Visualizing Regression Results in *Stata*

Andrew Taeho Kim

Population Studies Center
University of Pennsylvania

Digital Korea Workshop
April 26, 2024

Penn
Population Studies Center

Why Visualize Quantitative Results?

- Whether in a paper or presentation, you want to be:
  - Seen
  - Attended to
  - Understood

Why Visualize Regression Results?

1. Visualization improves your ability to quickly and effectively communicate your research
2. Visualization simplifies complex relationships you want your audience to understand
3. Visualization makes your presentation more interesting and enjoyable for your audience

How many threes do you see?

126954852361235698745824 5
012403698570206956831278 1
243986201247813698217325 6

How about now?

126954852**3**612**3**56987458245
012403698570206956**3**12781
24**3**9862012478**3**6982173256

Other Benefits

- Instant gratification
- No good arguments can be formulated in a day
- Graphs can be improved with little time

Why *Stata*?

- Unlike object oriented languages commonly used in quantitaive analysis, `R` or `Python`, *Stata* is a proprietary software

- Typical work flow of quantitative analysis using *Stata*
- How to plot coefficients?
- How do we plot changes in coefficients over time?

Necessary Packages

```
. ssc install estout
. ssc install estwrite
. ssc install coefplot
```

Data

- Data

```
. use "../workingdata/wrkdt", clear
. desc
Contains data from ../workingdata/wrkdt.dta
 Observations:     1,086,199
    Variables:             9                  25 Apr 2024 18:38
```

Variable name	Storage type	Display format	Value label	Variable label
year	int	%9.0g		year
wt	float	%9.0g		survey weight
fem	float	%9.0g	fem	women
mst	float	%9.0g	mst	marital status
red	float	%9.0g	red	level of education
rra	float	%9.0g	rra	race/ethnicity
baa	float	%9.0g		bachelor´s degree
age	float	%9.0g		age
wage	float	%9.0g		log hourly wage

```
Sorted by:
```

Data

- Variables

```
. sum [aw=wt]
```

Variable	Obs	Weight	Mean	Std. dev.	Min	Max
year	1086199	1.6955e+09	2009.586	5.736961	2000	2019
wt	1086199	1.6955e+09	2438.349	1799.55	25.33	31392.52
fem	1086199	1.6955e+09	.4844167	.4997573	0	1
mst	1086199	1.6955e+09	2.700341	.9614301	1	4
red	1086199	1.6955e+09	3.140196	1.194407	1	7
rra	1086199	1.6955e+09	1.490462	1.012816	1	5
baa	1086199	1.6955e+09	.3623915	.4806913	0	1
age	1086199	1.6955e+09	39.41675	8.681732	25	54
wage	1086199	1.6955e+09	3.064864	.7094721	0	13.08339

Macros

Typing all

```
. reg wage i.fem c.age##c.age i.red i.rra i.mst i.year [pw=wt]
. eststo m1
```

Set up globals & locals

```
. global dv wage
. local ctr1 c.age##c.age i.red i.rra i.mst i.year
.
. reg $dv i.fem `ctr1´ [pw=wt]
. eststo m1
```

- *global* can be called again as long as you don't exit out of Stata
- *local* needs to be executed with the part of the code you call on it
- *eststo* stores the regression result on memory until you exit out of Stata

How information is stored

```
. ereturn list
scalars:
                  e(rank) =  36
                  e(ll_0) =  -1168428.630568378
                    e(ll) =  -1028861.83358448
                  e(r2_a) =  .2265929461988193
                   e(rss) =  422837.817660327
                   e(mss) =  123900.7217761002
                  e(rmse) =  .6239350873002019
                    e(r2) =  .2266178672968908
                     e(F) =  5487.438106568069
                  e(df_r) =  1086163
                  e(df_m) =  35
                     e(N) =  1086199
macros:
             e(cmdline) : "regress wage i.fem c.age##c.age i.red i.rra i.mst i.year [pw=wt]"
               e(title) : "Linear regression"
          e(marginsok) : "XB default"
                e(vce) : "robust"
             e(depvar) : "wage"
                e(cmd) : "regress"
          e(properties) : "b V"
             e(predict) : "regres_p"
               e(model) : "ols"
           e(estat_cmd) : "regress_estat"
```

How coefficient is stored

```
. mat list e(b)
e(b)[1,41]
            0b.          1.                   c.age#          1b.          2.          3.          4.          5.
            fem          fem         age      c.age           red          red         red         red         red
y1            0  -.26522467  .05179767  -.00052695            0   .24207552   .40858491   .73581507   .89066307
            6.           7.          1b.          2.          3.          4.          5.          1b.          2.
           red          red         rra          rra         rra         rra         rra         mst          mst
y1   1.1882831    1.0815534           0   -.1065094   .04696405   -.0444498   -.09027914           0   .06936895
            3.           4.       2000b.       2001.       2002.       2003.       2004.       2005.       2006.
           mst          mst        year         year        year        year        year        year        year
y1   .16792693   .05710176           0   .02955896   .04752433   .04599905   .0379516   .02273897   .0141797
         2007.        2008.       2009.       2010.       2011.       2012.       2013.       2014.       2015.
          year         year        year        year        year        year        year        year        year
y1   .02004731   .01102703   .03243765   .03199826   .00129681   -.00657934  -.01656676  -.01740182   .00105062
         2016.        2017.       2018.       2019.
          year         year        year        year       _cons
y1   .01961571   .02690269   .02966864   .03976278   1.4136592
```

Grab coefficient

```
. di _b[1.fem]
-.26522467
```

Utilize loops #1

Different models

```
. local ctr1 c.age##c.age i.rra i.mst i.year
. local ctr2 c.age##c.age i.rra i.mst i.year i.red
.
. reg $dv i.fem `ctr1´ [pw=wt]
. eststo m1
.
. reg $dv i.fem `ctr2´ [pw=wt]
. eststo m2
```

Loop over models

```
. local ctr1 c.age##c.age i.rra i.mst i.year
. local ctr2 c.age##c.age i.rra i.mst i.year i.red
.
. foreach md in 1 2 {
.   reg $dv i.fem `ctr`md´´ [pw=wt]
.   eststo m`md´
. }
```

Results

```
. esttab m1 m2, ///
>   mtitle(m1 m2) ///
>   b(3) se(3) r2(3) ar2(3) keep(*.fem) lab
```

	(1) m1	(2) m2
men	0.000	0.000
	(.)	(.)
women	-0.228***	-0.265***
	(0.002)	(0.002)
Observations	1086199	1086199
R-squared	0.089	0.227
Adjusted R-squared	0.089	0.227

Standard errors in parentheses
* p<0.05, ** p<0.01, *** p<0.001

Utilize loops #2

Same model for multiple years

```
. local ctr3 c.age##c.age i.rra i.mst
.
. reg $dv i.fem `ctr3´ [pw=wt] if year == 2000
. eststo m3_y2000
.
. reg $dv i.fem `ctr3´ [pw=wt] if year == 2010
. eststo m3_y2010
.
. reg $dv i.fem `ctr3´ [pw=wt] if year == 2019
. eststo m3_y2019
```

Loop same model over years

```
. local ctr3 c.age##c.age i.rra i.mst
.
. foreach y in 2000 2010 2019 {
.   reg $dv i.fem `ctr3´ [pw=wt] if year == `y´
.   eststo m3_y`y´
. }
```

Results

```
. esttab m3_y2000 m3_y2010 m3_y2019, ///
>   mtitle(y2000 y2010 y2019) ///
>   b(3) se(3) r2(3) ar2(3) keep(*.fem) lab
```

	(1) y2000	(2) y2010	(3) y2019
men	0.000	0.000	0.000
	(.)	(.)	(.)
women	-0.292***	-0.231***	-0.183***
	(0.008)	(0.007)	(0.008)
Observations	40023	56033	44917
R-squared	0.093	0.085	0.099
Adjusted R-squared	0.093	0.085	0.098

Standard errors in parentheses
* p<0.05, ** p<0.01, *** p<0.001

Loops can be nested

Loop over each model & each year

```
. local ctr3 c.age##c.age i.rra i.mst
. local ctr4 c.age##c.age i.rra i.mst i.red
.
. foreach md in 3 4 {
.   foreach y in 2000 2010 2019 {
.     reg $dv i.fem `ctr`md´´ [pw=wt] if year == `y´
.     eststo m`md´_y`y´
.   }
. }
```

- above codes run 6 regressions (3 years $\times$ 2 models)

What if there are lots of years?

Put values of variable *year* into local *yrs* and loop over each

```
. levelsof year, local(yrs)
. local ctr3 c.age##c.age i.rra i.mst
.
. foreach y in `yrs´ {
.   reg $dv i.fem `ctr3´ [pw=wt] if year == `y´
.   eststo m3_y`y´
. }
```

Same results, different way to loop

```
. levelsof year, local(yrs)
. local yrsn : word count `yrs´
. local ctr3 c.age##c.age i.rra i.mst
.
. forvalues i = 1(1)`yrsn´ {
.   local y : word `i´ of `yrs´
.
.   reg $dv i.fem `ctr3´ [pw=wt] if year == `y´
.   eststo m3_y`y´
. }
```

Loop over models and years

```
. levelsof year, local(yrs)
. local yrsn : word count `yrs´
. local ctr3 c.age##c.age i.rra i.mst
. local ctr4 c.age##c.age i.rra i.mst i.red
.
. foreach md in 3 4 {
.   forvalues i = 1(1)`yrsn´ {
.         local y : word `i´ of `yrs´
.     reg $dv i.fem `ctr`md´´ [pw=wt] if year == `y´
.     eststo m`md´_y`y´
.   }
. }
```

Save results

Save all stored estimates through *eststo*

```
. estwrite * using "../estimates/results", replace
```

Save the results with names that starts with m4

```
. estwrite m4_* using "../estimates/results_m4", replace
```

- do this so we can use stored regression results later without having to run everything again

Load/Read

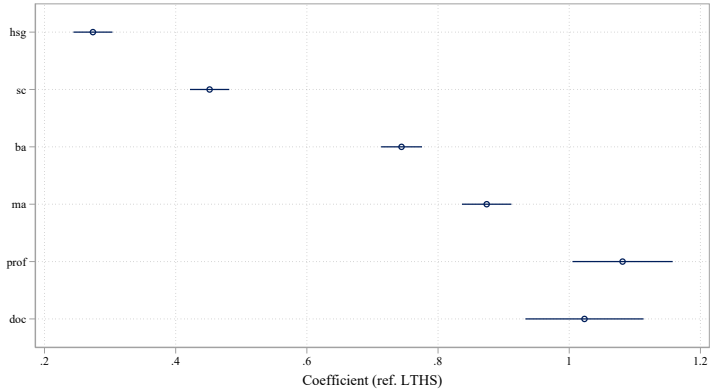
Load all estimates

```
. estread using "../estimates/results"
```

- if you already have an estimate stored with *eststo* and it happens to have the same name as one of the estimates in the file you just loaded, it will be overwritten

coefplot: typical usage

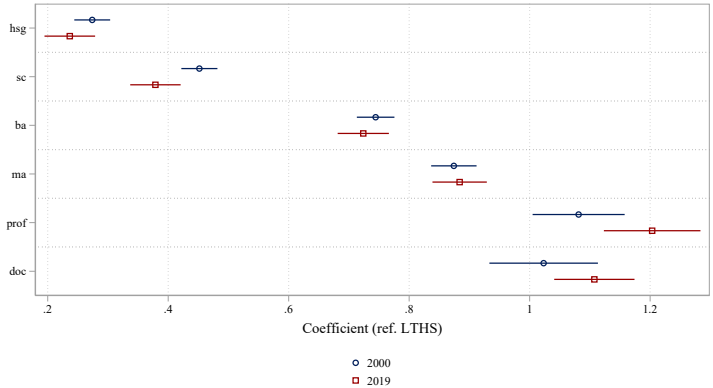
```
. coefplot m4_y2000, ///
>   keep(*.red) lab xtitle("Coefficient (ref. LTHS)") xsize(7) ysize(4) name(fig1, replace)
```



- returns to education in year 2000

coefplot: typical usage, multiple years

```
. coefplot m4_y2000 m4_y2019, ///
>   keep(*.red) lab xtitle("Coefficient (ref. LTHS)") xsize(7) ysize(4) name(fig2, replace) ///
>   legend(label(2 "2000") label(4 "2019"))
```



- returns to education in year 2000 and 2019

Plot coefficient for each year (2000 to 2019), Prep

- Create a matrix with a name *m3*
  - # of rows = number of years
  - # of columns = 4 : year, coefficient, CI lower, CI upper

```
. levelsof year, local(yrs)
. local row : word count `yrs´
. local col = 4
.
. mat define m3 = J(`row´, `col´, .)
. mat colnames m3 = year b ci_l ci_u
```

Plot coefficient for each year (2000 to 2019), Prep

```
. mat list m3
m3[20,4]
     year     b  ci_l  ci_u
 r1     .     .     .     .
 r2     .     .     .     .
 r3     .     .     .     .
 r4     .     .     .     .
 r5     .     .     .     .
 r6     .     .     .     .
 r7     .     .     .     .
 r8     .     .     .     .
 r9     .     .     .     .
r10     .     .     .     .
r11     .     .     .     .
r12     .     .     .     .
r13     .     .     .     .
r14     .     .     .     .
r15     .     .     .     .
r16     .     .     .     .
r17     .     .     .     .
r18     .     .     .     .
r19     .     .     .     .
r20     .     .     .     .
```

Plot coefficient for each year (2000 to 2019), Prep

```
. levelsof year, local(yrs)
. local row : word count `yrs´
.
. forvalues i = 1(1)`row´ {
.         local y : word `i´ of `yrs´
.
.         est res m3_y`y´
.
.         mat m3[`i´, 1] = `y´
.         mat m3[`i´, 2] = _b[1.fem]
.         mat m3[`i´, 3] = _b[1.fem] - invttail(e(df_r),0.025)*_se[1.fem]
.         mat m3[`i´, 4] = _b[1.fem] + invttail(e(df_r),0.025)*_se[1.fem]
. }
```

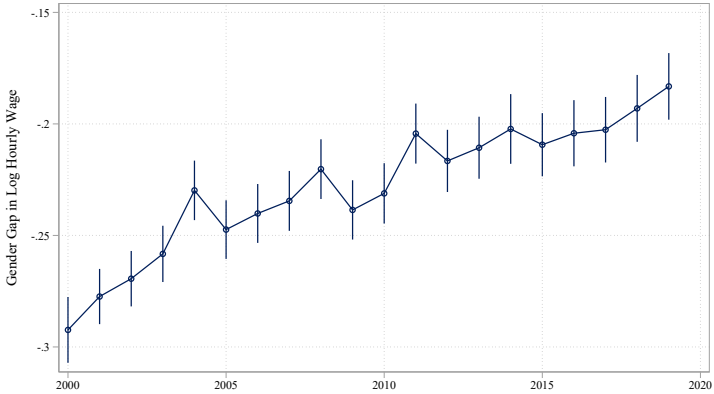
- *est res* call estimation result to memory
- model 3 : m3
  - _b[1.fem] : coefficient for *fem*
  - _se[1.fem] : standard error for *fem*

Plot coefficient gap for each year (2000 to 2019), Prep

```
. mat list m3
m3[20,4]
           year           b        ci_l        ci_u
 r1        2000  -.29235787  -.30711459  -.27760114
 r2        2001  -.27739282  -.28976644   -.2650192
 r3        2002  -.26939411  -.28182055  -.25696768
 r4        2003  -.25827349  -.27088947   -.2456575
 r5        2004  -.22979475  -.24313102  -.21645848
 r6        2005  -.24738287  -.26052561  -.23424012
 r7        2006  -.24014884  -.25335165  -.22694602
 r8        2007  -.23450857  -.24793353  -.22108362
 r9        2008  -.22027155  -.23365661   -.2068865
r10        2009  -.23855376  -.25183987  -.22526765
r11        2010  -.23114819  -.24468867  -.21760771
r12        2011  -.20433722  -.21778939  -.19088504
r13        2012  -.21658869  -.23053071  -.20264666
r14        2013  -.21068068  -.22458691  -.19677446
r15        2014  -.20226432   -.2178909  -.18663774
r16        2015  -.20931961  -.22348404  -.19515519
r17        2016  -.20417878  -.21901576   -.1893418
r18        2017  -.20260159  -.21729543  -.18790775
r19        2018   -.1930324  -.20802629   -.1780385
r20        2019  -.18317506  -.19810839  -.16824173
```

Plot coefficient for each year (2000 to 2019)

```
. coefplot (matrix(m3[, 2]), ci((m3[, 3] m3[, 4])) ), at(matrix(m3[, 1])) ///
>   vert recast(connected) ytitle("Gender Gap in Log Hourly Wage") xsize(7) ysize(4) ///
>   name(fig3, replace)
```



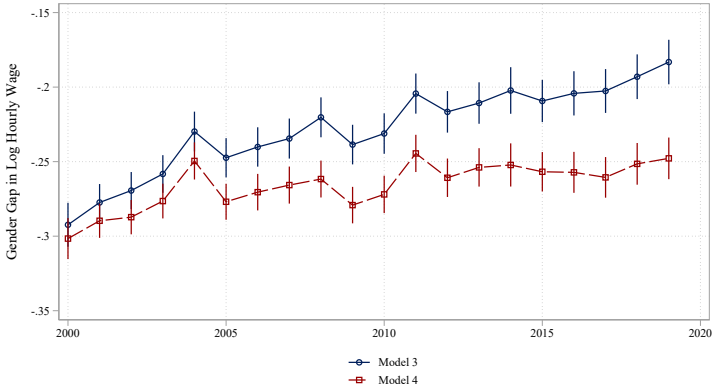
Plot multiple models

- create a matrix for model 4 results on gender gap

```
. levelsof year, local(yrs)
. local row : word count `yrs´
. local col = 4
.
. mat define m4 = J(`row´, `col´, .)
. mat colnames m4 = year b ci_l ci_u
.
. forvalues i = 1(1)`row´ {
.         local y : word `i´ of `yrs´
.
.         est res m4_y`y´
.
.         mat m4[`i´, 1] = `y´
.         mat m4[`i´, 2] = _b[1.fem]
.         mat m4[`i´, 3] = _b[1.fem] - invttail(e(df_r),0.025)*_se[1.fem]
.         mat m4[`i´, 4] = _b[1.fem] + invttail(e(df_r),0.025)*_se[1.fem]
. }
```

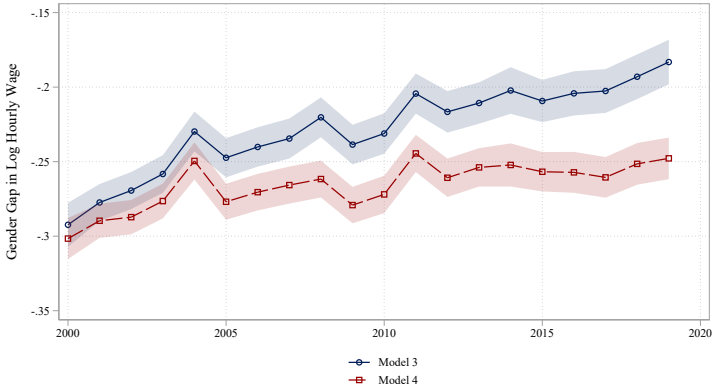
Plot coefficient for each year (2000 to 2019), m3 & m4

```
. coefplot (matrix(m3[, 2]), ci((m3[, 3] m3[, 4])) ) (matrix(m4[, 2]), ci((m4[, 3] m4[, 4])) ), ///
>   at(matrix(m3[, 1])) ///
>   vert recast(connected) ytitle("Gender Gap in Log Hourly Wage") ///
>   legend(label(2 "Model 3") label(4 "Model 4")) xsize(7) ysize(4) ///
>   name(fig4, replace)
```



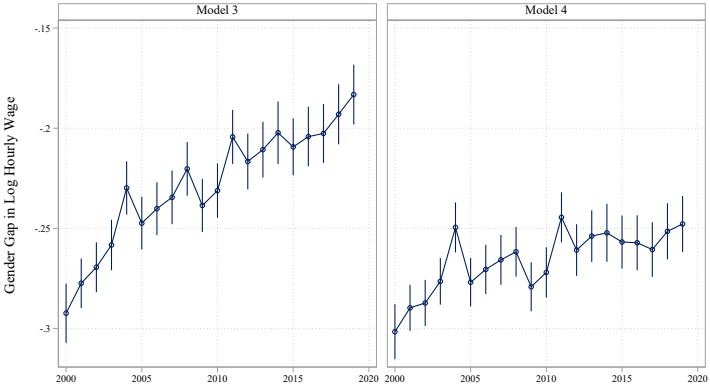
Plot coefficient for each year (2000 to 2019), m3 & m4

```
. coefplot (matrix(m3[, 2]), ci((m3[, 3] m3[, 4])) ) (matrix(m4[, 2]), ci((m4[, 3] m4[, 4])) ), ///
>   at(matrix(m3[, 1])) ciopts(recast(rarea) fcolor(%20) lcolor(%0)) ///
>   vert recast(connected) ytitle("Gender Gap in Log Hourly Wage") ///
>   legend(label(2 "Model 3") label(4 "Model 4")) xsize(7) ysize(4) ///
>   name(fig5, replace)
```



Plot coefficient for each year (2000 to 2019), m3 & m4

```
. coefplot (matrix(m3[, 2]), ci((m3[, 3] m3[, 4])) ), bylabel("Model 3") ///
>   || (matrix(m4[, 2]), ci((m4[, 3] m4[, 4])) ), bylabel("Model 4") ///
>   at(matrix(m3[, 1])) vert recast(connected) ytitle("Gender Gap in Log Hourly Wage") ///
>   byopts(rows(1)) xsize(7) ysize(4) ///
>   name(fig6, replace)
```



Scheme

- *Stata* use graphic schemes
  - default is **s2color** as of version 17

```
. help scheme
```

- *Stata* graphic scheme with Penn colors

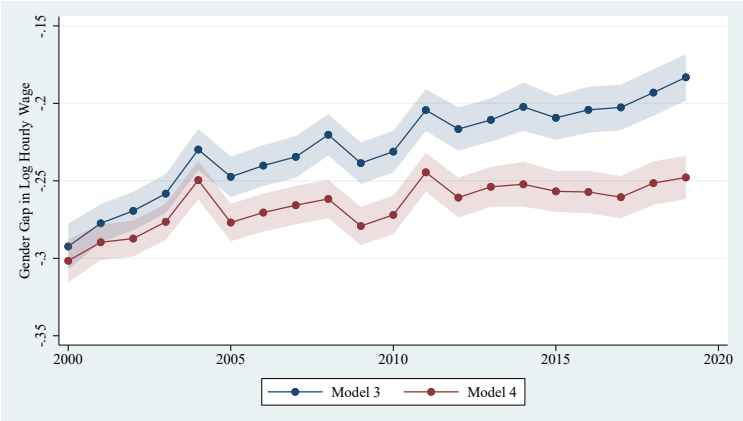
```
. net install plotUPenn, from("https://atkim1103.github.io/website/files/plotUPenn")
```

- `cleanplots`

```
. net install cleanplots, from("https://tdmize.github.io/data/cleanplots")
```

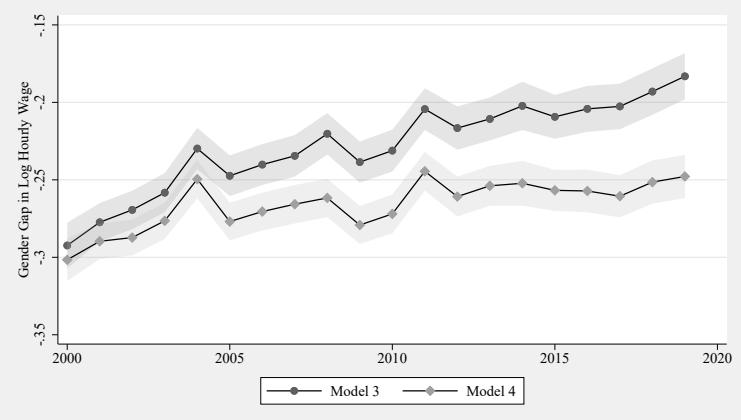
Scheme: s2color

```
. set scheme s2color
. coefplot (matrix(m3[, 2]), ci((m3[, 3] m3[, 4])) ) (matrix(m4[, 2]), ci((m4[, 3] m4[, 4])) ), ///
>   at(matrix(m3[, 1])) vert recast(connected) ytitle("Gender Gap in Log Hourly Wage") ///
>   legend(label(2 "Model 3") label(4 "Model 4")) ///
>   ciopts(recast(rarea) fcolor(%20) lcolor(%0)) xsize(7) ysize(4) name(fig7, replace)
```



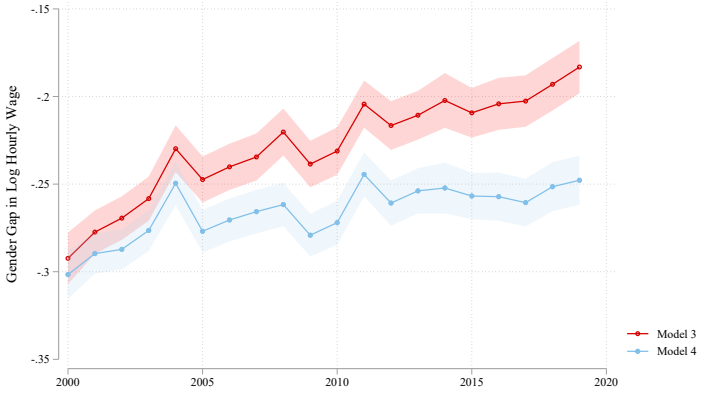
Scheme: sj → *Stata Journal*

```
. set scheme sj
. coefplot (matrix(m3[, 2]), ci((m3[, 3] m3[, 4])) ) (matrix(m4[, 2]), ci((m4[, 3] m4[, 4])) ), ///
>   at(matrix(m3[, 1])) vert recast(connected) ytitle("Gender Gap in Log Hourly Wage") ///
>   legend(label(2 "Model 3") label(4 "Model 4")) ///
>   ciopts(recast(rarea) fcolor(%20) lcolor(%0)) xsize(7) ysize(4) name(fig8, replace)
```



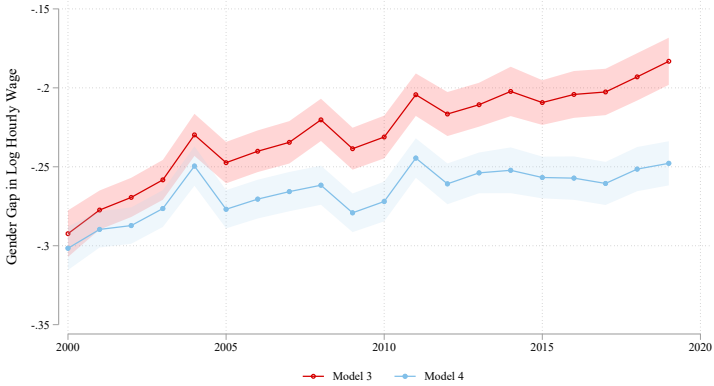
Scheme: cleanplots

```
. set scheme cleanplots
. coefplot (matrix(m3[, 2]), ci((m3[, 3] m3[, 4])) ) (matrix(m4[, 2]), ci((m4[, 3] m4[, 4])) ), ///
>   at(matrix(m3[, 1])) vert recast(connected) ytitle("Gender Gap in Log Hourly Wage") ///
>   legend(label(2 "Model 3") label(4 "Model 4")) ///
>   ciopts(recast(rarea) fcolor(%20) lcolor(%0)) xsize(7) ysize(4) name(fig9, replace)
```



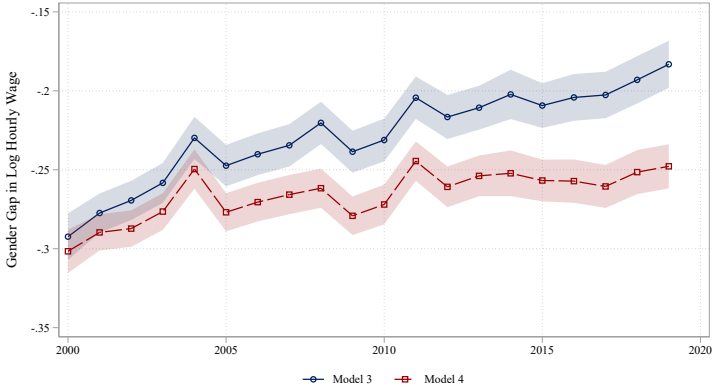
Scheme: cleanplots

```
. set scheme cleanplots
. coefplot (matrix(m3[, 2]), ci((m3[, 3] m3[, 4])) ) (matrix(m4[, 2]), ci((m4[, 3] m4[, 4])) ), ///
>   at(matrix(m3[, 1])) vert recast(connected) ytitle("Gender Gap in Log Hourly Wage") ///
>   legend(label(2 "Model 3") label(4 "Model 4") pos(6) rows(1)) ///
>   ciopts(recast(rarea) fcolor(%20) lcolor(%0)) xsize(7) ysize(4) name(fig10, replace)
```



Scheme: plotUPenn

```
. set scheme plotUPenn
. coefplot (matrix(m3[, 2]), ci((m3[, 3] m3[, 4])) ) (matrix(m4[, 2]), ci((m4[, 3] m4[, 4])) ), ///
>   at(matrix(m3[, 1])) vert recast(connected) ytitle("Gender Gap in Log Hourly Wage") ///
>   legend(label(2 "Model 3") label(4 "Model 4") pos(6) rows(1)) ///
>   ciopts(recast(rarea) fcolor(%20) lcolor(%0)) xsize(7) ysize(4) name(fig11, replace)
```



Saving Graphs

```
. forvalues i = 1(1)11 {
  2.         gr export "../table_figure/fig`i´.pdf", name(fig`i´) replace
  3. }
file ../table_figure/fig1.pdf saved as PDF format
file ../table_figure/fig2.pdf saved as PDF format
file ../table_figure/fig3.pdf saved as PDF format
file ../table_figure/fig4.pdf saved as PDF format
file ../table_figure/fig5.pdf saved as PDF format
file ../table_figure/fig6.pdf saved as PDF format
file ../table_figure/fig7.pdf saved as PDF format
file ../table_figure/fig8.pdf saved as PDF format
file ../table_figure/fig9.pdf saved as PDF format
file ../table_figure/fig10.pdf saved as PDF format
file ../table_figure/fig11.pdf saved as PDF format
```

Further Customizations

- *Stata* package `grstyle` by Benn Jann
  - Can do whatever you want
  - `https://repec.sowi.unibe.ch/stata/grstyle/index.html`

Put coefficient from multiple years into a single model

```
. prog appendmd, eclass
 1.         syntax namelist
 2.         tempname b V tmp
 3.         foreach name of local namelist {
 4.                 qui est restore `name´
 5.                 mat `b´ = nullmat(`b´) , e(b)
 6.                 mat `b´ = `b´[1,1..colsof(`b´)]
 7.                 mat `tmp´ = e(V)
 8.                 mat `tmp´ = `tmp´[1..rowsof(`tmp´),1..colsof(`tmp´)]
 9.                 capt confirm matrix `V´
10.                 if _rc {
11.                         mat `V´ = `tmp´
12.                 }
13.                 else {
14.                         mat `V´ = ///
>                                 ( `V´ , J(rowsof(`V´),colsof(`tmp´),0) ) \ ///
>                                 ( J(rowsof(`tmp´),colsof(`V´),0) , `tmp´ )
15.                 }
16.         }
17.         local names: colfullnames `b´
18.         mat coln `V´ = `names´
19.         mat rown `V´ = `names´
20.         eret post `b´ `V´
21.         eret local cmd "whatever"
22. end
```

Put coefficient from multiple years into a single model

```
. levelsof year, local(yrs)
.
. foreach y in `yrs´ {
.         est res m3_y`y´
.
.         local fem_coln = colnumb(e(b), "1.fem")
.         local fem_rown = colnumb(e(V), "1.fem")
.
.         mat b = e(b)[1, `fem_coln´]
.         mat v = e(V)[`fem_rown´, `fem_coln´]
.
.         mat colnames b = "y`y´"
.         mat colnames v = "y`y´"
.         mat rownames v = "y`y´"
.
.         ereturn post b v
.         eststo y`y´
. }
```

- extract coefficient `1.fem` from models on each year and save it as estimate name y[year]

Put coefficient from multiple years into a single model

```
. levelsof year, local(yrs)
. local yrsn : word count `yrs´
.
. local mods "y2000"
. forvalues i = 2(1)`yrsn´ {
.         local y : word `i´ of `yrs´
.         local mods "`mods´ y`y´"
. }
.
. appendmd `mods´
. eststo m3_fem
```

- combine y[year] estimates with only `1.fem` into a single model

Plot the combined estimation

```
. coefplot m3_fem, ///
>         vert recast(connected) ytitle("Gender Gap in Log Hourly Wage") ///
>         xlabel(1 "2000" 6 "2005" 11 "2010" 16 "2015" 21 "2020", grid) ///
>         xsize(7) ysize(4) name(fig12, replace)
```