Economics: The Open-Access, Open-Assessment E-Journal

Stata .do files to produce all tables and figures in the paper:

W. Robert Reed (2015). A Monte Carlo Analysis of Alternative Meta-Analysis Estimators in the Presence of Publication Bias. *Economics: The Open-Access, Open-Assessment E-Journal*, 9 (2015-30): 1—40.
http://dx.doi.org/10.5018/economics-ejournal.ja.2015-30

Note #1: These .do files are included to make it possible for readers to confirm this study's findings, and investigate alternative DGP specifications. Many of the programs perform 1,000 simulations and take several hours to run. However, the running time can be decreased without substantially affecting the results by reducing the number of replications to 100.

Note #2: Some programs consist of two parts. Part A should be run first.

.do file for TABLE 1 – Part A

NOTE: The yellow-highlighted part of the code determines which type of publication bias is operative

```
clear
set more off
program drop _all
program define sample, rclass
version 13.1

syntax, alpha(real) studies(integer) select(real) obs(integer)

// Remove existing variables
drop _all

//We first create the matrix to store the results of each study
matrix A = J(`studies',1,0)
matrix B = J(`studies',1,0)
matrix C = J(`studies',1,0)

forvalues i = 1/`studies' {
        clear
        // STEP ONE: Create the data for each study and estimate an effect
        set obs `obs'
        generate x = rnormal()
        // Note that each "study" has the same number of observations (100)
        // but differ in the variance of their respective error terms. This
        // causes the estimate of the effect to be estimated with varying degrees
        // of precision
        // This is for random effects
        scalar lambda = 0.5+30*runiform()
        generate e = lambda*rnormal()
        scalar alpha = `alpha' + rnormal()
        generate y = 1 + alpha*x + e
```

```
        quietly regress y x
        scalar coef = _b[x]
        scalar secoef = _se[x]
        scalar tcoef = coef/secoef

        // First run this program once to get the pre-publication study sample data
        // To get post-publication study sample data, uncomment one of the two sections
        // below.

        /*if abs(tcoef) < 2 {
        scalar dummy = cond(runiform()<`select',1,.)
        scalar coef = dummy*coef
        scalar secoef = dummy*secoef
        scalar tcoef = dummy*tcoef
        }
        */

        /*if coef < 0 {
        scalar dummy = cond(runiform()<`select',1,.)
        scalar coef = dummy*coef
        scalar secoef = dummy*secoef
        scalar tcoef = dummy*tcoef
        }
        */

        matrix A[`i',1] = coef
        matrix B[`i',1] = secoef
        matrix C[`i',1] = tcoef
}

matrix bob = A,B,C
svmat bob
rename bob1 effect
rename bob2 seeffect
```

```
rename bob3 teffect
generate pet = (1/seeffect)

metareg effect , wsse(seeffect) mm
return scalar I2 = e(I2)

summ effect, detail
return scalar effectMED = r(p50)
return scalar effectMIN = r(min)
return scalar effectP5 = r(p5)
return scalar effectP95 = r(p95)
return scalar effectMAX = r(max)
return scalar N = r(N)

summ teffect, detail
return scalar teffectMED = r(p50)
return scalar teffectMIN = r(min)
return scalar teffectP5 = r(p5)
return scalar teffectP95 = r(p95)
return scalar teffectMAX = r(max)

summ pet, detail
return scalar petMED = r(p50)
return scalar petMIN = r(min)
return scalar petP5 = r(p5)
return scalar petP95 = r(p95)
return scalar petMAX = r(max)

end
```

.do file for TABLE 1 – Part B

```
// This program takes about 90 minutes to run on my laptop for each section
etime, start
drop _all
clear
set matsize 5000
graph drop _all
set more off
set seed 13
matrix RESULTS = J(4,6,0)
local studies = 1000
// Select is used to set the probability of being included in the data set
// when the study is subject to publication bias, either because (i) abs(tstat) < 2, or
// coef < 0.
local select = 0.10
local obs = 100
local alpha = 1

simulate effectMED = r(effectMED) effectMIN = r(effectMIN) effectP5 = r(effectP5) ///
        effectP95 = r(effectP95) effectMAX = r(effectMAX) ///
        teffectMED = r(teffectMED) teffectMIN = r(teffectMIN) teffectP5 = r(teffectP5)
///
        teffectP95 = r(teffectP95) teffectMAX = r(teffectMAX) ///
                petMED = r(petMED) petMIN = r(petMIN) petP5 = r(petP5) ///
        petP95 = r(petP95) petMAX = r(petMAX) N = r(N) I2 = r(I2), ///
    reps(10000): sample, alpha(`alpha') studies(`studies') select(`select') obs(`obs')

summ effectMED, meanonly
matrix RESULTS[1,1] = r(mean)
summ effectMIN, meanonly
matrix RESULTS[1,2] = r(mean)
summ effectP5, meanonly
matrix RESULTS[1,3] = r(mean)
summ effectP95, meanonly
```

```
matrix RESULTS[1,4] = r(mean)
summ effectMAX, meanonly
matrix RESULTS[1,5] = r(mean)
summ N, meanonly
matrix RESULTS[1,6] = r(mean)

summ teffectMED, meanonly
matrix RESULTS[2,1] = r(mean)
summ teffectMIN, meanonly
matrix RESULTS[2,2] = r(mean)
summ teffectP5, meanonly
matrix RESULTS[2,3] = r(mean)
summ teffectP95, meanonly
matrix RESULTS[2,4] = r(mean)
summ teffectMAX, meanonly
matrix RESULTS[2,5] = r(mean)
summ N, meanonly
matrix RESULTS[2,6] = r(mean)

summ petMED, meanonly
matrix RESULTS[3,1] = r(mean)
summ petMIN, meanonly
matrix RESULTS[3,2] = r(mean)
summ petP5, meanonly
matrix RESULTS[3,3] = r(mean)
summ petP95, meanonly
matrix RESULTS[3,4] = r(mean)
summ petMAX, meanonly
matrix RESULTS[3,5] = r(mean)
summ N, meanonly
matrix RESULTS[3,6] = r(mean)

summ I2, detail
matrix RESULTS[4,1] = r(mean)
matrix RESULTS[4,2] = r(min)
```

```
matrix RESULTS[4,3] = r(p5)
matrix RESULTS[4,4] = r(p95)
matrix RESULTS[4,5] = r(max)

summ N, meanonly
matrix RESULTS[4,6] = r(mean)

matrix colnames RESULTS = MED MIN P5 P95 MAX N
matrix rownames RESULTS = EFFECT TEFFECT PET HETERO
matrix list RESULTS

etime
```

.do file for TABLES 2&3 – Part A

NOTE: The yellow-highlighted part of the code determines which type of publication bias is operative (TABLE 2 = bias against insignificance; TABLE 3 = bias against wrong sign)

```
set more off
program drop _all
program define REpbias, rclass
version 13.1

syntax, alpha(real) studies(integer) select(real) obs(integer)

// Remove existing variables
drop _all

//We first create the matrix to store the results of each study
matrix A = J(`studies',1,0)
matrix B = J(`studies',1,0)
matrix C = J(`studies',1,0)

forvalues i = 1/`studies' {
        clear
        // STEP ONE: Create the data for each study and estimate an effect
        set obs `obs'
        generate x = rnormal()
        // Note that each "study" has the same number of observations (100)
        // but differ in the variance of their respective error terms. This
        // causes the estimate of the effect to be estimated with varying degrees
        // of precision
        scalar lambdai = 0.5+30*runiform()
        generate e = lambdai*rnormal()
        scalar alphai = `alpha' + rnormal()
        generate y = 1 + alphai*x + e
```

```
        quietly regress y x
        scalar coef = _b[x]
        scalar secoef = _se[x]
        scalar tcoef = coef/secoef

        //if abs(tcoef) < 2 {
        // An alternative criterion is that results that show a negative effect
        // have a harder time getting published. To study that case, substitute
        // the line below for the line above
        if coef < 0 {
        scalar dummy = cond(runiform()<`select',1,.)
        // The statement above creates a dummy variable that randomly selects which
        // studies will get "published" if they fail to meet the "publication criterion"
        // either (i) abs(tcoef) >= 2 or (ii) if coef >= 0. Studies that are not "published"
        // receive missing values and thus are not included in the "meta-analysis."
         scalar coef = dummy*coef
        scalar secoef = dummy*secoef
        scalar tcoef = dummy*tcoef
        }

        matrix A[`i',1] = coef
        matrix B[`i',1] = secoef
        matrix C[`i',1] = tcoef
}

// The next set of commands moves the data out of matrices and reformats them as
// standard Stata data series. We have now completed generating our individual
// studies and we now move into the "meta-analysis" stage.
matrix bob = A,B,C
svmat bob
rename bob1 effect
rename bob2 seeffect
rename bob3 teffect
generate pet = (1/seeffect)
```

```
// This estimate produces the OLS estimate of the effect
regress effect
return scalar effect_OLS = _b[_cons]
test _b[_cons] = `alpha'
return scalar pvalue_OLS = r(p)

// This estimate produces the PET version of the "publication bias"-
// corrected effect estimate
regress teffect pet, vce(robust)
return scalar effect_PET = _b[pet]
test pet = `alpha'
return scalar pvalue_PET = r(p)

// This estimate produces the PEESE version of the "publication bias"-
// corrected effect estimate
regress teffect seeffect pet, noc vce(robust)
return scalar effect_PEESE = _b[pet]
test pet = `alpha'
return scalar pvalue_PEESE = r(p)

// This estimate produces the FE estimate of the effect
generate constant = 1
vwls effect constant, sd(seeffect) nocon
return scalar effect_FE = _b[constant]
test _b[constant] = `alpha'
return scalar pvalue_FE = r(p)

// This estimate produces the WLS estimate of the effect
regress teffect pet, noc
return scalar effect_WLS = _b[pet]
test pet = `alpha'
return scalar pvalue_WLS = r(p)

// This estimate produces the RE estimate of the effect
// NOTE: We use the Method of Moments (mm) option of metareg
```

```
// because the maximum likelihood procedure had too many instances
// of failure to optimize. Method of Moments does not require
// iteration and thus avoids this problem.
metareg effect , wsse(seeffect) mm
matrix bill = e(b)
return scalar effect_RE = bill[1,1]
test _cons = `alpha'
return scalar pvalue_RE = r(p)

// This last command keeps track of how many studies are in our "meta-analysis"
return scalar N = e(N)

end
```

.do file for TABLE 2&3 – Part B

```
// This program takes about 7 days to run on my laptop
etime, start
drop _all
clear
graph drop _all
set more off
set seed 13
set matsize 5000
matrix EFFECT = J(9,7,0)
matrix MSE = J(9,7,0)
matrix TEST = J(9,7,0)
local studies = 1000
// Select is used to set the probability of being included in the data set
// when the study is subject to publication bias, either because (i) abs(tstat) < 2, or
// coef < 0.
local select = 0.10
local obs = 100

local i = 1
foreach alpha in 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 {
simulate effect_OLS = r(effect_OLS) effect_FE = r(effect_FE) effect_RE = r(effect_RE) ///
        effect_PET = r(effect_PET) effect_PEESE = r(effect_PEESE) ///
        pvalue_OLS = r(pvalue_OLS) pvalue_FE = r(pvalue_FE) pvalue_RE = r(pvalue_RE)
///
        pvalue_PET = r(pvalue_PET) pvalue_PEESE = r(pvalue_PEESE) N = r(N) ///
        effect_WLS = r(effect_WLS) pvalue_WLS = r(pvalue_WLS) , ///
    reps(10000): REpbias, alpha(`alpha') studies(`studies') select(`select') obs(`obs')

        summ N, meanonly
        matrix EFFECT[`i',1] = r(mean)
        summ effect_OLS, meanonly
        matrix EFFECT[`i',2] = r(mean)
        summ effect_PET, meanonly
```

```
matrix EFFECT[`i',3] = r(mean)
summ effect_PEESE, meanonly
matrix EFFECT[`i',4] = r(mean)
summ effect_FE, meanonly
matrix EFFECT[`i',5] = r(mean)
summ effect_WLS, meanonly
matrix EFFECT[`i',6] = r(mean)
summ effect_RE, meanonly
matrix EFFECT[`i',7] = r(mean)

summ N, meanonly
matrix MSE[`i',1] = r(mean)
generate mse_OLS = (effect_OLS - `alpha')^2
summ mse_OLS, meanonly
matrix MSE[`i',2] = r(mean)
generate mse_PET = (effect_PET - `alpha')^2
summ mse_PET, meanonly
matrix MSE[`i',3] = r(mean)
generate mse_PEESE = (effect_PEESE - `alpha')^2
summ mse_PEESE, meanonly
matrix MSE[`i',4] = r(mean)
generate mse_FE = (effect_FE - `alpha')^2
summ mse_FE, meanonly
matrix MSE[`i',5] = r(mean)
generate mse_WLS = (effect_WLS - `alpha')^2
summ mse_WLS, meanonly
matrix MSE[`i',6] = r(mean)
generate mse_RE = (effect_RE - `alpha')^2
summ mse_RE, meanonly
matrix MSE[`i',7] = r(mean)

summ N, meanonly
matrix TEST[`i',1] = r(mean)
generate RROLS = 0
replace RROLS = cond(pvalue_OLS<0.05,1,0)
```

```
        summ RROLS, meanonly
        matrix TEST[`i',2] = r(mean)
        generate RRPET = 0
        replace RRPET = cond(pvalue_PET<0.05,1,0)
        summ RRPET, meanonly
        matrix TEST[`i',3] = r(mean)
        generate RRPEESE = 0
        replace RRPEESE = cond(pvalue_PEESE<0.05,1,0)
        summ RRPEESE, meanonly
        matrix TEST[`i',4] = r(mean)
        generate RRFE = 0
        replace RRFE = cond(pvalue_FE<0.05,1,0)
        summ RRFE, meanonly
        matrix TEST[`i',5] = r(mean)
        generate RRWLS = 0
        replace RRWLS = cond(pvalue_WLS<0.05,1,0)
        summ RRWLS, meanonly
        matrix TEST[`i',6] = r(mean)
        generate RRRE = 0
        replace RRRE = cond(pvalue_RE<0.05,1,0)
        summ RRRE, meanonly
        matrix TEST[`i',7] = r(mean)

        local `++i'
}

matrix colnames EFFECT = N OLS PET PEESE FE WLS RE
matrix rownames EFFECT = A0 A0P5 A1 A1P5 A2 A2P5 A3 A3P5 A4
matrix colnames MSE = N OLS PET PEESE FE WLS RE
matrix rownames MSE = A0 A0P5 A1 A1P5 A2 A2P5 A3 A3P5 A4
matrix colnames TEST = N OLS PET PEESE FE WLS RE
matrix rownames TEST = A0 A0P5 A1 A1P5 A2 A2P5 A3 A3P5 A4
matrix list EFFECT
matrix list MSE
matrix list TEST
```

etime

.do file for TABLE 4 – Part A

NOTE: The yellow-highlighted part of the code determines which type of publication bias is operative

```
set more off
program drop _all
program define PANELsample, rclass
version 13.1

syntax, studies(integer) estperstudy(integer) totalobs(integer) alpha(real) ///
theta(real) obs(integer)

// Remove existing variables
drop _all

//We first create the matrix to store the results of each study
matrix A = J(`totalobs',1,.)
matrix B = J(`totalobs',1,.)
matrix C = J(`totalobs',1,.)
matrix D = J(`totalobs',1,.)

forvalues i = 1/`studies' {
        scalar lambdai = 0.5+30*runiform()
        scalar alphai = `alpha'+2*rnormal()
        forvalues j = 1/`estperstudy' {
                // STEP ONE: Create the data for each study and estimate an effect
                clear
                set obs 100
                generate x = rnormal()
                // Note that each "study" has a difference error variance, causing the
                estimate
                // of the effect to be estimated with varying degrees of precision
                // This is for random effects
                scalar lambdaij = lambdai+`theta'*runiform()
```

```
                scalar alphaij = alphai+0.5*rnormal()
                generate e = lambdaij*rnormal()
                generate y = 1 + alphaij*x + e
                quietly regress y x
                scalar coef = _b[x]
                scalar secoef = _se[x]
                scalar tcoef = coef/secoef
                scalar ID = `i'
                scalar obsno = (`i'-1)*`estperstudy'+`j'

                matrix A[obsno,1] = coef
                matrix B[obsno,1] = secoef
                matrix C[obsno,1] = tcoef
                matrix D[obsno,1] = ID
        }
}

matrix bob = A,B,C,D
svmat bob
rename bob1 effect
rename bob2 seeffect
rename bob3 teffect
rename bob4 ID
generate pet = (1/seeffect)

// First run this program once to get the pre-publication study sample data
// To get post-publication study sample data, uncomment one of the two sections
// below.
generate dummy = 1
//replace dummy = 0 if abs(teffect) < 2
//replace dummy = 0 if effect < 0
by ID, sort: egen select = mean(dummy)
replace effect = cond(select<0.65,.,effect)
replace seeffect = cond(select<0.65,.,seeffect)
replace teffect = cond(select<0.65,.,teffect)
```

```
replace pet = cond(select<0.65,.,pet)

metareg effect , wsse(seeffect) mm
return scalar I2 = e(I2)

summ effect, detail
return scalar effectMED = r(p50)
return scalar effectMIN = r(min)
return scalar effectP5 = r(p5)
return scalar effectP95 = r(p95)
return scalar effectMAX = r(max)
return scalar N = r(N)

summ teffect, detail
return scalar teffectMED = r(p50)
return scalar teffectMIN = r(min)
return scalar teffectP5 = r(p5)
return scalar teffectP95 = r(p95)
return scalar teffectMAX = r(max)

summ pet, detail
return scalar petMED = r(p50)
return scalar petMIN = r(min)
return scalar petP5 = r(p5)
return scalar petP95 = r(p95)
return scalar petMAX = r(max)

end
```

.do file for TABLE 4 – Part B

```
// This program takes approximately 2 hours to run on my laptop for each section
etime, start
drop _all
clear
set matsize 5000
graph drop _all
set more off
set seed 52256
matrix RESULTS = J(4,6,0)
local studies = 100
local estperstudy = 10
local totalobs = `studies'*`estperstudy'
local alpha = 1
local theta = 1
local obs = 100

simulate effectMED = r(effectMED) effectMIN = r(effectMIN) effectP5 = r(effectP5) ///
        effectP95 = r(effectP95) effectMAX = r(effectMAX) ///
        teffectMED = r(teffectMED) teffectMIN = r(teffectMIN) teffectP5 = r(teffectP5)
///
        teffectP95 = r(teffectP95) teffectMAX = r(teffectMAX) ///
                        petMED = r(petMED) petMIN = r(petMIN) petP5 = r(petP5) ///
        petP95 = r(petP95) petMAX = r(petMAX) N = r(N) I2 = r(I2), ///
    reps(1000): PANELsample, studies(`studies') estperstudy(`estperstudy')
totalobs(`totalobs') ///    alpha(`alpha') theta(`theta') obs(`obs')


summ effectMED, meanonly
matrix RESULTS[1,1] = r(mean)
summ effectMIN, meanonly
matrix RESULTS[1,2] = r(mean)
summ effectP5, meanonly
matrix RESULTS[1,3] = r(mean)
```

```
summ effectP95, meanonly
matrix RESULTS[1,4] = r(mean)
summ effectMAX, meanonly
matrix RESULTS[1,5] = r(mean)
summ N, meanonly
matrix RESULTS[1,6] = r(mean)

summ teffectMED, meanonly
matrix RESULTS[2,1] = r(mean)
summ teffectMIN, meanonly
matrix RESULTS[2,2] = r(mean)
summ teffectP5, meanonly
matrix RESULTS[2,3] = r(mean)
summ teffectP95, meanonly
matrix RESULTS[2,4] = r(mean)
summ teffectMAX, meanonly
matrix RESULTS[2,5] = r(mean)
summ N, meanonly
matrix RESULTS[2,6] = r(mean)

summ petMED, meanonly
matrix RESULTS[3,1] = r(mean)
summ petMIN, meanonly
matrix RESULTS[3,2] = r(mean)
summ petP5, meanonly
matrix RESULTS[3,3] = r(mean)
summ petP95, meanonly
matrix RESULTS[3,4] = r(mean)
summ petMAX, meanonly
matrix RESULTS[3,5] = r(mean)
summ N, meanonly
matrix RESULTS[3,6] = r(mean)

summ I2, detail
matrix RESULTS[4,1] = r(mean)
```

```
matrix RESULTS[4,2] = r(min)
matrix RESULTS[4,3] = r(p5)
matrix RESULTS[4,4] = r(p95)
matrix RESULTS[4,5] = r(max)

summ N, meanonly
matrix RESULTS[4,6] = r(mean)

matrix colnames RESULTS = MED MIN P5 P95 MAX N
matrix rownames RESULTS = EFFECT TEFFECT PET HETERO
matrix list RESULTS

etime
```

.do file for TABLES 5&6 – Part A

NOTE: The yellow-highlighted part of the code determines which type of publication bias is operative (TABLE 5 = bias against insignificance; TABLE 6 = bias against wrong sign)

```
program drop _all
program define PANELpbias, rclass
version 13.1

syntax, studies(integer) estperstudy(integer) totalobs(integer) alpha(real) ///
theta(real) obs(integer)

// Remove existing variables
drop _all

//We first create the matrix to store the results of each study
set matsize 5000
matrix A = J(`totalobs',1,.)
matrix B = J(`totalobs',1,.)
matrix C = J(`totalobs',1,.)
matrix D = J(`totalobs',1,.)

forvalues i = 1/`studies' {
        scalar lambdai = 0.5+30*runiform()
        scalar alphai = `alpha'+2*rnormal()
        forvalues j = 1/`estperstudy' {
                // STEP ONE: Create the data for each study and estimate an effect
                clear
                set obs 100
                generate x = rnormal()
                // Note that each "study" has a difference error variance, causing the
                estimate
                // of the effect to be estimated with varying degrees of precision
```

```
            // This is for random effects
            scalar lambdaij = lambdai+`theta'*runiform()
            scalar alphaij = alphai+0.5*rnormal()
            generate e = lambdaij*rnormal()
            generate y = 1 + alphaij*x + e
            // This is for fixed effects
            // scalar lambda = 0.2+30*runiform()
            // generate e = lambda*rnormal()
            // generate y = 1 + `alpha'*x + e
            quietly regress y x
            scalar coef = _b[x]
            scalar secoef = _se[x]
            scalar tcoef = coef/secoef
            scalar ID = `i'
            scalar obsno = (`i'-1)*`estperstudy'+`j'

            // First run this program once to get the pre-publication study sample
            data
            // To get post-publication study sample data, uncomment one of the
            two sections
            // below.
            matrix A[obsno,1] = coef
            matrix B[obsno,1] = secoef
            matrix C[obsno,1] = tcoef
        matrix D[obsno,1] = ID
        }
}

// The next set of commands moves the data out of matrices and reformats them as
// standard Stata data series. We have now completed generating our individual
// studies and we now move into the "meta-analysis" stage.
matrix bob = A,B,C,D
svmat bob
rename bob1 effect
rename bob2 seeffect
```

```
rename bob3 teffect
rename bob4 ID
generate pet = (1/seeffect)

// This set of commands imposese the publication bias, where the publication criterion
// is either that (i) the t-stat must be greater than or equal to 2, or (ii) the
// estimated effect is positive. The commands below implement the assumption that a
// study must have at least 7 out of 10 estimates that satisfy the publication
// criterion in order for the study to be "published."
generate dummy = 1
//replace dummy = 0 if abs(teffect) < 2
replace dummy = 0 if effect < 0
by ID, sort: egen select = mean(dummy)
// Not sure why this happens, but if I put select<0.70, it kicks out the studies that
// have 7 estimates that satisfy the publication criterion. So I set select<0.65.
// Studies are omitted from the "meta-analysis" sample by replacing the relevant variables
// with missing values.
replace effect = cond(select<0.65,.,effect)
replace seeffect = cond(select<0.65,.,seeffect)
replace teffect = cond(select<0.65,.,teffect)
replace pet = cond(select<0.65,.,pet)

// This creates dummy variables for each of the 100 studies
// The dummy variables take names dum1 to dum100
tab ID, gen(dum)

// This creates study-specific SE terms for use in the PEESE
// according to equation 5.7 on page 85 of S&D
forvalues i = 1/100 {
        generate SE`i' = seeffect*dum`i'
}

// This estimate produces the OLS estimate of the effect
regress effect, vce(cluster ID)
```

```
return scalar effect_OLS = _b[_cons]
test _b[_cons] = `alpha'
return scalar pvalue_OLS = r(p)

// This estimate produces the PET version of the "publication bias"-
// corrected effect estimate
// NOTE1: According to equation 5.6 on page 85 of S&D, the bias-corrected
// effect is given by the coefficients on the respective precision terms, pet*.
// The specification below forces all the effects to be the same, while
// allowing for fixed effects to correct for bias-associated with estimate SEs.
// NOTE2: Also note that while all the dummy variables will not be included in the
// meta-analysis sample, this is not a problem because STATA will automatically
// kick out the ones that don't belong.
regress teffect dum1-dum100 pet, vce(cluster ID)
return scalar effect_PET = _b[pet]
test pet = `alpha'
return scalar pvalue_PET = r(p)

// This estimate produces the PEESE version of the "publication bias"-
// corrected effect estimate. It is based on equation 5.7 on page 85 of S&D.
// See notes from above.
regress teffect SE1-SE100 pet, noc vce(cluster ID)
return scalar effect_PEESE = _b[pet]
test pet = `alpha'
return scalar pvalue_PEESE = r(p)

// This estimate produces the FE estimate of the effect
// Note that the FE estimator cannot do cluster robust
generate constant = 1
vwls effect constant, sd(seeffect) nocon
return scalar effect_FE = _b[constant]
test _b[constant] = `alpha'
return scalar pvalue_FE = r(p)

// This estimate produces the WLS estimate of the effect
```

```
regress teffect pet, noc vce(cluster ID)
return scalar effect_WLS = _b[pet]
test pet = `alpha'
return scalar pvalue_WLS = r(p)

// This estimate produces the RE estimate of the effect
// NOTE: We use the Method of Moments (mm) option of metareg
// because the maximum likelihood procedure had too many instances
// of failure to optimize. Method of Moments does not require
// iteration and thus avoids this problem.
quietly metareg effect, wsse(seeffect) mm
scalar tau2 = e(tau2)
gen revarR= seeffect^2 + tau2
gen reseR = sqrt(revarR)
gen reteffect = effect/reseR
gen repet = 1/reseR
regress reteffect repet, noc vce(cluster ID)
return scalar effect_RE = _b[repet]
test repet = `alpha'
return scalar pvalue_RE = r(p)

// This last command keeps track of how many estimates are in our "meta-analysis"
return scalar N = e(N)

end
```

.do file for TABLE 5&6 – Part B

```
// This program takes about 8 days to run on my laptop
etime, start
drop _all
clear
graph drop _all
set more off
set seed 13
set matsize 5000
matrix EFFECT = J(9,7,0)
matrix MSE = J(9,7,0)
matrix TEST = J(9,7,0)
local studies = 100
local estperstudy = 10
local totalobs = `studies'*`estperstudy'
local theta = 1
local obs = 100

local i = 1
foreach alpha in 0 0.5 1 1.5 2 2.5 3 3.5 4 {
    simulate effect_OLS = r(effect_OLS) effect_FE = r(effect_FE) effect_RE = r(effect_RE)
///
        effect_PET = r(effect_PET) effect_PEESE = r(effect_PEESE) ///
        pvalue_OLS = r(pvalue_OLS) pvalue_FE = r(pvalue_FE) pvalue_RE = r(pvalue_RE)
///
        pvalue_PET = r(pvalue_PET) pvalue_PEESE = r(pvalue_PEESE) N = r(N) ///
        effect_WLS = r(effect_WLS) pvalue_WLS = r(pvalue_WLS) , ///
    reps(10000): PANELpbias, studies(`studies') estperstudy(`estperstudy')
totalobs(`totalobs') ///    alpha(`alpha') theta(`theta') obs(`obs')

    summ N, meanonly
    matrix EFFECT[`i',1] = r(mean)
        summ effect_OLS, meanonly
    matrix EFFECT[`i',2] = r(mean)
```

```
summ effect_PET, meanonly
matrix EFFECT[`i',3] = r(mean)
summ effect_PEESE, meanonly
matrix EFFECT[`i',4] = r(mean)
    summ effect_FE, meanonly
matrix EFFECT[`i',5] = r(mean)
summ effect_WLS, meanonly
matrix EFFECT[`i',6] = r(mean)
summ effect_RE, meanonly
matrix EFFECT[`i',7] = r(mean)

    summ N, meanonly
matrix MSE[`i',1] = r(mean)
    generate mse_OLS = (effect_OLS - `alpha')^2
summ mse_OLS, meanonly
matrix MSE[`i',2] = r(mean)
    generate mse_PET = (effect_PET - `alpha')^2
    summ mse_PET, meanonly
matrix MSE[`i',3] = r(mean)
    generate mse_PEESE = (effect_PEESE - `alpha')^2
summ mse_PEESE, meanonly
matrix MSE[`i',4] = r(mean)
    generate mse_FE = (effect_FE - `alpha')^2
summ mse_FE, meanonly
matrix MSE[`i',5] = r(mean)
    generate mse_WLS = (effect_WLS - `alpha')^2
summ mse_WLS, meanonly
matrix MSE[`i',6] = r(mean)
    generate mse_RE = (effect_RE - `alpha')^2
summ mse_RE, meanonly
matrix MSE[`i',7] = r(mean)

summ N, meanonly
matrix TEST[`i',1] = r(mean)
    generate RROLS = 0
```

```
        replace RROLS = cond(pvalue_OLS<0.05,1,0)
        summ RROLS, meanonly
    matrix TEST[`i',2] = r(mean)
        generate RRPET = 0
        replace RRPET = cond(pvalue_PET<0.05,1,0)
        summ RRPET, meanonly
    matrix TEST[`i',3] = r(mean)
        generate RRPEESE = 0
        replace RRPEESE = cond(pvalue_PEESE<0.05,1,0)
        summ RRPEESE, meanonly
    matrix TEST[`i',4] = r(mean)
        generate RRFE = 0
        replace RRFE = cond(pvalue_FE<0.05,1,0)
        summ RRFE, meanonly
    matrix TEST[`i',5] = r(mean)
        generate RRWLS = 0
        replace RRWLS = cond(pvalue_WLS<0.05,1,0)
        summ RRWLS, meanonly
    matrix TEST[`i',6] = r(mean)
        generate RRRE = 0
        replace RRRE = cond(pvalue_RE<0.05,1,0)
        summ RRRE, meanonly
    matrix TEST[`i',7] = r(mean)

    local `++i'
}

matrix colnames EFFECT = N OLS PET PEESE FE WLS RE
matrix rownames EFFECT = A0 A0P5 A1 A1P5 A2 A2P5 A3 A3P5 A4
matrix colnames MSE = N OLS PET PEESE FE WLS RE
matrix rownames MSE = A0 A0P5 A1 A1P5 A2 A2P5 A3 A3P5 A4
matrix colnames TEST = N OLS PET PEESE FE WLS RE
matrix rownames TEST = A0 A0P5 A1 A1P5 A2 A2P5 A3 A3P5 A4
matrix list EFFECT
matrix list MSE
```

```
matrix list TEST

etime
```

.do file for FIGURE 2

NOTE: The yellow-highlighted part of the code determines which type of publication bias is operative

```
clear
set seed 13
set more off
set matsize 5000
local studies = 1000
local alpha = 1
local select = 0.10
matrix A = J(`studies',1,0)
matrix B = J(`studies',1,0)
matrix C = J(`studies',1,0)

forvalues i = 1/`studies' {
        // STEP ONE: Create the data for each study and estimate an effect
        clear
        set obs 100
        generate x = rnormal()
        // Note that each "study" has a difference error variance, causing the estimate
        // of the effect to be estimated with varying degrees of precision
        scalar lambda = 0.5+30*runiform()
        generate e = lambda*rnormal()
        scalar alpha = `alpha' + rnormal()
```

```
        generate y = 1 + alpha*x + e
        quietly regress y x
        scalar coef = _b[x]
        scalar secoef = _se[x]
        scalar tcoef = abs(coef/secoef)

        // First run this program once to get the pre-publication study sample data
        // To get post-publication study sample data, uncomment one of the two
        sections
        // below.

        /*if tcoef < 2 {
        scalar dummy = cond(runiform()<`select',1,.)
        scalar coef = dummy*coef
        scalar secoef = dummy*secoef
        scalar tcoef = dummy*tcoef
        }
        */

        /*if coef < 0 {
        scalar dummy = cond(runiform()<`select',1,.)
        scalar coef = dummy*coef
        scalar secoef = dummy*secoef
        scalar tcoef = dummy*tcoef
        }
        */

        matrix A[`i',1] = coef
        matrix B[`i',1] = secoef
        matrix C[`i',1] = tcoef
}

matrix bob = A,B,C
svmat bob
rename bob1 effect
```

```
rename bob2 seeffect
rename bob3 teffect
generate pet = (1/seeffect)

metafunnel effect seeffect, xline(`alpha') ylabel(#8) nolines
```

.do file for FIGURE 3 – Part A

NOTE: The yellow-highlighted part of the code determines which type of publication bias is operative

```
set more off
program drop _all
program define REfig, rclass
version 13.1

syntax, alpha(real) studies(integer) select(real) obs(integer)

// Remove existing variables
drop _all

//We first create the matrix to store the results of each study
matrix A = J(`studies',1,0)
matrix B = J(`studies',1,0)
matrix C = J(`studies',1,0)

forvalues i = 1/`studies' {
        clear
        // STEP ONE: Create the data for each study and estimate an effect
        set obs `obs'
        generate x = rnormal()
        // Note that each "study" has the same number of observations (100)
        // but differ in the variance of their respective error terms. This
        // causes the estimate of the effect to be estimated with varying degrees
        // of precision
        scalar lambda = 0.5+30*runiform()
        generate e = lambda*rnormal()
        scalar alpha = `alpha' + rnormal()
        generate y = 1 + alpha*x + e
        quietly regress y x
        scalar coef = _b[x]
```

```
        scalar secoef = _se[x]
        scalar tcoef = coef/secoef

        //if abs(tcoef) < 2 {
        // An alternative criterion is that results that show a negative effect
        // have a harder time getting published. To study that case, substitute
        // the line below for the line above
        if coef < 0 {
                scalar dummy = cond(runiform()<`select',1,.)
                // The statement above creates a dummy variable that randomly selects
                which
                // studies will get "published" if they fail to meet the "publication
                criterion"
                // either (i) abs(tcoef) >= 2 or (ii) if coef >= 0. Studies that are not
                "published"
                // receive missing values and thus are not included in the "meta-
                analysis."
                scalar coef = dummy*coef
                scalar secoef = dummy*secoef
                scalar tcoef = dummy*tcoef
        }

        matrix A[`i',1] = coef
        matrix B[`i',1] = secoef
        matrix C[`i',1] = tcoef
}

// The next set of commands moves the data out of matrices and reformats them as
// standard Stata data series. We have now completed generating our individual
// studies and we now move into the "meta-analysis" stage.
matrix bob = A,B,C
svmat bob
rename bob1 effect
rename bob2 seeffect
rename bob3 teffect
```

```
generate pet = (1/seeffect)

// This estimate produces the RE estimate of the effect
metareg effect , wsse(seeffect) mm
matrix bill = e(b)
return scalar effect_RE = bill[1,1]

// This estimate produces the PEESE version of the "publication bias"-
// corrected effect estimate.
regress teffect seeffect pet, noc
return scalar effect_PEESE = _b[pet]

end
```

.do file for FIGURE 3 – Part B

```
// This program takes about an hour and 20 minutes to run on my laptop
etime, start
drop _all
clear
set matsize 5000
graph drop _all
set more off
set seed 13
matrix EFFECT = J(9,6,0)
matrix MSE = J(9,6,0)
matrix TEST = J(9,6,0)
local studies = 1000
// Select is used to set the probability of being included in the data set
// when the study is subject to publication bias, either because (i) abs(tstat) < 2, or
// coef < 0.
local select = 0.10
local obs = 100

local i = 1
foreach alpha in 3 {
      simulate effect_RE = r(effect_RE) effect_PEESE = r(effect_PEESE), ///
      reps(1000): REfig, studies(`studies') alpha(`alpha') select(`select') obs(`obs')
                                    local `++i'
}


kdensity effect_PEESE , xline(3) name(PEESE) //xlabel(-4(1) 4)
kdensity effect_RE , xline(3) name(Combined) addplot(kdensity effect_PEESE) //xlabel(-
4(1) 4)

etime
```

.do file for FIGURE 5

NOTE: The yellow-highlighted part of the code determines which type of publication bias is operative.

```
etime, start
clear
set seed 13
set more off
local studies = 100
local estperstudy = 10
local totalobs = `studies'*`estperstudy'
local alpha = 1
local theta = 1
set matsize 10000
matrix A = J(`totalobs',1,.)
matrix B = J(`totalobs',1,.)
matrix C = J(`totalobs',1,.)
matrix D = J(`totalobs',1,.)


forvalues i = 1/`studies' {
        scalar lambdai = 0.5+30*runiform()
        scalar alphai = `alpha'+2*rnormal()
        forvalues j = 1/`estperstudy' {
                // STEP ONE: Create the data for each study and estimate an effect
                clear
                set obs 100
                generate x = rnormal()
                // Note that each "study" has a difference error variance, causing the
                estimate
                // of the effect to be estimated with varying degrees of precision
                // This is for random effects
                scalar lambdai = lambdai+`theta'*runiform()
                scalar alphaij = alphai+0.5*rnormal()
```

```
                generate e = lambdai*rnormal()
                generate y = 1 + alphaij*x + e
                quietly regress y x
                scalar coef = _b[x]
                scalar secoef = _se[x]
                scalar tcoef = coef/secoef
                scalar ID = `i'
                scalar obsno = (`i'-1)*`estperstudy'+`j'

                matrix A[obsno,1] = coef
                matrix B[obsno,1] = secoef
                matrix C[obsno,1] = tcoef
                matrix D[obsno,1] = ID
        }
}

matrix bob = A,B,C,D
svmat bob
rename bob1 effect
rename bob2 seeffect
rename bob3 teffect
rename bob4 ID
generate pet = (1/seeffect)

// First run this program once to get the pre-publication study sample data
// To get post-publication study sample data, uncomment one of the two sections
// below.
generate dummy = 1
//replace dummy = 0 if abs(teffect) < 2
//replace dummy = 0 if effect < 0
by ID, sort: egen select = mean(dummy)
replace effect = cond(select<0.65,.,effect)
replace seeffect = cond(select<0.65,.,seeffect)
replace teffect = cond(select<0.65,.,teffect)
replace pet = cond(select<0.65,.,pet)
```

metafunnel effect seeffect, xline(1) nolines ylabel(#8)

etime

.do file for FIGURE 6 – Part A

NOTE: The yellow-highlighted part of the code determines which type of publication bias is operative

```
set more off
program drop _all
program define PANELfig, rclass
version 13.1

syntax, studies(integer) estperstudy(integer) totalobs(integer) alpha(real) ///
theta(real) obs(integer)

// Remove existing variables
drop _all

//We first create the matrix to store the results of each study
set matsize 10000
matrix A = J(`totalobs',1,.)
matrix B = J(`totalobs',1,.)
matrix C = J(`totalobs',1,.)
matrix D = J(`totalobs',1,.)

forvalues i = 1/`studies' {
        scalar lambdai = 0.5+30*runiform()
        scalar alphai = `alpha'+2*rnormal()
        forvalues j = 1/`estperstudy' {
                // STEP ONE: Create the data for each study and estimate an effect
                clear
                set obs 100
                generate x = rnormal()
                // Note that each "study" has a difference error variance, causing the
                estimate
                // of the effect to be estimated with varying degrees of precision
                // This is for random effects
```

```
            scalar lambda = lambdai+`theta'*runiform()
            scalar alphaij = alphai+0.5*rnormal()
            generate e = lambda*rnormal()
            generate y = 1 + alphaij*x + e
            // This is for fixed effects
            // scalar lambda = 0.2+30*runiform()
            // generate e = lambda*rnormal()
            // generate y = 1 + `alpha'*x + e
            quietly regress y x
            scalar coef = _b[x]
            scalar secoef = _se[x]
            scalar tcoef = coef/secoef
            scalar ID = `i'
            scalar obsno = (`i'-1)*`estperstudy'+`j'

            // First run this program once to get the pre-publication study sample
            data
            // To get post-publication study sample data, uncomment one of the
            two sections
            // below.
            matrix A[obsno,1] = coef
            matrix B[obsno,1] = secoef
            matrix C[obsno,1] = tcoef
            matrix D[obsno,1] = ID
        }
}

// The next set of commands moves the data out of matrices and reformats them as
// standard Stata data series. We have now completed generating our individual
// studies and we now move into the "meta-analysis" stage.
matrix bob = A,B,C,D
svmat bob
rename bob1 effect
rename bob2 seeffect
rename bob3 teffect
```

```
rename bob4 ID
generate pet = (1/seeffect)

// This set of commands imposese the publication bias, where the publication criterion
// is either that (i) the t-stat must be greater than or equal to 2, or (ii) the
// estimated effect is positive. The commands below implement the assumption that a
// study must have at least 7 out of 10 estimates that satisfy the publication
// criterion in order for the study to be "published."
generate dummy = 1
replace dummy = 0 if abs(teffect) < 2
//replace dummy = 0 if effect < 0
by ID, sort: egen select = mean(dummy)
// Not sure why this happens, but if I put select<0.70, it kicks out the studies that
// have 7 estimates that satisfy the publication criterion. So I set select<0.70.
// Studies are omitted from the "meta-analysis" sample by replacing the relevant
variables
// with missing values.
replace effect = cond(select<0.65,.,effect)
replace seeffect = cond(select<0.65,.,seeffect)
replace teffect = cond(select<0.65,.,teffect)
replace pet = cond(select<0.65,.,pet)

// This creates dummy variables for each of the 100 studies
// The dummy variables take names dum1 to dum100
tab ID, gen(dum)

// This creates study-specific SE terms for use in the PEESE
// according to equation 5.7 on page 85 of S&D
forvalues i = 1/100 {
        generate SE`i' = seeffect*dum`i'
}

// This estimate produces the OLS estimate of the effect
regress effect
return scalar effect_OLS = _b[_cons]
```

```
// This estimate produces the FE estimate of the effect
generate constant = 1
vwls effect constant, sd(seeffect) nocon
return scalar effect_FE = _b[constant]

// This estimate produces the RE estimate of the effect
// NOTE: We use the Method of Moments (mm) option of metareg
// because the maximum likelihood procedure had too many instances
// of failure to optimize. Method of Moments does not require
// iteration and thus avoids this problem.
metareg effect , wsse(seeffect) mm
matrix bill = e(b)
return scalar effect_RE = bill[1,1]

// This estimate produces the PET version of the "publication bias"-
// corrected effect estimate
// NOTE1: According to equation 5.6 on page 85 of S&D, the bias-corrected
// effect is given by the coefficients on the respective precision terms, pet*.
// The specification below forces all the effects to be the same, while
// allowing for fixed effects to correct for bias-associated with estimate SEs.
// NOTE2: Also note that while all the dummy variables will not be included in the
// meta-analysis sample, this is not a problem because STATA will automatically
// kick out the ones that don't belong.
regress teffect dum1-dum100 pet, vce(cluster ID)
return scalar effect_PET = _b[pet]

// This estimate produces the PEESE version of the "publication bias"-
// corrected effect estimate. It is based on equation 5.7 on page 85 of S&D.
// See notes from above.
regress teffect SE1-SE100 pet, noc
return scalar effect_PEESE = _b[pet]

end
```

.do file for FIGURE 6 – Part B

```
// This program takes about an hour and 20 minutes to run on my laptop
etime, start
drop _all
clear
set matsize 5000
graph drop _all
set more off
set seed 13
matrix EFFECT = J(9,6,0)
matrix MSE = J(9,6,0)
matrix TEST = J(9,6,0)
local studies = 1000
// Select is used to set the probability of being included in the data set
// when the study is subject to publication bias, either because (i) abs(tstat) < 2, or
// coef < 0.
local select = 0.10
local obs = 100

local i = 1
foreach alpha in 3 {
      simulate effect_RE = r(effect_RE) effect_PEESE = r(effect_PEESE), ///
      reps(1000): REfig, studies(`studies') alpha(`alpha') select(`select') obs(`obs')
                                    local `++i'
}

kdensity effect_PEESE , xline(3) name(PEESE) //xlabel(-4(1) 4)
kdensity effect_RE , xline(3) name(Combined) addplot(kdensity effect_PEESE) //xlabel(-
4(1) 4)

etime
```