APPENDIX:
Stata .do files that accompany this paper

NOTE #1: These .do files are included to make it possible for readers to (i) confirm this study's findings, and (ii) investigate alternative DGP specifications.

NOTE #2: The output for each table is produced by a two-part program. For Table X, the TABLEXA program must be run first, followed by the TABLEXB program. The latter program calls the former program and produces the results reported in the table. The computing time necessary to produce the results for a table are given at the beginning of the respective B programs.

“TABLE1A” program

```
program drop _all
program define hurwicz, rclass
version 13
syntax, beta0(real) betay(real) betau(real) smallobs(integer) bigobs(integer)
// Remove existing variables
drop _all

// Create the data
set obs `bigobs'
gen t = _n
tsset t
// We initialize y as its LR equilibrium value
gen y = `beta0'/(1-`betay')
replace y = `beta0' + `betay'*L.y + `betau'*rnormal() in 2/l
regress y L.y in -`smallobs'/l
return scalar bhaty = _b[L.y]
test _b[L.y] = `betay'
return scalar pvalue = r(p)

end
```

“TABLE1B” program

```
// This program takes approximately 30 mins to run on my laptop.

// The programs must be run in the following order: (i) hurwicz1A, then (ii) hurwicz1B.

// The program “etime” is a user-written .do file that can be obtained online

etime, start
drop _all
clear
graph drop _all
set more off
set seed 13
matrix meanbhaty = J(5,5,0)
matrix meanRR = J(5,5,0)

local i = 1
foreach betay in 0.1 0.5 0.9 0.95 0.99 {
  local j = 1
  foreach smallobs in 15 30 50 100 1000 {
    simulate bhaty = r(bhaty) pvalue = r(pvalue), ///
       reps(10000): hurwicz, betay(`betay') smallobs(`smallobs') bigobs(1100) ///
                    beta0(1) betau(1)
    summ bhaty, meanonly
    matrix meanbhaty[`i', `j'] = r(mean)
            generate RejectRate = 0
    replace RejectRate = 1 if pvalue < 0.05
            summ RejectRate, meanonly
    matrix meanRR[`i', `j'] = r(mean)
    local `++j'
  }
  local `++i'
}
matrix colnames meanbhaty = T15 T30 T50 T100 T1000
matrix rownames meanbhaty= B10 B50 B90 B95 B99
matrix colnames meanRR = T15 T30 T50 T100 T1000
matrix rownames meanRR = B10 B50 B90 B95 B99
matrix list meanbhaty
matrix list meanRR
etime
```

“TABLE2A” program

```
program drop _all
program define ARDLprog, rclass
version 13
syntax, beta0(real) betay(real) betax(real) betau(real) smallobs(integer) bigobs(integer)
// Remove existing variables
drop _all

// Create the data
set obs `bigobs'
gen t = _n
tsset t
// We initialize y as its LR equilibrium value
gen x=rnormal()
gen u=rnormal()
gen y = `beta0'/(1-`betay')
replace y = `beta0' + `betax'*x+`betay'*L.y + u in 2/l
regress y L.y x in -`smallobs'/l
return scalar LRP = _b[x]/(1-_b[L.y])
testnl _b[x]/(1-_b[L.y])=`betax'/(1-`betay')
return scalar pLRP = r(p)

end
```

“TABLE2B” program

```
// This program takes approximately 20 mins to run on my laptop.

// The programs must be run in the following order: (i) hurwicz2A, then (ii) hurwicz2B.

// The program “etime” is a user-written .do file that can be obtained online

etime, start
drop _all
clear
graph drop _all
set more off
set seed 13
matrix medLRP = J(5,3,0)
matrix confidLRP = J(5,6,0)
matrix meanRR = J(5,3,0)

local i = 1
foreach betay in 0.1 0.5 0.9 0.95 0.99 {
  local j = 1
  foreach smallobs in 10 50 1000 {
    simulate LRP = r(LRP) pLRP = r(pLRP), ///
      reps(10000): ARDLprog, betay(`betay') smallobs(`smallobs') bigobs(1100) ///
                      beta0(1) betau(1) betax(1)
    summ LRP, detail
    matrix medLRP[`i', `j'] = r(p50)
    matrix confidLRP[`i', (`j'-1)*2+1] = r(p5)
    matrix confidLRP[`i', (`j'-1)*2+2] = r(p95)
            generate RejectRate = 0
    replace RejectRate = 1 if pLRP < 0.05
            summ RejectRate, meanonly
    matrix meanRR[`i', `j'] = r(mean)
    local `++j'
  }
  local `++i'
}
matrix colnames medLRP = "T10(P50)" "T50(P50)" "T1000(P50)"
matrix rownames medLRP= B10 B50 B90 B95 B99
matrix colnames confidLRP = "T10(P5)" "T10(P95)" "T50(P5)" "T50(P95)" ///
                   "T1000(P5)" "T1000(P95)"
matrix rownames confidLRP= B10 B50 B90 B95 B99
matrix colnames meanRR = T10 T50 T1000
matrix rownames meanRR = B10 B50 B90 B95 B99
matrix list medLRP
matrix list confidLRP
matrix list meanRR

etime
```

“TABLE3A” program

```
program drop _all
program define DPDprog, rclass
version 13
syntax, beta0(real) betax(real) betay(real) numN(integer) numT(integer) numNT(integer) ///
      beffect(real)
drop _all
set obs `numN'
gen id = _n
// "ai" is the part of the error term that doesn't change over time for a given
// unit.
gen ai = rnormal()
expand `numT'
bysort id: gen t=_n
xtset id t
gen x = (`beffect'*ai + rnormal())/sqrt(1+`beffect'^2)
// "uit" is the part of the error term that varies across time
gen uit = rnormal()
// The total error term is the sum of "a" and "e"
gen error = ai + uit
gen y = `beta0'/(1-`betay')
replace y = `beta0' + `betax'*x + `betay'*L.y + error if t > 1


//This section calculates AH estimates
xtivreg y x (L.y = L(2/3).y) if t > 10, fd
return scalar AHLRP = _b[D.x]/(1-_b[LD.y])
testnl _b[D.x]/(1-_b[LD.y])=`betax'/(1-`betay')
return scalar pAHLRP = r(p)

// This section calculates Difference GMM
xtabond y x if t > 10
return scalar DGMMLRP = _b[x]/(1-_b[L.y])
testnl _b[x]/(1-_b[L.y])=`betax'/(1-`betay')
return scalar pDGMMLRP = r(p)

// This section calculates System GMM
xtdpdsys  y x if t > 10
return scalar SGMMLRP = _b[x]/(1-_b[L.y])
testnl _b[x]/(1-_b[L.y])=`betax'/(1-`betay')
return scalar pSGMMLRP = r(p)

end
```

“TABLE3B” program

```
// This program takes approximately 70 mins to run on my laptop.

// The programs must be run in the following order: (i) hurwicz3A, then (ii) hurwicz3B.

// The program “etime” is a user-written .do file that can be obtained online


etime, start
drop _all
clear
set more off
set seed 13
matrix medLRP = J(5,3,0)
matrix confidLRP = J(5,6,0)
matrix meanRR = J(5,3,0)
// The local commands below set all the parameters for the experiments.
local beffect = 0
// “beffect” controls the degree of correlation between x and the fixed effect.
// When beffect = 0, there is no omitted variable bias.  As beffect increases,
// omitted variable bias increases.
local numN = 50 // This sets the number of cross-sectional units
local numT = 20 // This sets the number of time observations per unit
// Note that the estimations will only use the last 10 of these observations
local numNT = `numN'*`numT' // This sets the total number of observations
local beta0 = 1 // This sets the intercept term
local betax = 1 // This sets the slope coefficient for x
local reps = 10000

local i = 1
foreach betay in 0.1 0.5 0.9 0.95 0.99 {
    simulate AHLRP = r(AHLRP) pAHLRP = r(pAHLRP) DGMMLRP = r(DGMMLRP) ///
    pDGMMLRP = r(pDGMMLRP) SGMMLRP = r(SGMMLRP) pSGMMLRP = r(pSGMMLRP) , ///
    reps(`reps'): DPDprog, betay(`betay') betax(`betax') beta0(`beta0') ///
    numN(`numN') numT(`numT') numNT(`numNT') beffect(`beffect')

    summ AHLRP, detail
    matrix medLRP[`i',1] = r(p50)
    matrix confidLRP[`i',1] = r(p5)
    matrix confidLRP[`i',2] = r(p95)
    generate RRAHLRP = 0
    replace RRAHLRP = 1 if pAHLRP < 0.05
    summ RRAHLRP, meanonly
    matrix meanRR[`i', 1] = r(mean)

    summ DGMMLRP, detail
    matrix medLRP[`i',2] = r(p50)
    matrix confidLRP[`i',3] = r(p5)
    matrix confidLRP[`i',4] = r(p95)
    generate RRDGMMLRP = 0
    replace RRDGMMLRP = 1 if pDGMMLRP < 0.05
    summ RRDGMMLRP, meanonly
    matrix meanRR[`i', 2] = r(mean)
    summ SGMMLRP, detail
    matrix medLRP[`i',3] = r(p50)
    matrix confidLRP[`i',5] = r(p5)
    matrix confidLRP[`i',6] = r(p95)
    generate RRSGMMLRP = 0
    replace RRSGMMLRP = 1 if pSGMMLRP < 0.05
    summ RRSGMMLRP, meanonly
    matrix meanRR[`i', 3] = r(mean)

    local `++i'
}

// These commands print out the results
matrix colnames medLRP = AHLRP(p50) DGMMLRP(p50) SGMMLRP(p50)
matrix rownames medLRP = B10 B50 B90 B95 B99

matrix colnames confidLRP = AHLRP(p5) AHLRP(p95) DGMMLRP(p5) DGMMLRP(p95) ///
SGMMLRP(p5) SGMMLRP(p95)

matrix rownames confidLRP = B10 B50 B90 B95 B99
matrix colnames meanRR = AHLRP DGMMLRP  SGMMLRP
matrix rownames meanRR = B10 B50 B90 B95 B99
matrix list medLRP
matrix list confidLRP
matrix list meanRR

etime
```

“TABLE4A” program

```
program drop _all
program define DPDprog, rclass
version 13

syntax, beta0(real) betax(real) betay(real) numN(integer) numT(integer) numNT(integer) ///
      beffect(real)

drop _all
set obs `numN'
gen id = _n
gen ai = rnormal()
// "ai" is the part of the error term that doesn't change over time for a given unit.
expand `numT'
bysort id: gen t=_n
xtset id t
gen x = (`beffect'*ai + rnormal())/sqrt(1+`beffect'^2)
gen uit = rnormal()
// "uit" is the part of the error term that varies across time
// The total error term is the sum of "a" and "e"
gen error = ai + uit
gen y = `beta0'/(1-`betay')
replace y = `beta0' + `betax'*x + `betay'*L.y + error if t > 1


//This section calculates AH estimates
xtivreg y x (L.y = L(2/3).y) if t > 15, fd
return scalar AHLRP = _b[D.x]/(1-_b[LD.y])
testnl _b[D.x]/(1-_b[LD.y])=`betax'/(1-`betay')
return scalar pAHLRP = r(p)

// This section calculates Difference GMM
xtabond y x if t > 15
return scalar DGMMLRP = _b[x]/(1-_b[L.y])
testnl _b[x]/(1-_b[L.y])=`betax'/(1-`betay')
return scalar pDGMMLRP = r(p)

// This section calculates System GMM
xtdpdsys  y x if t > 15
return scalar SGMMLRP = _b[x]/(1-_b[L.y])
testnl _b[x]/(1-_b[L.y])=`betax'/(1-`betay')
return scalar pSGMMLRP = r(p)

end
```

"TABLE4B" program

```
// This program takes approximately 100 mins to run on my laptop.

// The programs must be run in the following order: (i) hurwicz3A, then (ii) hurwicz3B.

// The program "etime" is a user-written .do file that can be obtained online

etime, start
drop _all
clear
set more off
set seed 13
matrix medLRP = J(5,3,0)
matrix confidLRP = J(5,6,0)
matrix meanRR = J(5,3,0)
// The local commands below set all the parameters for the experiments.
local beffect = 0
// "beffect" controls the degree of correlation between x and the fixed effect.
// When beffect = 0, there is no omitted variable bias.  As beffect increases,
// omitted variable bias increases.
local numN = 140 // This sets the number of cross-sectional units
local numT = 20 // This sets the number of time observations per unit
// Note that the estimations will only use the last 5 of these observations
local numNT = `numN'*`numT' // This sets the total number of observations
local beta0 = 1 // This sets the intercept term
local betax = 1 // This sets the slope coefficient for x
local reps = 10000

local i = 1
foreach betay in 0.1 0.5 0.9 0.95 0.99 {
   simulate AHLRP = r(AHLRP) pAHLRP = r(pAHLRP) DGMMLRP = r(DGMMLRP) ///
   pDGMMLRP = r(pDGMMLRP) SGMMLRP = r(SGMMLRP) pSGMMLRP = r(pSGMMLRP) , ///
   reps(`reps'): DPDprog, betay(`betay') betax(`betax') beta0(`beta0') ///
   numN(`numN') numT(`numT') numNT(`numNT') beffect(`beffect')

   summ AHLRP, detail
   matrix medLRP[`i',1] = r(p50)
   matrix confidLRP[`i',1] = r(p5)
   matrix confidLRP[`i',2] = r(p95)
   generate RRAHLRP = 0
   replace RRAHLRP = 1 if pAHLRP < 0.05
   summ RRAHLRP, meanonly
   matrix meanRR[`i', 1] = r(mean)

   summ DGMMLRP, detail
   matrix medLRP[`i',2] = r(p50)
   matrix confidLRP[`i',3] = r(p5)
   matrix confidLRP[`i',4] = r(p95)
   generate RRDGMMLRP = 0
   replace RRDGMMLRP = 1 if pDGMMLRP < 0.05
   summ RRDGMMLRP, meanonly
   matrix meanRR[`i', 2] = r(mean)
```

```
    summ SGMMLRP, detail
    matrix medLRP[`i',3] = r(p50)
    matrix confidLRP[`i',5] = r(p5)
    matrix confidLRP[`i',6] = r(p95)
    generate RRSGMMLRP = 0
    replace RRSGMMLRP = 1 if pSGMMLRP < 0.05
    summ RRSGMMLRP, meanonly
    matrix meanRR[`i', 3] = r(mean)

    local `++i'
}

// These commands print out the results
matrix colnames medLRP = AHLRP(p50) DGMMLRP(p50) SGMMLRP(p50)
matrix rownames medLRP = B10 B50 B90 B95 B99

matrix colnames confidLRP = AHLRP(p5) AHLRP(p95) DGMMLRP(p5) DGMMLRP(p95) ///
SGMMLRP(p5) SGMMLRP(p95)

matrix rownames confidLRP = B10 B50 B90 B95 B99
matrix colnames meanRR = AHLRP DGMMLRP  SGMMLRP
matrix rownames meanRR = B10 B50 B90 B95 B99
matrix list medLRP
matrix list confidLRP
matrix list meanRR

etime
```