Time Trends and FEs

Drew Dimmery drewd@nyu.edu

March 14, 2014

Structure

- Quick Homework Talk
- Abadie's κ
- Fixed effects
- Time Trends

On CIA

- There seemed to be some confusion on CIA.
- The "traditional" method of choosing a covariate set is something like:
  - Find the most significant or strongest predictors that you have
  - Include those
  - Don't include the weak predictors
  - Include things that other studies have included
- This isn't what we want, though.
- CIA is a story about conditional randomization.
- You don't justify based on raw correlations.
- Because independence implies zero correlation
- But not vice versa
- Instead, you should think about where you might be seeing random variation, and then adjust your conditioning strategy to ensure that you only work with that variation.

More IV Stuff

- We're going to be looking at Ananat (2011) in AEJ

- This study looks at the effect of racial segregation on economic outcomes.
- Outcome: Poverty rate & Inequality (Gini index)
- Treatment: Segregation
- Instrument: “railroad division index”
- Main covariate of note: railroad length in a town
- I’m dichotomizing treatment and instrument for simplicity.
- And my outcomes are for the Black subsample

```
. . .
```

```
require(foreign)
d <- read.dta("aej_maindata.dta")
d$herf_b <- with(d, ifelse(herf >= quantile(herf, 0.5), 1, 0))
d$dism1990_b <- with(d, ifelse(dism1990 >= quantile(dism1990, 0.5), 1, 0))
first.stage <- lm(dism1990 ~ herf + lenper, d)
first.stage.b <- lm(dism1990_b ~ herf_b + lenper, d)
require(AER)
gini.iv <- ivreg(lngini_b ~ dism1990 + lenper, ~herf + lenper, d)
gini.iv.b <- ivreg(lngini_b ~ dism1990_b + lenper, ~herf_b + lenper, d)
pov.iv <- ivreg(povrate_b ~ dism1990 + lenper, ~herf + lenper, d)
pov.iv.b <- ivreg(povrate_b ~ dism1990_b + lenper, ~herf_b + lenper, d)
```

Base Results

```
round(summary(first.stage)$coefficients[2, ], 3)
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
##      0.357      0.081      4.395      0.000
```

```
round(summary(first.stage.b)$coefficients[2, ], 3)
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
##      0.372      0.083      4.481      0.000
```

```
round(summary(gini.iv)$coefficients[2, ], 3)
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
##      0.875      0.302      2.895      0.005
```

```
round(summary(gini.iv.b)$coefficients[2, ], 3)
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
##      0.211      0.081      2.615      0.010
```

```r
round(summary(pov.iv)$coefficients[2, ], 3)
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
##      0.258      0.144      1.798      0.075
```

```r
round(summary(pov.iv.b)$coefficients[2, ], 3)
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
##      0.059      0.039      1.543      0.125
```

Abadie's κ

- Recall from the lecture that we can use a weighting scheme to calculate statistics on the compliant population.
- $E[g(Y, D, X)|D_1 > D_0] = \frac{1}{p(D_1 > D_0)} E[\kappa g(Y, D, X)]$
- $\kappa = 1 - \frac{D_i(1-Z_i)}{p(Z_i=0|X)} - \frac{(1-D_i)Z_i}{p(Z_i=1|X)}$
- $E[\kappa|X] = E[D_1 - D_0|X] = E[D|X, Z = 1] - E[D|X, Z = 0]$
- Take $w_i = \frac{\kappa_i}{E[D_1 - D_0|X_i]}$
- Use this in calculating any interesting statistics (means, variance, etc)
- This let's you explore the units composing your LATE.

. . .

```r
getKappaWt <- function(D, Z) {
    pz <- mean(Z)
    pcomp <- mean(D[Z == 1]) - mean(D[Z == 0])
    if (pcomp < 0)
        stop("Assuming p(D|Z) > .5")
    kappa <- 1 - D * (1 - Z)/(1 - pz) - (1 - D) * Z/pz
    # Note that pcomp = mean(kappa)
    kappa/pcomp
}
w <- with(d, getKappaWt(D = dism1990_b, Z = herf_b))
varlist <- c("closeness", "area1910", "ctyliterate1920", "hsdrop_b", "manshr",
    "ctymanuf_wkrs1920", "ngov62")
samp.stats <- sapply(varlist, function(v) mean(d[, v], na.rm = TRUE))
comp.stats <- sapply(varlist, function(v) weighted.mean(d[, v], w, na.rm = TRUE))
```

Examine Complier Statistics

```
summary(w)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   -2.51   -2.43    2.47    1.00    2.47    2.47
```

```
rbind(sample = samp.stats, compliers = comp.stats)
```

```
##           closeness area1910 ctyliterate1920 hsdrop_b manshr
## sample       -362.4    14626          0.9585   0.2516 0.1892
## compliers    -299.1    18013          0.9515   0.2424 0.2110
##           ctymanuf_wkrs1920 ngov62
## sample               0.4619  55.55
## compliers            0.4266  83.65
```

New Example Data

- Now we're going to look at Gentzkow (2006)
- This study examined how the introduction of television had an effect on voter turnout
- Outcome: turnout in national elections
- Treatment: Years since the introduction of TV
- Identification comes from the idea that there are random variations in the timing of this introduction.
- It uses local variations in this introduction.
- Where locality is defined as a *within* effect.

Rough and Dirty Sweeping Function

- `plm` also does this, but I like to be able to use (most of) the functions written for `lm` objects.
- This assumes that you'll be clustering at the same level as your swept effects.

```
sweeplm <- function(formula, dat, ind) {
    newd <- model.matrix(formula, model.frame(formula, dat, na.action = na.pass))
    newd <- newd[, -1]
    newd <- cbind(dat[, as.character(formula[2])], newd)
    ok <- complete.cases(newd)
    newd <- newd[ok, ]
```

```
    ind <- ind[ok]
    newd <- apply(newd, 2, function(x) unlist(tapply(x, ind, function(z) z -
        mean(z, na.rm = TRUE))))
    list(lm(newd[, 1] ~ newd[, -1] - 1, as.data.frame(newd)), newd, as.character(ind))
}
```

Robust SEs again

```
robust.se <- function(model, cluster) {
    require(sandwich)
    require(lmtest)
    M <- length(unique(cluster))
    N <- length(cluster)
    K <- model$rank
    dfc <- (M/(M - 1)) * ((N - 1)/(N - K))
    uj <- apply(estfun(model), 2, function(x) tapply(x, cluster, sum))
    rcse.cov <- dfc * sandwich(model, meat. = crossprod(uj)/N)
    rcse.se <- coeftest(model, rcse.cov)
    return(rcse.se)
}
```

Fixed Effects

- Let's demonstrate some of the benefits of sweeping out fixed effects.
- Algebraic equivalence between these various methods.
  - De-mean variables by group
  - Apply the sweep matrix:
    - $\dot{Q} = (I_T - \frac{1}{T}\iota_T\iota_T') \otimes I_N$
    - $\beta = (X_{tv}'\dot{Q}X^{tv})^{-1}X_{tv}'\dot{Q}Y$
  - Just throw the dummies in the regression.

. . .

```
dat <- read.csv("gentzkow_fixed.csv")
dat$reg5year <- paste0(dat$regions5, dat$year)
dat <- subset(dat, year >= 1930)
base.lm <- lm(turnout ~ yearsoftv + advote, dat)
simple.fe <- lm(turnout ~ yearsoftv + advote + factor(reg5year), dat)
dummy.time <- system.time(lm(turnout ~ yearsoftv + advote + factor(reg5year),
    dat))
# all.fe <-
```

```r
# lm(turnout~yearsoftv+advote+factor(reg5year)+factor(stcounty),dat)
simple.swept.fe <- sweeplm(turnout ~ yearsoftv + advote, dat = dat, ind = dat$reg5year)
sweep.time <- system.time(sweeplm(turnout ~ yearsoftv + advote, dat = dat, ind = dat$reg5yea
swept.fe <- sweeplm(turnout ~ yearsoftv + advote + factor(reg5year), dat = dat,
    ind = dat$stcounty)
rbind(dummy.time, sweep.time)
```

```
##            user.self sys.self elapsed user.child sys.child
## dummy.time     1.025    0.107   1.160          0         0
## sweep.time     0.257    0.042   0.299          0         0
```

FE Results

```r
base.est <- robust.se(base.lm, dat$stcounty[-base.lm$na.action])[2, ]
```

```
## Note: no visible binding for global variable 'lmtest'
```

```r
simple.est <- robust.se(simple.fe, dat$stcounty[-simple.fe$na.action])[2, ]
simple.swept.est <- robust.se(simple.swept.fe[[1]], simple.swept.fe[[3]])[1,
    ]
swept.est <- robust.se(swept.fe[[1]], swept.fe[[3]])[1, ]
all.ests <- rbind(base.est, simple.est, simple.swept.est, swept.est)
rownames(all.ests) <- c("base", "simple", "simple.swept", "full.swept")
all.ests
```

```
##              Estimate Std. Error t value  Pr(>|t|)
## base           0.2475    0.01309  18.915 1.497e-79
## simple        -0.1729    0.07258  -2.382 1.722e-02
## simple.swept  -0.1729    0.03230  -5.353 8.706e-08
## full.swept    -0.3863    0.04987  -7.746 9.619e-15
```

Time Trends

- Instead of estimating fixed effects for each region-year, we could impose some structure.
- What if there is some unobserved process by which the outcome is *increasing* over time.
- Estimating fixed effects for each year may vastly reduce power (a new parameter for each time period under study)
- And if we are comfortable assuming this linear relationship with time, then we don't need to lose that power.

- Of course, we need only assume linearity in parameters, so we can include polynomials in time, too.
- Remember that you can perfectly interpolate N points with an $N-1$ degree polynomial.
- Think of it like fitting a polynomial *to the fixed effects.* It's sort of like that.

. . .

```
dat$trend <- dat$year - min(dat$year)
linear.trend <- sweeplm(turnout ~ yearsoftv + advote + trend, dat = dat, ind = dat$stcounty)
linear.reg.trend <- sweeplm(turnout ~ yearsoftv + advote + factor(regions5) *
    trend, dat = dat, ind = dat$stcounty)
cubic.trend <- sweeplm(turnout ~ yearsoftv + advote + poly(trend, 3), dat = dat,
    ind = dat$stcounty)
cubic.reg.trend <- sweeplm(turnout ~ yearsoftv + advote + factor(regions5) *
    poly(trend, 3), dat = dat, ind = dat$stcounty)
```

Results with Trends

- We'll come back to this example next recitation.

```
linear.est <- robust.se(linear.trend[[1]], linear.trend[[3]])[1, ]
linear.reg.est <- robust.se(linear.reg.trend[[1]], linear.reg.trend[[3]])[1,
    ]
cubic.est <- robust.se(cubic.trend[[1]], cubic.trend[[3]])[1, ]
cubic.reg.est <- robust.se(cubic.reg.trend[[1]], cubic.trend[[3]])[1, ]
all.ests <- rbind(all.ests, linear.est, linear.reg.est, cubic.est, cubic.reg.est)
rownames(all.ests)[5:8] <- c("linear", "linear.region", "cubic", "cubic.region")
all.ests
```

```
##               Estimate Std. Error t value  Pr(>|t|)
## base            0.2475    0.01309  18.915 1.497e-79
## simple         -0.1729    0.07258  -2.382 1.722e-02
## simple.swept   -0.1729    0.03230  -5.353 8.706e-08
## full.swept     -0.3863    0.04987  -7.746 9.619e-15
## linear          0.1476    0.01549   9.528 1.658e-21
## linear.region   0.1985    0.01468  13.524 1.301e-41
## cubic          -0.2946    0.04486  -6.568 5.151e-11
## cubic.region   -0.1188    0.04130  -2.877 4.017e-03
```

Splines

- You can also do this with splines, which have a slightly more non-parametric flavor.

. . .

```
require(splines)
spline.trend <- sweeplm(turnout ~ yearsoftv + advote + ns(trend, 3), dat = dat,
    ind = dat$stcounty)
spline.est <- robust.se(spline.trend[[1]], spline.trend[[3]])[1, ]
all.ests <- rbind(all.ests, spline.est)
rownames(all.ests)[9] <- "splines"
all.ests
```

```
##                Estimate Std. Error t value  Pr(>|t|)
## base             0.2475    0.01309  18.915 1.497e-79
## simple          -0.1729    0.07258  -2.382 1.722e-02
## simple.swept    -0.1729    0.03230  -5.353 8.706e-08
## full.swept      -0.3863    0.04987  -7.746 9.619e-15
## linear           0.1476    0.01549   9.528 1.658e-21
## linear.region    0.1985    0.01468  13.524 1.301e-41
## cubic           -0.2946    0.04486  -6.568 5.151e-11
## cubic.region    -0.1188    0.04130  -2.877 4.017e-03
## splines         -0.4105    0.04833  -8.493 2.058e-17
```