Maximum-likelihood estimator for tau^2
#- Logit transformation
#- Clopper-Pearson confidence interval for individual studies
```

eAppendix C. Descriptions of two real data examples

eFigure presents the forest plots of the meta-analyses reported by Woodd et al. (2019, https://doi.org/10.1371/journal.pmed.1002984) on chorioamnionitis and by Rotenstein et al. (2016, http://dx.doi.org/10.1001/jama.2016.17324) on depression or depressive symptoms. Of note, the original article by Rotenstein et al. stratified all extracted studies based on various screening instruments and cutoff scores; we focused on the meta-analysis of 6 studies with Beck Depression Inventory Score ≥17. Each forest plot presents study-specific event counts, sample sizes, estimated proportions with 95% CIs derived based on the logit transformation.

In the first meta-analysis, the median study-specific sample size is 5,158 with an interquartile range (IQR) 916–242,715, and the median study-specific crude proportion estimate is 2.18% (IQR, 1.03%–6.61%). In the second meta-analysis, the study-specific sample sizes have a median of 202 (IQR, 164–352), and the crude proportion estimates have a median of 11.88% (IQR, 11.72%–21.88%). The sample sizes are generally much smaller than the first meta-analysis, and the estimated proportions are higher.



eFigure. Forest plots of the two meta-analyses on chorioamnionitis (A) and on depression or depressive symptoms (B).

eAppendix D. Real data analyses using the two-step methods via the REML estimation

GLMMs are usually estimated via the maximum-likelihood (ML) approach, as in the function glmer() in "lme4," rma.glmm() in "metafor," and metaprop() in "meta." The restricted maximum-likelihood (REML) approach is not commonly applied to GLMMs, primarily because of computational difficulties. To provide fair comparisons between the estimates produced by GLMMs and two-step methods, the main content uses the ML estimation for both methods. Nevertheless, the REML estimation has been known to have superior performance when using the two-step methods; see, e.g., Langan et al. (2019, https://doi.org/10.1002/jrsm.1316).

We also obtained the estimates from two-step methods via the REML approach for the two real-world meta-analyses; eTable 1 presents the results. The point estimates of the overall median proportions were very similar to those produced via the ML approach. The point estimates of the population-averaged proportions and the interval estimates were slightly different from those produced via the ML approach, likely because these estimates depend on the between-study variance, and the ML and REML approaches mainly differ in terms of the estimation of variance parameters.

eTable 1. Overall proportion estimates with 95% confidence intervals in the two meta-analyses on chorioamnionitis and on depression or depressive symptoms using the two-step methods via the REML estimation.

Method	Overall median proportion, % [95% bootstrap CI] (95% CI)	Population-averaged proportion, % [95% bootstrap CI]
Meta-analysis on chorioamnionitis		
Log transformation	3.38 [2.20, 5.18] (2.18, 5.26)	5.53 [3.20, 8.65]
Logit transformation	3.44 [2.21, 5.30] (2.19, 5.37)	5.35 [3.18, 7.99]
Arcsine transformation	4.13 [2.55, 6.28] (2.45, 6.22)	5.22 [3.17, 7.62]
Double-arcsine transformation at		
harmonic mean of sample sizes = 664	4.14 [2.58, 6.30] (2.46, 6.23)	5.21 [3.17, 7.61]
geometric mean of sample sizes = 17,358	4.21 [2.65, 6.36] (2.53, 6.30)	5.27 [3.24, 7.67]
arithmetic mean of sample sizes = 1,207,998	4.21 [2.65, 6.36] (2.53, 6.30)	5.27 [3.24, 7.67]
inverse of variance = 436	4.11 [2.55, 6.26] (2.42, 6.20)	5.18 [3.14, 7.58]
Meta-analysis on depression or depressive symptoms		
Log transformation	20.31 [12.71, 35.24] (11.52, 35.78)	25.76 [13.26, 42.63]
Logit transformation	21.66 [12.74, 37.07] (11.15, 37.85)	25.31 [13.22, 38.68]
Arcsine transformation	23.10 [12.81, 37.77] (10.64, 38.60)	25.39 [13.08, 38.67]
Double-arcsine transformation at		
harmonic mean of sample sizes = 231	23.12 [12.83, 37.78] (10.65, 38.60)	25.38 [13.10, 38.71]
geometric mean of sample sizes = 266	23.14 [12.85, 37.79] (10.67, 38.60)	25.40 [13.12, 38.71]
arithmetic mean of sample sizes = 311	23.15 [12.87, 37.80] (10.69, 38.61)	25.41 [13.14, 38.72]
inverse of variance = 34	22.47 [11.94, 37.49] (9.71, 38.32)	24.83 [12.22, 38.45]

eAppendix E. Simulation studies

In addition to the two illustrative real data examples, we also conducted simulation studies to compare the performance of the various methods for meta-analyses of proportions. The meta-analyses were simulated based on the GLMM with the logit link. We chose the logit link because it transforms proportions to the whole real range $(-\infty, \infty)$, and it is conventional to assume the logit proportions to follow the normal distribution. Therefore, the simulation settings may favor the GLMM with the logit link.

We considered various simulation settings. The number of studies in a meta-analysis was set to N= 5 (small meta-analysis) or 30 (large mete-analysis). The true overall median proportion p was set to 1% (relatively rare event) and 20% (common event). The between-study standard deviation of proportions on the logit scale was set to τ=0.1 or 0.5. Moreover, sample sizes within studies n_i were selected from two sets {100, 200, 300, 400, 500} (relatively small sample sizes) and {1000, 2000, 3000, 4000, 5000} (relatively large sample sizes); for simulated meta-analyses with N=30 studies, the five values in each set were duplicated for six times to yield 30 sample sizes.

In each setting, the logit-transformed overall proportion was $\mu = \log[p/(1-p)]$. The study-specific true proportions (on the logit scale) were sampled as $\mu_i \sim N(\mu, \tau^2)$ for i=1, …, N, so the true proportions were $p_i = e^{\mu_i}/(1+e^{\mu_i})$, and the event counts were subsequently sampled as $e_i \sim bin(n_i, p_i)$. The values $\{(e_i, n_i)\}_{i=1}^{N}$ formed the data of a simulated meta-analysis. We applied GLMMs with the log, logit, probit, cauchit, and cloglog links, as well as two-step methods with the log, logit, arcsine, and double-arcsine (at the harmonic, geometric, and arithmetic means of sample sizes and at the inverse of variance) transformations to each simulated meta-analysis. In each setting, 1000 meta-analyses were simulated.

We may roughly quantify the heterogeneity in each simulation setting as follows. The I^2 statistic is about $\frac{\tau^2}{\tau^2+1/[np(1-p)]} \times 100\%$, where n is the median sample size among the n_i's (i.e., 300 or 3000 in the settings), and $1/[np(1-p)]$ roughly represents the within-study variance. When p=1% and n_i=100–500, I^2≈3% for τ=0.1 and I^2≈43% for τ=0.5; when p=1% and n_i=1000–5000, I^2≈23% for τ=0.1 and I^2≈88% for τ=0.5; when p=20% and n_i=100–500, I^2≈32% for τ=0.1 and I^2≈92% for τ=0.5; and when p=20% and n_i=1000–5000, I^2≈83% for τ=0.1 and I^2≈99% for τ=0.5.

Of note, when implementing GLMMs and two-step methods via the ML estimation, the algorithm may not sufficiently converge, and the R functions, including glmer() in "lme4" and rma() in "metafor," used in our simulation studies may report warnings or errors. We discarded simulated meta-analyses that led to any warning or error, and continued the simulations until 1000 replicates without any warning or error were obtained.

We evaluated the performance of each meta-analysis method in terms of bias, root mean squared error, and 95% CI coverage probability. eTable 2 presents the results

among simulated meta-analysis with N=5 studies, and Table 2 (in the main content) presents those with N=30 studies.

eTable 2. Biases, root mean squared errors (RMSEs), and 95% confidence interval coverage probabilities (CPs, in percentage, %) of the estimated overall median proportions (in percentage, %) based on 1000 simulated meta-analyses with N=5 studies under various simulation settings.

Method	p=1%						p=20%					
	τ=0.1			τ=0.5			τ=0.1			τ=0.5		
	Bias	RMSE	CP	Bias	RMSE	CP	Bias	RMSE	CP	Bias	RMSE	CP
Sample sizes ranging in 100–500												
Generalized linear mixed model:												
Log	−0.01	0.26	96	0.05	0.37	90	0.06	1.32	91	−0.08	3.59	84
Logit	−0.01	0.26	96	0.05	0.37	90	0.06	1.32	92	0.18	3.65	84
Probit	−0.01	0.26	96	0.05	0.38	90	0.07	1.32	92	0.33	3.66	84
Cauchit	0.00	0.26	96	0.06	0.37	88	0.04	1.32	92	−0.67	3.76	85
Complementary log-log	−0.01	0.26	96	0.05	0.37	90	0.06	1.32	92	0.05	3.62	84
Two-step method:												
Log	0.20	0.33	89	0.34	0.52	78	0.17	1.28	93	0.15	3.61	82
Logit	0.20	0.33	89	0.34	0.52	79	0.09	1.27	93	0.29	3.65	83
Arcsine	0.03	0.25	97	0.12	0.37	91	−0.03	1.27	93	0.49	3.70	84
Double-arcsine (harmonic)	−0.02	0.25	98	0.08	0.36	92	−0.06	1.28	93	0.49	3.70	84
Double-arcsine (geometric)	0.02	0.25	98	0.11	0.37	93	−0.04	1.27	92	0.51	3.70	84
Double-arcsine (arithmetic)	0.04	0.25	98	0.14	0.38	92	−0.02	1.27	93	0.53	3.70	83
Double-arcsine (IV)	0.16	0.29	95	0.25	0.43	89	0.05	1.27	93	0.44	3.68	84
Sample sizes ranging in 1000–5000												
Generalized linear mixed model:												
Log	0.00	0.09	92	0.02	0.25	84	−0.01	0.81	84	0.03	3.50	84
Logit	0.00	0.09	92	0.03	0.25	84	0.00	0.81	84	0.36	3.61	84
Probit	0.00	0.09	93	0.04	0.25	84	0.00	0.81	84	0.55	3.64	84
Cauchit	0.00	0.09	94	−0.03	0.24	84	−0.04	0.81	84	−0.81	3.70	84
Complementary log-log	0.00	0.09	92	0.03	0.25	84	−0.01	0.81	84	0.20	3.55	84
Two-step method:												
Log	0.02	0.10	93	0.06	0.26	83	0.00	0.80	84	0.08	3.52	84
Logit	0.02	0.10	93	0.06	0.26	84	0.00	0.80	84	0.40	3.62	84
Arcsine	0.00	0.10	93	0.07	0.28	84	0.00	0.80	84	0.84	3.72	84
Double-arcsine (harmonic)	−0.01	0.10	93	0.07	0.28	84	0.00	0.80	84	0.84	3.72	84
Double-arcsine (geometric)	0.00	0.10	93	0.07	0.28	84	0.00	0.80	84	0.84	3.72	84
Double-arcsine (arithmetic)	0.00	0.10	93	0.07	0.28	84	0.00	0.80	84	0.84	3.72	84
Double-arcsine (IV)	0.01	0.10	93	0.07	0.27	84	0.00	0.80	84	0.67	3.67	84

Note: p denotes the true overall median proportion, and τ denotes the between-study standard deviation (of proportions on the logit scale). For the double-arcsine transformation used by the two-step method, the "harmonic," "geometric," and "arithmetic" denote the harmonic, geometric, and arithmetic means of sample sizes, and the "IV" denotes the inverse of variance.

eAppendix F. R code for reproducing all analyses

The following R code is used to perform all analyses (for both real and simulated datasets) and generate forest plots of the two real data examples. In these analyses, we use the “lme4” package for meta-analyses of proportions via GLMMs and use the “metafor” package for those via two-step methods.

Also, based on the glmer() function, we develop functions, named maprop.glmm() and maprop.twostep(), to further produce the population-averaged (marginalized) proportion. This quantity cannot be readily estimated by the packages “lme4,” “metafor,” and “meta.” To obtain the CI of the population-averaged proportion, we use the bootstrap resampling method, which takes longer time than implementing the illustrative code described in eAppendix B.

Before implementing the following R code, users need to load the two real datasets in eAppendix B.

```
############################################################
## Reproducible code for all results in the main content
############################################################

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

## Function for catching warnings and errors;
##   it is used when deriving the bootstrap CI, as the
##   optimization algorithm may not converge from some
##   bootstrapped samples.
tryCatch.W.E <- function(expr){
  W <- NULL
  w.handler <- function(w){
    W <<- w
    invokeRestart("muffleWarning")
  }
  list(value = withCallingHandlers(tryCatch(expr,
    error = function(e) e), warning = w.handler), warning = W)
}

## Function for the generalized linear mixed model that
##   produces both the median overall proportion (conditional proportion)
##   and the population-averaged proportion (marginal proportion)
##   with bootstrap CI.
## Arguments:
## e and n specify the event counts and sample sizes;
## data specifies the dataset;
## link specifies the link function, which can be
##   "log" (log link),
##   "logit" (corresponding to the logistic CDF),
##   "probit" (corresponding to the normal CDF),
##   "cauchit" (corresponding to the Cauchy CDF), or
##   "cloglog" (the complementary log-log link);
## alpha specifies the significance level;
## pop.avg specifies a logical value indicating whether the
##   population-averaged (marginal) proportion will be produced;
## int.approx specifies the number of standard normal samples
##   for approximating the population-averaged proportion;
## b.iter specifies the number of iterations for deriving
##  the bootstrap CI;
## seed specifies the random seed for reproducing the results.
maprop.glmm <- function(e, n, data, link = "logit",
  alpha = 0.05, pop.avg = TRUE, int.approx = 10000, b.iter = 1000,
  seed = 1234){
  if(missing(e)) stop("need to specify events.")
  if(missing(n)) stop("need to specify sample sizes.")
  if(!missing(data)){
    e <- eval(substitute(e), data, parent.frame())
    n <- eval(substitute(n), data, parent.frame())
  }
  if(length(e) != length(n)) stop("lengths of e and n differ.")
  N <- length(e)
  data <- data.frame(sid = 1:N, e = e, n = n)
  rslt <- glmer(cbind(e, n - e) ~ 1 + (1 | sid),
    data = data, family = binomial(link = link))
  mu.est <- summary(rslt)$coefficients[1]
  mu.se <- summary(rslt)$coefficients[2]
  mu.ci <- c(mu.est - qnorm(1 - alpha/2) * mu.se,
    mu.est + qnorm(1 - alpha/2) * mu.se)
  tau.est <- sqrt(as.numeric(summary(rslt)$varcor))
  if(link == "log"){
```

```
  back.trans <- function(x) exp(x)
}
if(link == "logit"){
  back.trans <- function(x) plogis(x)
}
if(link == "probit"){
  back.trans <- function(x) pnorm(x)
}
if(link == "cauchit"){
  back.trans <- function(x) pcauchy(x)
}
if(link == "cloglog"){
  back.trans <- function(x) 1 - exp(-exp(x))
}
back.trans <- Vectorize(back.trans)
prop.c.est <- back.trans(mu.est)
prop.c.ci <- back.trans(mu.ci)
out <- list(prop.c.est = prop.c.est, prop.c.ci = prop.c.ci)
if(pop.avg){
  if(link == "probit"){
    prop.m.est <- pnorm(mu.est/sqrt(1 + tau.est^2))
  }else{
    set.seed(seed)
    stdnorms <- rnorm(int.approx)
    prop.m.est <- mean(back.trans(mu.est + tau.est * stdnorms))
  }
  mu.est.b <- prop.m.est.b <- rep(NA, b.iter)
  set.seed(seed)
  b.idx <- 0
  b.w <- b.e <- 0
  while(b.idx < b.iter){
    idx.temp <- sample(N, replace = TRUE)
    data.temp <- data[idx.temp,]
    data.temp$sid <- 1:N
    suppressMessages(out.W.E <- tryCatch.W.E(
      rslt.temp <- glmer(cbind(e, n - e) ~ 1 + (1 | sid),
        data = data.temp, family = binomial(link = link))))
    if(is.null(out.W.E$warning) & !inherits(out.W.E$value, "error")){
      b.idx <- b.idx + 1
      mu.est.b[b.idx] <- summary(rslt.temp)$coefficients[1]
      tau.est.temp <- sqrt(as.numeric(summary(rslt.temp)$varcor))
      if(link == "probit"){
        prop.m.est.b[b.idx] <-
          pnorm(mu.est.b[b.idx]/sqrt(1 + tau.est.temp^2))
      }else{
        prop.m.est.b[b.idx] <- mean(back.trans(mu.est.b[b.idx] +
          tau.est.temp * stdnorms))
      }
    }
    if(!is.null(out.W.E$warning)){
      b.w <- b.w + 1
    }
    if(inherits(out.W.E$value, "error")){
      b.e <- b.e + 1
    }
  }
  mu.ci.b <- quantile(mu.est.b, probs = c(alpha/2, 1 - alpha/2))
  prop.c.ci.b <- back.trans(mu.ci.b)
  prop.m.ci.b <- quantile(prop.m.est.b,
    probs = c(alpha/2, 1 - alpha/2))
  out <- c(out, list(prop.c.ci.b = prop.c.ci.b,
    prop.m.est = prop.m.est, prop.m.ci.b = prop.m.ci.b,
    b.w.e = c("bootstrap warnings" = b.w, "bootstrap errors" = b.e)))
```

```
  }
  return(out)
}

## Function for the two-step methods.
## Arguments are similar to those of maprop.glmm();
## link specifies the transformation from the
##   proportion scale to a linear scale, which can be
##   "log", "logit", "arcsine", or "double.arcsine" (Freeman--Tukey)
maprop.twostep <- function(e, n, data, link = "logit", method = "ML",
  alpha = 0.05, pop.avg = TRUE, int.approx = 10000, b.iter = 1000,
  seed = 1234){
  if(missing(e)) stop("need to specify events.")
  if(missing(n)) stop("need to specify sample sizes.")
  if(!missing(data)){
    e <- eval(substitute(e), data, parent.frame())
    n <- eval(substitute(n), data, parent.frame())
  }
  if(length(e) != length(n)) stop("lengths of e and n differ.")
  N <- length(e)
  if(any(e == 0) | any(e == n)){
    e <- e + 0.5
    n <- n + 1
  }
  p <- e/n
  if(link == "log"){
    y <- log(p)
    v <- 1/e - 1/n
    back.trans <- function(x) exp(x)
  }
  if(link == "logit"){
    y <- log(p/(1 - p))
    v <- 1/e + 1/(n - e)
    back.trans <- function(x) exp(x)/(1 + exp(x))
  }
  if(link == "arcsine"){
    y <- asin(sqrt(p))
    v <- 1/(4 * n)
    back.trans <- function(x) (sin(x))^2
  }
  if(link == "double.arcsine"){
    y <- asin(sqrt(e/(n + 1))) + asin(sqrt((e + 1)/(n + 1)))
    v <- 1/(n + 0.5)
    back.trans <- function(x, pooled.n){
      if(x > asin(sqrt(1/(pooled.n + 1))) &
        x < (asin(sqrt(pooled.n/(pooled.n + 1))) + asin(1))){
        output <- 0.5 * (1 - sign(cos(x)) * sqrt(
          1 - (sin(x) + (sin(x) - 1/sin(x))/pooled.n)^2))
      }else{
        output <- (sin(x/2))^2
      }
      return(output)
    }
    pooled.n.harmonic <- 1/mean(1/n)
    pooled.n.geometric <- exp(mean(log(n)))
    pooled.n.arithmetic <- mean(n)
  }
  back.trans <- Vectorize(back.trans)
  rslt <- rma(yi = y, vi = v, method = method)
  mu.est <- as.numeric(rslt$beta)
  mu.se <- rslt$se
  mu.ci <- c(mu.est - qnorm(1 - alpha/2) * mu.se,
    mu.est + qnorm(1 - alpha/2) * mu.se)
```

```
tau.est <- sqrt(rslt$tau2)
if(link != "double.arcsine"){
  prop.c.est <- back.trans(mu.est)
  prop.c.ci <- back.trans(mu.ci)
}
if(link == "double.arcsine"){
  pooled.n.invvar <- 1/mu.se^2
  prop.c.est <- c(
    "harmonic" = back.trans(mu.est, pooled.n.harmonic),
    "geometric" = back.trans(mu.est, pooled.n.geometric),
    "arithmetic" = back.trans(mu.est, pooled.n.arithmetic),
    "invvar" = back.trans(mu.est, pooled.n.invvar))
  prop.c.ci <- rbind(
    "harmonic" = back.trans(mu.ci, pooled.n.harmonic),
    "geometric" = back.trans(mu.ci, pooled.n.geometric),
    "arithmetic" = back.trans(mu.ci, pooled.n.arithmetic),
    "invvar" = back.trans(mu.ci, pooled.n.invvar))
}
out <- list(prop.c.est = prop.c.est, prop.c.ci = prop.c.ci)
if(pop.avg){
  set.seed(seed)
  stdnorms <- rnorm(int.approx)
  if(link != "double.arcsine"){
    prop.m.est <- mean(back.trans(mu.est + tau.est * stdnorms))
    mu.est.b <- prop.m.est.b <- rep(NA, b.iter)
  }
  if(link == "double.arcsine"){
    prop.m.est <- c(
      "harmonic" = mean(back.trans(mu.est + tau.est * stdnorms,
        pooled.n.harmonic)),
      "geometric" = mean(back.trans(mu.est + tau.est * stdnorms,
        pooled.n.geometric)),
      "arithmetic" = mean(back.trans(mu.est + tau.est * stdnorms,
        pooled.n.arithmetic)),
      "invvar" = mean(back.trans(mu.est + tau.est * stdnorms,
        pooled.n.invvar)))
    mu.est.b <- rep(NA, b.iter)
    prop.m.est.b <- matrix(NA, b.iter, 4)
  }
  set.seed(seed)
  b.idx <- 0
  b.w <- b.e <- 0
  while(b.idx < b.iter){
    idx.temp <- sample(N, replace = TRUE)
    y.temp <- y[idx.temp]
    v.temp <- v[idx.temp]
    suppressMessages(out.W.E <- tryCatch.W.E(
      rslt.temp <- rma(yi = y.temp, vi = v.temp, method = method)))
    if(is.null(out.W.E$warning) & !inherits(out.W.E$value, "error")){
      b.idx <- b.idx + 1
      mu.est.b[b.idx] <- as.numeric(rslt.temp$beta)
      tau.est.temp <- sqrt(rslt.temp$tau2)
      if(link != "double.arcsine"){
        prop.m.est.b[b.idx] <- mean(back.trans(mu.est.b[b.idx] +
          tau.est.temp * stdnorms))
      }
      if(link == "double.arcsine"){
        prop.m.est.b[b.idx,1] <- mean(back.trans(mu.est.b[b.idx] +
          tau.est.temp * stdnorms, pooled.n.harmonic))
        prop.m.est.b[b.idx,2] <- mean(back.trans(mu.est.b[b.idx] +
          tau.est.temp * stdnorms, pooled.n.geometric))
        prop.m.est.b[b.idx,3] <- mean(back.trans(mu.est.b[b.idx] +
          tau.est.temp * stdnorms, pooled.n.arithmetic))
```

```
          prop.m.est.b[b.idx,4] <- mean(back.trans(mu.est.b[b.idx] +
            tau.est.temp * stdnorms, pooled.n.invvar))
        }
      }
      if(!is.null(out.W.E$warning)){
        b.w <- b.w + 1
      }
      if(inherits(out.W.E$value, "error")){
        b.e <- b.e + 1
      }
    }
    mu.ci.b <- quantile(mu.est.b, probs = c(alpha/2, 1 - alpha/2))
    if(link != "double.arcsine"){
      prop.c.ci.b <- back.trans(mu.ci.b)
      prop.m.ci.b <- quantile(prop.m.est.b,
        probs = c(alpha/2, 1 - alpha/2))
    }
    if(link == "double.arcsine"){
      prop.c.ci.b <- rbind(
        "harmonic" = back.trans(mu.ci.b, pooled.n.harmonic),
        "geometric" = back.trans(mu.ci.b, pooled.n.geometric),
        "arithmetic" = back.trans(mu.ci.b, pooled.n.arithmetic),
        "invvar" = back.trans(mu.ci.b, pooled.n.invvar))
      prop.m.ci.b <- rbind(
        "harmonic" = quantile(prop.m.est.b[,1],
          probs = c(alpha/2, 1 - alpha/2)),
        "geometric" = quantile(prop.m.est.b[,2],
          probs = c(alpha/2, 1 - alpha/2)),
        "arithmetic" = quantile(prop.m.est.b[,3],
          probs = c(alpha/2, 1 - alpha/2)),
        "invvar" = quantile(prop.m.est.b[,4],
          probs = c(alpha/2, 1 - alpha/2)))
    }
    out <- c(out, list(prop.c.ci.b = prop.c.ci.b,
      prop.m.est = prop.m.est, prop.m.ci.b = prop.m.ci.b))
    if(link == "double.arcsine"){
      out <- c(out, list(pooled.n =
        c("harmonic" = pooled.n.harmonic,
        "geometric" = pooled.n.geometric,
        "arithmetic" = pooled.n.arithmetic,
        "invvar" = pooled.n.invvar)))
    }
    out <- c(out, list(b.w.e =
      c("bootstrap warnings" = b.w, "bootstrap errors" = b.e)))
  }
  return(out)
}
```

```
##############
## Analysis for the chorioamnionitis data
##############

quantile(chor$n, probs = c(0, 0.25, 0.5, 0.75, 1), type = 3)
#      0%      25%      50%      75%     100%
#     102      916     5158   242715 10458616
quantile(chor$e/chor$n, probs = c(0, 0.25, 0.5, 0.75, 1),
  type = 3)
#         0%         25%         50%         75%        100%
#0.006060606 0.010333107 0.021779714 0.066110153 0.196078431


## GLMMs

# log link
begin <- Sys.time()
rslt.chor.glmm.log <-
  maprop.glmm(e, n, data = chor, link = "log",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 1.144098 mins
rslt.chor.glmm.log
#$prop.c.est
#[1] 0.03265205
#
#$prop.c.ci
#[1] 0.02110436 0.05051830
#
#$prop.c.ci.b
#      2.5%      97.5%
#0.02121845 0.04967762
#
#$prop.m.est
#[1] 0.05320085
#
#$prop.m.ci.b
#      2.5%      97.5%
#0.03058273 0.08307413
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                   0

# logit link
begin <- Sys.time()
rslt.chor.glmm.logit <-
  maprop.glmm(e, n, data = chor, link = "logit",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 1.550589 mins
rslt.chor.glmm.logit
#$prop.c.est
#[1] 0.03325553
#
#$prop.c.ci
#[1] 0.02125086 0.05168358
#
#$prop.c.ci.b
#      2.5%      97.5%
#0.02142075 0.05113936
```

```
#
#$prop.m.est
#[1] 0.05167813
#
#$prop.m.ci.b
#      2.5%      97.5%
#0.03052302 0.07771703
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# probit link
begin <- Sys.time()
rslt.chor.glmm.probit <-
  maprop.glmm(e, n, data = chor, link = "probit",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 1.287074 mins
rslt.chor.glmm.probit
#$prop.c.est
#[1] 0.03542661
#
#$prop.c.ci
#[1] 0.02207435 0.05478180
#
#$prop.c.ci.b
#      2.5%      97.5%
#0.02272322 0.05525192
#
#$prop.m.est
#[1] 0.05113666
#
#$prop.m.ci.b
#      2.5%      97.5%
#0.03072841 0.07617872
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                30                  0
30/(1000 + 30) # warning rate during bootstrapping

# cauchit link
begin <- Sys.time()
rslt.chor.glmm.cauchit <-
  maprop.glmm(e, n, data = chor, link = "cauchit",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 2.925113 mins
rslt.chor.glmm.cauchit
#$prop.c.est
#[1] 0.02405381
#
#$prop.c.ci
#[1] 0.01795415 0.03640323
#
#$prop.c.ci.b
#      2.5%      97.5%
#0.01694010 0.03548455
#
#$prop.m.est
```

```
#[1] 0.1134418
#
#$prop.m.ci.b
#      2.5%      97.5%
#0.05964213 0.16100335
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                305                 4
(305 + 4)/(1000 + 305 + 4) # warning and error rate during bootstrapping

# cloglog link
begin <- Sys.time()
rslt.chor.glmm.cloglog <-
  maprop.glmm(e, n, data = chor, link = "cloglog",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 1.192675 mins
rslt.chor.glmm.cloglog
#$prop.c.est
#[1] 0.03295156
#
#$prop.c.ci
#[1] 0.02117721 0.05109929
#
#$prop.c.ci.b
#      2.5%      97.5%
#0.02131884 0.05047530
#
#$prop.m.est
#[1] 0.05220357
#
#$prop.m.ci.b
#      2.5%      97.5%
#0.03057284 0.07955954
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                   0


## Two-step methods (via the ML estimation)

# log transformation
begin <- Sys.time()
rslt.chor.twostep.log <-
  maprop.twostep(e, n, data = chor, link = "log",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 19.57314 secs
rslt.chor.twostep.log
#$prop.c.est
#[1] 0.03391666
#
#$prop.c.ci
#[1] 0.02206290 0.05213908
#
#$prop.c.ci.b
#      2.5%      97.5%
#0.02203085 0.05208696
#
```

```
#$prop.m.est
#[1] 0.05406946
#
#$prop.m.ci.b
#      2.5%      97.5%
#0.03158354 0.08382169
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# logit transformation
begin <- Sys.time()
rslt.chor.twostep.logit <-
  maprop.twostep(e, n, data = chor, link = "logit",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 14.72922 secs
rslt.chor.twostep.logit
#$prop.c.est
#[1] 0.0344766
#
#$prop.c.ci
#[1] 0.02218636 0.05320464
#
#$prop.c.ci.b
#      2.5%      97.5%
#0.02217594 0.05311491
#
#$prop.m.est
#[1] 0.05253289
#
#$prop.m.ci.b
#      2.5%      97.5%
#0.03135427 0.07820211
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# arcsine transformation
begin <- Sys.time()
rslt.chor.twostep.arcsine <-
  maprop.twostep(e, n, data = chor, link = "arcsine",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 13.58505 secs
rslt.chor.twostep.arcsine
#$prop.c.est
#[1] 0.04128272
#
#$prop.c.ci
#[1] 0.02485482 0.06163975
#
#$prop.c.ci.b
#      2.5%      97.5%
#0.02558480 0.06279374
#
#$prop.m.est
#[1] 0.05160069
#
```

```
#$prop.m.ci.b
#      2.5%      97.5%
#0.03135427 0.07550149
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# double-arcsine transformation
begin <- Sys.time()
rslt.chor.twostep.double.arcsine <-
  maprop.twostep(e, n, data = chor, link = "double.arcsine",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 2.0813 mins
rslt.chor.twostep.double.arcsine
#$prop.c.est
#  harmonic  geometric arithmetic     invvar
#0.04143204 0.04209214 0.04211813 0.04112789
#
#$prop.c.ci
#                 [,1]       [,2]
#harmonic   0.02497610 0.06172339
#geometric  0.02565785 0.06235522
#arithmetic 0.02568478 0.06238007
#invvar     0.02466325 0.06143165
#
#$prop.c.ci.b
#                 2.5%      97.5%
#harmonic   0.02584843 0.06295809
#geometric  0.02652910 0.06358818
#arithmetic 0.02655597 0.06361296
#invvar     0.02553597 0.06266713
#
#$prop.m.est
#  harmonic  geometric arithmetic     invvar
#0.05153716 0.05215018 0.05217487 0.05125990
#
#$prop.m.ci.b
#                 2.5%      97.5%
#harmonic   0.03148616 0.07539564
#geometric  0.03210718 0.07598335
#arithmetic 0.03213252 0.07600684
#invvar     0.03119470 0.07512818
#
#$pooled.n
#    harmonic    geometric   arithmetic       invvar
#    663.9025   17358.4370 1207998.0000     459.1022
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0


## Two-step methods (via the REML estimation)

# log transformation
begin <- Sys.time()
rslt.chor.twostep.log.reml <-
  maprop.twostep(e, n, data = chor, link = "log", method = "REML",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
```

```
end - begin
#Time difference of 13.8282 secs
rslt.chor.twostep.log.reml
#$prop.c.est
#[1] 0.03382256
#
#$prop.c.ci
#[1] 0.02175152 0.05259242
#
#$prop.c.ci.b
#      2.5%      97.5%
#0.02198691 0.05184071
#
#$prop.m.est
#[1] 0.05534902
#
#$prop.m.ci.b
#      2.5%      97.5%
#0.03202686 0.08645067
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# logit transformation
begin <- Sys.time()
rslt.chor.twostep.logit.reml <-
  maprop.twostep(e, n, data = chor, link = "logit", method = "REML",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 17.81827 secs
rslt.chor.twostep.logit.reml
#$prop.c.est
#[1] 0.034387
#
#$prop.c.ci
#[1] 0.02186921 0.05367669
#
#$prop.c.ci.b
#      2.5%      97.5%
#0.02214696 0.05296746
#
#$prop.m.est
#[1] 0.05348826
#
#$prop.m.ci.b
#      2.5%      97.5%
#0.03181462 0.07987447
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# arcsine transformation
begin <- Sys.time()
rslt.chor.twostep.arcsine.reml <-
  maprop.twostep(e, n, data = chor, link = "arcsine", method = "REML",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 19.40291 secs
rslt.chor.twostep.arcsine.reml
```

```
#$prop.c.est
#[1] 0.04129417
#
#$prop.c.ci
#[1] 0.02448924 0.06223556
#
#$prop.c.ci.b
#      2.5%      97.5%
#0.02554905 0.06281879
#
#$prop.m.est
#[1] 0.05215816
#
#$prop.m.ci.b
#      2.5%      97.5%
#0.03165166 0.07618957
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                 0

# double-arcsine transformation
begin <- Sys.time()
rslt.chor.twostep.double.arcsine.reml <-
  maprop.twostep(e, n, data = chor, link = "double.arcsine", method = "REML",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 2.069883 mins
rslt.chor.twostep.double.arcsine.reml
#$prop.c.est
#  harmonic  geometric arithmetic     invvar
#0.04144734 0.04210741 0.04213341 0.04109132
#
#$prop.c.ci
#                 [,1]       [,2]
#harmonic   0.02461220 0.06231990
#geometric  0.02529440 0.06295089
#arithmetic 0.02532134 0.06297571
#invvar     0.02424589 0.06197881
#
#$prop.c.ci.b
#                 2.5%      97.5%
#harmonic   0.02581856 0.06298169
#geometric  0.02649927 0.06361175
#arithmetic 0.02652615 0.06363653
#invvar     0.02545286 0.06264109
#
#$prop.m.est
#  harmonic  geometric arithmetic     invvar
#0.05208767 0.05269704 0.05272161 0.05176595
#
#$prop.m.ci.b
#                 2.5%      97.5%
#harmonic   0.03173402 0.07607069
#geometric  0.03237875 0.07665463
#arithmetic 0.03240477 0.07667801
#invvar     0.03139602 0.07576066
#
#$pooled.n
#    harmonic    geometric   arithmetic       invvar
#    663.9025   17358.4370 1207998.0000     436.1000
#
```

```
#$b.w.e
#bootstrap warnings   bootstrap errors
#                  0                   0
```

```
##############
## Analysis for the depression data
##############

quantile(beck17$n, probs = c(0, 0.25, 0.5, 0.75, 1), type = 3)
#  0%  25%  50%  75% 100%
# 128  164  202  352  668
quantile(beck17$e/beck17$n, probs = c(0, 0.25, 0.5, 0.75, 1),
  type = 3)
#       0%       25%       50%       75%      100%
#0.1016949 0.1171875 0.1188119 0.2187500 0.5792683


## GLMMs

# log link
begin <- Sys.time()
rslt.beck17.glmm.log <-
  maprop.glmm(e, n, data = beck17, link = "log",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 1.216138 mins
rslt.beck17.glmm.log
#$prop.c.est
#[1] 0.2012347
#
#$prop.c.ci
#[1] 0.1193295 0.3393576
#
#$prop.c.ci.b
#     2.5%     97.5%
#0.1259438 0.3505807
#
#$prop.m.est
#[1] 0.2461105
#
#$prop.m.ci.b
#     2.5%     97.5%
#0.1303579 0.4034569
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 2                  0
2/(1000 + 2) # warning rate during bootstrapping

# logit link
begin <- Sys.time()
rslt.beck17.glmm.logit <-
  maprop.glmm(e, n, data = beck17, link = "logit",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 1.517496 mins
rslt.beck17.glmm.logit
#$prop.c.est
#[1] 0.2152933
#
#$prop.c.ci
#[1] 0.1171091 0.3620398
#
#$prop.c.ci.b
#     2.5%     97.5%
```

```
#0.1265425 0.3701205
#
#$prop.m.est
#[1] 0.2470891
#
#$prop.m.ci.b
#     2.5%     97.5%
#0.1303986 0.3841899
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# probit link
begin <- Sys.time()
rslt.beck17.glmm.probit <-
  maprop.glmm(e, n, data = beck17, link = "probit",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 1.637059 mins
rslt.beck17.glmm.probit
#$prop.c.est
#[1] 0.2215186
#
#$prop.c.ci
#[1] 0.1166794 0.3660259
#
#$prop.c.ci.b
#     2.5%     97.5%
#0.1271957 0.3733982
#
#$prop.m.est
#[1] 0.2481958
#
#$prop.m.ci.b
#     2.5%     97.5%
#0.1303282 0.3838476
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# cauchit link
begin <- Sys.time()
rslt.beck17.glmm.cauchit <-
  maprop.glmm(e, n, data = beck17, link = "cauchit",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 1.516028 mins
rslt.beck17.glmm.cauchit
#$prop.c.est
#[1] 0.180481
#
#$prop.c.ci
#[1] 0.1175418 0.3381954
#
#$prop.c.ci.b
#     2.5%     97.5%
#0.1193129 0.3221480
#
#$prop.m.est
```

```
#[1] 0.2395022
#
#$prop.m.ci.b
#     2.5%     97.5%
#0.1267812 0.3732827
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#               247                  0
247/(1000 + 247) # warning rate during bootstrapping

# cloglog link
begin <- Sys.time()
rslt.beck17.glmm.cloglog <-
  maprop.glmm(e, n, data = beck17, link = "cloglog",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 1.090536 mins
rslt.beck17.glmm.cloglog
#$prop.c.est
#[1] 0.2081778
#
#$prop.c.ci
#[1] 0.1183422 0.3511680
#
#$prop.c.ci.b
#     2.5%     97.5%
#0.1262498 0.3616989
#
#$prop.m.est
#[1] 0.2449618
#
#$prop.m.ci.b
#     2.5%     97.5%
#0.1303796 0.3857308
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0


## Two-step methods (via the ML estimation)

# log transformation
begin <- Sys.time()
rslt.beck17.twostep.log <-
  maprop.twostep(e, n, data = beck17, link = "log",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 13.27123 secs
rslt.beck17.twostep.log
#$prop.c.est
#[1] 0.2038565
#
#$prop.c.ci
#[1] 0.1214481 0.3421832
#
#$prop.c.ci.b
#     2.5%     97.5%
#0.1275784 0.3534538
#
```

```
#$prop.m.est
#[1] 0.2483628
#
#$prop.m.ci.b
#     2.5%     97.5%
#0.1321933 0.4061459
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# logit transformation
begin <- Sys.time()
rslt.beck17.twostep.logit <-
  maprop.twostep(e, n, data = beck17, link = "logit",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 14.48288 secs
rslt.beck17.twostep.logit
#$prop.c.est
#[1] 0.2169585
#
#$prop.c.ci
#[1] 0.1187267 0.3629885
#
#$prop.c.ci.b
#     2.5%     97.5%
#0.1277700 0.3713804
#
#$prop.m.est
#[1] 0.248264
#
#$prop.m.ci.b
#     2.5%     97.5%
#0.1317567 0.3850360
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# arcsine transformation
begin <- Sys.time()
rslt.beck17.twostep.arcsine <-
  maprop.twostep(e, n, data = beck17, link = "arcsine",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 13.25846 secs
rslt.beck17.twostep.arcsine
#$prop.c.est
#[1] 0.2309946
#
#$prop.c.ci
#[1] 0.1157969 0.3715545
#
#$prop.c.ci.b
#     2.5%     97.5%
#0.1282288 0.3776901
#
#$prop.m.est
#[1] 0.2501637
#
```

```
#$prop.m.ci.b
#     2.5%     97.5%
#0.1304489 0.3850995
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                 0

# double-arcsine transformation
begin <- Sys.time()
rslt.beck17.twostep.double.arcsine <-
  maprop.twostep(e, n, data = beck17, link = "double.arcsine",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 2.19528 mins
rslt.beck17.twostep.double.arcsine
#$prop.c.est
#  harmonic  geometric arithmetic     invvar
# 0.2312305  0.2313837  0.2315301  0.2259703
#
#$prop.c.ci
#                [,1]      [,2]
#harmonic   0.1159119 0.3715353
#geometric  0.1161298 0.3716086
#arithmetic 0.1163383 0.3716787
#invvar     0.1085030 0.3690092
#
#$prop.c.ci.b
#                2.5%     97.5%
#harmonic   0.1285004 0.3778245
#geometric  0.1287114 0.3778943
#arithmetic 0.1289131 0.3779609
#invvar     0.1213135 0.3754218
#
#$prop.m.est
#  harmonic  geometric arithmetic     invvar
# 0.2501461  0.2502849  0.2504181  0.2454873
#
#$prop.m.ci.b
#                2.5%     97.5%
#harmonic   0.1307096 0.3851173
#geometric  0.1309193 0.3851827
#arithmetic 0.1311197 0.3852452
#invvar     0.1235832 0.3828780
#
#$pooled.n
#  harmonic  geometric arithmetic     invvar
# 230.56235  265.84057  311.33333   41.00524
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                 0


## Two-step methods (via the REML estimation)

# log transformation
begin <- Sys.time()
rslt.beck17.twostep.log.reml <-
  maprop.twostep(e, n, data = beck17, link = "log", method = "REML",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
```

```
end - begin
#Time difference of 13.51329 secs
rslt.beck17.twostep.log.reml
#$prop.c.est
#[1] 0.2030759
#
#$prop.c.ci
#[1] 0.1152479 0.3578355
#
#$prop.c.ci.b
#     2.5%     97.5%
#0.1271257 0.3524065
#
#$prop.m.est
#[1] 0.2576287
#
#$prop.m.ci.b
#     2.5%     97.5%
#0.1326497 0.4262765
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 2                  0
2/(1000 + 2) # warning rate during bootstrapping

# logit transformation
begin <- Sys.time()
rslt.beck17.twostep.logit.reml <-
  maprop.twostep(e, n, data = beck17, link = "logit", method = "REML",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 20.75458 secs
rslt.beck17.twostep.logit.reml
#$prop.c.est
#[1] 0.2165638
#
#$prop.c.ci
#[1] 0.1114949 0.3784700
#
#$prop.c.ci.b
#     2.5%     97.5%
#0.1274018 0.3706589
#
#$prop.m.est
#[1] 0.2530671
#
#$prop.m.ci.b
#     2.5%     97.5%
#0.1321655 0.3868350
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# arcsine transformation
begin <- Sys.time()
rslt.beck17.twostep.arcsine.reml <-
  maprop.twostep(e, n, data = beck17, link = "arcsine", method = "REML",
  alpha = 0.05, int.approx = 10000, b.iter = 1000, seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 17.25383 secs
```

```
rslt.beck17.twostep.arcsine.reml
#$prop.c.est
#[1] 0.2309858
#
#$prop.c.ci
#[1] 0.1064205 0.3860094
#
#$prop.c.ci.b
#     2.5%     97.5%
#0.1280987 0.3777034
#
#$prop.m.est
#[1] 0.2538862
#
#$prop.m.ci.b
#     2.5%     97.5%
#0.1307509 0.3867119
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                 0                  0

# double-arcsine transformation
begin <- Sys.time()
rslt.beck17.twostep.double.arcsine.reml <-
  maprop.twostep(e, n, data = beck17, link = "double.arcsine",
  method = "REML", alpha = 0.05, int.approx = 10000, b.iter = 1000,
  seed = 1234)
end <- Sys.time()
end - begin
#Time difference of 2.754157 mins
rslt.beck17.twostep.double.arcsine.reml
#$prop.c.est
#  harmonic  geometric arithmetic     invvar
# 0.2312235  0.2313766  0.2315230  0.2246872
#
#$prop.c.ci
#                 [,1]      [,2]
#harmonic   0.10649896 0.3859556
#geometric  0.10672212 0.3860207
#arithmetic 0.10693555 0.3860829
#invvar     0.09712007 0.3831661
#
#$prop.c.ci.b
#                2.5%     97.5%
#harmonic   0.1283217 0.3778412
#geometric  0.1285328 0.3779110
#arithmetic 0.1287347 0.3779776
#invvar     0.1194055 0.3748537
#
#$prop.m.est
#  harmonic  geometric arithmetic     invvar
# 0.2538320  0.2539670  0.2540964  0.2482846
#
#$prop.m.ci.b
#                2.5%     97.5%
#harmonic   0.1310174 0.3870820
#geometric  0.1312268 0.3871439
#arithmetic 0.1314271 0.3872034
#invvar     0.1221988 0.3845252
#
#$pooled.n
#  harmonic  geometric arithmetic     invvar
```

```
# 230.56235  265.84057  311.33333   34.09037
#
#$b.w.e
#bootstrap warnings   bootstrap errors
#                  0                   0
```

```
##############
## Generating forest plots for the two meta-analyses
##############

## Specify your working directory
setwd("...")
library("metafor")

pdf(file = "Figure.pdf", width = 6.4 * 2, height = 5)
par(mfrow = c(1, 2))

##(A) Meta-analysis on chorioamnionitis
par(mar = c(3, 3, 1, 1) + 0.1)
fp.chor <- rma(xi = e, ni = n, data = chor, measure = "PLO",
  method = "ML")
est.chor <- cbind(fp.chor$yi,
  fp.chor$yi - qnorm(0.975) * sqrt(fp.chor$vi),
  fp.chor$yi + qnorm(0.975) * sqrt(fp.chor$vi))
est.chor <- exp(est.chor)/(1 + exp(est.chor))*100
est.chor <- format(round(est.chor, digits = 2))
est.chor <- gsub(" ", "", est.chor)
pts.chor <- est.chor[,1]
ci.chor <- paste0("(", est.chor[,2], ", ", est.chor[,3], ")")
xtik <- c(0.1, 1, 5, 15, 30)
forest(fp.chor, addfit = FALSE, annotate = FALSE,
  atransf = transf.ilogit,
  xlim = c(-12.8, 4.8), xlab = "", col.axis = "white",
  at = transf.logit(xtik/100), slab = NA, refline = NA,
  ilab = cbind(1:length(chor$e),
  gsub(" ", "", format(cbind(chor$e, chor$n), big.mark = ",",
  scientific = FALSE)),
  pts.chor, ci.chor),
  ilab.xpos = c(-11.4, -9.5, -7.1, 2, 2),
  ilab.pos = c(2, 2, 2, 2, 4), cex = 0.75)
text(c(-11.4, -9.5, -7.1, 2, 2), 23,
  c("Study", "Event", "Sample size", "Proportion, %", "(95% CI)"),
  cex = 0.75, pos = c(2, 2, 2, 2, 4))
mtext(text = xtik, side = 1, line = 0.7,
  at = transf.logit(xtik/100), outer = FALSE, cex = 0.75)
mtext(text = "Proportion, % (logit scale)", side = 1,
  line = 1.6, outer = FALSE, cex = 0.75)
mtext(text =
  "(A) Meta\255analysis on chorioamnionitis",
  side = 3, line = 0, outer = FALSE, cex = 0.75)

##(B) Meta-analysis on depression or depressive symptoms
par(mar = c(3, 3, 0, 1) + 0.1)
fp.beck17 <- rma(xi = e, ni = n, data = beck17, measure = "PLO",
  method = "ML")
est.beck17 <- cbind(fp.beck17$yi,
  fp.beck17$yi - qnorm(0.975) * sqrt(fp.beck17$vi),
  fp.beck17$yi + qnorm(0.975) * sqrt(fp.beck17$vi))
est.beck17 <- exp(est.beck17)/(1 + exp(est.beck17))*100
est.beck17 <- format(round(est.beck17, digits = 2))
est.beck17 <- gsub(" ", "", est.beck17)
pts.beck17 <- est.beck17[,1]
ci.beck17 <- paste0("(", est.beck17[,2], ", ", est.beck17[,3], ")")
xtik <- c(5, 10, 30, 50, 70)
forest(fp.beck17, addfit = FALSE, annotate = FALSE,
  atransf = transf.ilogit,
  xlim = c(-6.2, 4), xlab = "", col.axis = "white",
  at = transf.logit(xtik/100), slab = NA, refline = NA,
  ilab = cbind(1:length(beck17$e),
```

```
  gsub(" ", "", format(cbind(beck17$e, beck17$n), big.mark = ",",
  scientific = FALSE)),
  pts.beck17, ci.beck17),
  ilab.xpos = c(-5.4, -4.4, -3, 2.3, 2.3),
  ilab.pos = c(2, 2, 2, 2, 4), cex = 0.75)
text(c(-5.4, -4.4, -3, 2.3, 2.3), 8,
  c("Study", "Event", "Sample size", "Proportion, %", "(95% CI)"),
  cex = 0.75, pos = c(2, 2, 2, 2, 4))
mtext(text = xtik, side = 1, line = 0.7,
  at = transf.logit(xtik/100), outer = FALSE, cex = 0.75)
mtext(text = "Proportion, % (logit scale)", side = 1,
  line = 1.6, outer = FALSE, cex = 0.75)
mtext(text =
  "(B) Meta\255analysis on depression or depressive symptoms",
  side = 3, line = -1, outer = FALSE, cex = 0.75)

dev.off()
```

```
##############
## Simulation studies
##############

## Specify your working directory
setwd("...")

Ns <- c(5, 30)
ps <- c(0.01, 0.2)
taus <- c(0.1, 0.5)
ns <- list(seq(100, 500, 100), seq(1000, 5000, 1000))

# rough I^2 when p = 0.01, n = 100--500, tau = 0.1
0.1^2/(1/(300*0.01) + 1/(300*(1 - 0.01)) + 0.1^2)
# rough I^2 when p = 0.01, n = 100--500, tau = 0.5
0.5^2/(1/(300*0.01) + 1/(300*(1 - 0.01)) + 0.5^2)
# rough I^2 when p = 0.01, n = 1000--5000, tau = 0.1
0.1^2/(1/(3000*0.01) + 1/(3000*(1 - 0.01)) + 0.1^2)
# rough I^2 when p = 0.01, n = 1000--5000, tau = 0.5
0.5^2/(1/(3000*0.01) + 1/(3000*(1 - 0.01)) + 0.5^2)
# rough I^2 when p = 0.2, n = 100--500, tau = 0.1
0.1^2/(1/(300*0.2) + 1/(300*(1 - 0.2)) + 0.1^2)
# rough I^2 when p = 0.2, n = 100--500, tau = 0.5
0.5^2/(1/(300*0.2) + 1/(300*(1 - 0.2)) + 0.5^2)
# rough I^2 when p = 0.2, n = 1000--5000, tau = 0.1
0.1^2/(1/(3000*0.2) + 1/(3000*(1 - 0.2)) + 0.1^2)
# rough I^2 when p = 0.2, n = 1000--5000, tau = 0.5
0.5^2/(1/(3000*0.2) + 1/(3000*(1 - 0.2)) + 0.5^2)

begin <- Sys.time()
methods <- c("glmm.log", "glmm.logit", "glmm.probit", "glmm.cauchit",
  "glmm.cloglog", "twostep.log", "twostep.logit", "twostep.arcsine",
  "twostep.double.arcsine.harmonic", "twostep.double.arcsine.geometric",
  "twostep.double.arcsine.arithmetic", "twostep.double.arcsine.invvar")
N.iter <- 1000
true.iters <- settings <- NULL
set.seed(1234)
for(N in Ns){
  for(p in ps){
    for(tau in taus){
      for(idx.n in 1:2){
        ni <- ns[[idx.n]]
        ni <- rep(ni, N/length(ni))
        mu <- log(p/(1 - p))
        idx.iter <- true.iter <- 0
        n.w.e <- 0
        prop.est <- ci.cover <- NULL
        while(idx.iter < N.iter){
          true.iter <- true.iter + 1
          mui <- rnorm(N, mean = mu, sd = tau)
          pi <- exp(mui)/(1 + exp(mui))
          ei <- rbinom(N, size = ni, prob = pi)
          suppressMessages(check <- tryCatch.W.E(glmm.log <-
            maprop.glmm(ei, ni, link = "log", pop.avg = FALSE)))
          if(!is.null(check$warning) | inherits(check$value, "error")) next
          suppressMessages(check <- tryCatch.W.E(glmm.logit <-
            maprop.glmm(ei, ni, link = "logit", pop.avg = FALSE)))
          if(!is.null(check$warning) | inherits(check$value, "error")) next
          suppressMessages(check <- tryCatch.W.E(glmm.probit <-
            maprop.glmm(ei, ni, link = "probit", pop.avg = FALSE)))
          if(!is.null(check$warning) | inherits(check$value, "error")) next
          suppressMessages(check <- tryCatch.W.E(glmm.cauchit <-
            maprop.glmm(ei, ni, link = "cauchit", pop.avg = FALSE)))
```

```
          if(!is.null(check$warning) | inherits(check$value, "error")) next
          suppressMessages(check <- tryCatch.W.E(glmm.cloglog <-
            maprop.glmm(ei, ni, link = "cloglog", pop.avg = FALSE)))
          if(!is.null(check$warning) | inherits(check$value, "error")) next
          ei <- rbinom(N, size = ni, prob = pi)
          suppressMessages(check <- tryCatch.W.E(twostep.log <-
            maprop.twostep(ei, ni, link = "log", pop.avg = FALSE)))
          if(!is.null(check$warning) | inherits(check$value, "error")) next
          suppressMessages(check <- tryCatch.W.E(twostep.logit <-
            maprop.twostep(ei, ni, link = "logit", pop.avg = FALSE)))
          if(!is.null(check$warning) | inherits(check$value, "error")) next
          suppressMessages(check <- tryCatch.W.E(twostep.arcsine <-
            maprop.twostep(ei, ni, link = "arcsine", pop.avg = FALSE)))
          if(!is.null(check$warning) | inherits(check$value, "error")) next
          suppressMessages(check <- tryCatch.W.E(twostep.double.arcsine <-
            maprop.twostep(ei, ni, link = "double.arcsine", pop.avg = FALSE)))
          if(!is.null(check$warning) | inherits(check$value, "error")) next
          idx.iter <- idx.iter + 1
          est.temp <- c(glmm.log$prop.c.est, glmm.logit$prop.c.est,
            glmm.probit$prop.c.est, glmm.cauchit$prop.c.est,
            glmm.cloglog$prop.c.est, twostep.log$prop.c.est,
            twostep.logit$prop.c.est, twostep.arcsine$prop.c.est,
            twostep.double.arcsine$prop.c.est)
          ci.temp <- rbind(glmm.log$prop.c.ci, glmm.logit$prop.c.ci,
            glmm.probit$prop.c.ci, glmm.cauchit$prop.c.ci,
            glmm.cloglog$prop.c.ci, twostep.log$prop.c.ci,
            twostep.logit$prop.c.ci, twostep.arcsine$prop.c.ci,
            twostep.double.arcsine$prop.c.ci)
          cover.temp <- as.numeric(ci.temp[,1] < p & ci.temp[,2] > p)
          prop.est <- rbind(prop.est, est.temp)
          ci.cover <- rbind(ci.cover, cover.temp)
          colnames(prop.est) <- colnames(ci.cover) <- methods
        }
        setting <- paste0("N=", N, " p=", p, " tau=", tau, " idx.n=", idx.n)
        write.csv(prop.est, paste0("prop.est ", setting, ".csv"),
          row.names = FALSE)
        write.csv(ci.cover, paste0("ci.cover ", setting, ".csv"),
          row.names = FALSE)
        true.iters <- c(true.iters, true.iter)
        settings <- c(settings, setting)
      }
    }
  }
}
end <- Sys.time()
end - begin
#Time difference of 1.728408 hours

# Number of iterations to obtain 1000 replicates without
#   any warnings or errors in each setting
cbind(settings, true.iters)
#      settings                      true.iters
# [1,] "N=5 p=0.01 tau=0.1 idx.n=1"  "1040"
# [2,] "N=5 p=0.01 tau=0.1 idx.n=2"  "1018"
# [3,] "N=5 p=0.01 tau=0.5 idx.n=1"  "1084"
# [4,] "N=5 p=0.01 tau=0.5 idx.n=2"  "1085"
# [5,] "N=5 p=0.2 tau=0.1 idx.n=1"   "1015"
# [6,] "N=5 p=0.2 tau=0.1 idx.n=2"   "1029"
# [7,] "N=5 p=0.2 tau=0.5 idx.n=1"   "1100"
# [8,] "N=5 p=0.2 tau=0.5 idx.n=2"   "1002"
# [9,] "N=30 p=0.01 tau=0.1 idx.n=1" "1071"
#[10,] "N=30 p=0.01 tau=0.1 idx.n=2" "1039"
#[11,] "N=30 p=0.01 tau=0.5 idx.n=1" "1133"
```

```
#[12,] "N=30 p=0.01 tau=0.5 idx.n=2" "1011"
#[13,] "N=30 p=0.2 tau=0.1 idx.n=1"  "1005"
#[14,] "N=30 p=0.2 tau=0.1 idx.n=2"  "1001"
#[15,] "N=30 p=0.2 tau=0.5 idx.n=1"  "1202"
#[16,] "N=30 p=0.2 tau=0.5 idx.n=2"  "1006"

## Summarize results
bias <- rmse <- cp <- settings <- NULL
for(N in Ns){
  for(p in ps){
    for(tau in taus){
      for(idx.n in 1:2){
        setting <- paste0("N=", N, " p=", p, " tau=", tau, " idx.n=", idx.n)
        prop.est <- read.csv(paste0("prop.est ", setting, ".csv"))
        ci.cover <- read.csv(paste0("ci.cover ", setting, ".csv"))
        bias.temp <- colMeans(prop.est) - p
        rmse.temp <- sqrt(colMeans((prop.est - p)^2))
        cp.temp <- colMeans(ci.cover)
        bias <- rbind(bias, bias.temp)
        rmse <- rbind(rmse, rmse.temp)
        cp <- rbind(cp, cp.temp)
        settings <- c(settings, setting)
      }
    }
  }
}
rownames(bias) <- rownames(rmse) <- rownames(cp) <- settings
bias
rmse
cp

bias <- round(bias*100, 2)
rmse <- round(rmse*100, 2)
cp <- round(cp*100, 0)

# eTable 2 in eAppendix E
rbind(cbind(bias[1,], rmse[1,], cp[1,], bias[3,], rmse[3,], cp[3,],
  bias[5,], rmse[5,], cp[5,], bias[7,], rmse[7,], cp[7,]),
  cbind(bias[2,], rmse[2,], cp[2,], bias[4,], rmse[4,], cp[4,],
  bias[6,], rmse[6,], cp[6,], bias[8,], rmse[8,], cp[8,]))

# Table 2 in the main content
rbind(cbind(bias[9,], rmse[9,], cp[9,], bias[11,], rmse[11,], cp[11,],
  bias[13,], rmse[13,], cp[13,], bias[15,], rmse[15,], cp[15,]),
  cbind(bias[10,], rmse[10,], cp[10,], bias[12,], rmse[12,], cp[12,],
  bias[14,], rmse[14,], cp[14,], bias[16,], rmse[16,], cp[16,]))
```