```
1  ********************************************************************************
2  *
3  *   1. Data Cleaning
4  *      A. Cattle dataset
5  *      B. LFP (Livestock Forage Disaster Program) dataset
6  *      C. Drought data
7  *      D. Control variables
8  *
9  *   2. Regressions
10 *      - Main models
11 *      - Any robustness checks or alternative specifications
12 ********************************************************************************
13 *****1. Data Cleaning*****
14
15 ********************************************************************************
16 * 1) CATTLE DATA (CattlePop0023.csv)
17 ********************************************************************************
18 import delimited "$Main/Data/CattlePop0023.csv", clear
19 save "cattle_inventory_data.dta", replace
20
21 * Rename State Variables
22 replace state = "AL" if state == "ALABAMA"
23 replace state = "AK" if state == "ALASKA"
24 replace state = "AZ" if state == "ARIZONA"
25 replace state = "AR" if state == "ARKANSAS"
26 replace state = "CA" if state == "CALIFORNIA"
27 replace state = "CO" if state == "COLORADO"
28 replace state = "CT" if state == "CONNECTICUT"
29 replace state = "DE" if state == "DELAWARE"
30 replace state = "FL" if state == "FLORIDA"
31 replace state = "GA" if state == "GEORGIA"
32 replace state = "HI" if state == "HAWAII"
33 replace state = "ID" if state == "IDAHO"
34 replace state = "IL" if state == "ILLINOIS"
35 replace state = "IN" if state == "INDIANA"
36 replace state = "IA" if state == "IOWA"
37 replace state = "KS" if state == "KANSAS"
38 replace state = "KY" if state == "KENTUCKY"
39 replace state = "LA" if state == "LOUISIANA"
40 replace state = "ME" if state == "MAINE"
41 replace state = "MD" if state == "MARYLAND"
42 replace state = "MA" if state == "MASSACHUSETTS"
43 replace state = "MI" if state == "MICHIGAN"
44 replace state = "MN" if state == "MINNESOTA"
```

```
replace state = "MS" if state == "MISSISSIPPI"
replace state = "MO" if state == "MISSOURI"
replace state = "MT" if state == "MONTANA"
replace state = "NE" if state == "NEBRASKA"
replace state = "NV" if state == "NEVADA"
replace state = "NH" if state == "NEW HAMPSHIRE"
replace state = "NJ" if state == "NEW JERSEY"
replace state = "NM" if state == "NEW MEXICO"
replace state = "NY" if state == "NEW YORK"
replace state = "NC" if state == "NORTH CAROLINA"
replace state = "ND" if state == "NORTH DAKOTA"
replace state = "OH" if state == "OHIO"
replace state = "OK" if state == "OKLAHOMA"
replace state = "OR" if state == "OREGON"
replace state = "PA" if state == "PENNSYLVANIA"
replace state = "RI" if state == "RHODE ISLAND"
replace state = "SC" if state == "SOUTH CAROLINA"
replace state = "SD" if state == "SOUTH DAKOTA"
replace state = "TN" if state == "TENNESSEE"
replace state = "TX" if state == "TEXAS"
replace state = "UT" if state == "UTAH"
replace state = "VT" if state == "VERMONT"
replace state = "VA" if state == "VIRGINIA"
replace state = "WA" if state == "WASHINGTON"
replace state = "WV" if state == "WEST VIRGINIA"
replace state = "WI" if state == "WISCONSIN"
replace state = "WY" if state == "WYOMING"

* Rename "value" -> "cattle"; clean numeric
rename value cattle
replace cattle = trim(cattle)
replace cattle = "" if cattle == "(D)"
replace cattle = subinstr(cattle, ",", "", .)
destring cattle, replace

* Create ANSI-based FIPS
gen state_fips = substr("00" + string(stateansi), -2, 2)
gen county_fips = substr("000" + string(countyansi), -3, 3)
gen fips_code = state_fips + county_fips

***Creating Balanced Panel***
sort fips_code year
duplicates drop fips_code year, force

drop if year < 2000 | year > 2023

by fips_code: egen n = count(cattle)
by fips_code: egen miny = min(year)
```

```
by fips_code: egen maxy = max(year)
drop if n != 24 | miny != 2000 | maxy != 2023

* Create natural log of cattle
gen ln_cattle = .
replace ln_cattle = ln(cattle)

* (B) Alternatively, if you want last year's *log of cattle*:
bysort fips_code (year): gen float avg_herd_l1 = .
bysort fips_code (year): replace avg_herd_l1 = ln_cattle[_n-1] if
_n>1

save "county_cattle_data.dta", replace

*************************************************************************
************
* 2) LFP DATA, 2014–2023
*************************************************************************
************
tempfile master_lfp
save `master_lfp', emptyok

*** 2014
import delimited "lfp_eligibility_2014.csv", varnames(1)
rename countyns fips
generate year = 2014
rename fsi2014 fsi
rename wsi2014 wsi
rename np2014  np
keep statefp countyfp fips name fsi np year
append using `master_lfp'
save `master_lfp', replace

*** 2015
import delimited "lfp_eligibility_2015.csv", varnames(1) clear
rename countyns fips
generate year = 2015
rename fsi2015 fsi
rename wsi2015 wsi
rename np2015  np
keep statefp countyfp fips name fsi wsi np year
append using `master_lfp'
save `master_lfp', replace

*** 2016
import delimited "lfp_eligibility_2016.csv", varnames(1) clear
rename countyns fips
generate year = 2016
```

```
rename fsi2016 fsi
rename wsi2016 wsi
rename np2016  np
keep statefp countyfp fips name fsi wsi np year
append using `master_lfp'
save `master_lfp', replace

*** 2017
import delimited "lfp_eligibility_2017.csv", varnames(1) clear
rename countyns fips
generate year = 2017
rename fsi2017 fsi
rename np2017  np
rename csi2017 csi
keep statefp countyfp fips name fsi csi np year
append using `master_lfp'
save `master_lfp', replace

*** 2018
import delimited "lfp_eligibility_2018.csv", varnames(1) clear
rename countyns fips
generate year = 2018
rename fsi2018 fsi
rename wsi2018 wsi
rename np2018  np
keep statefp countyfp fips name fsi wsi np year
append using `master_lfp'
save `master_lfp', replace

*** 2019
import delimited "lfp_eligibility_2019.csv", varnames(1) clear
rename countyns fips
generate year = 2019
rename fsi2019 fsi
rename wsi2019 wsi
rename np2019  np
keep statefp countyfp fips name fsi wsi np year
append using `master_lfp'
save `master_lfp', replace

*** 2020
import delimited "lfp_eligibility_2020.csv", varnames(1) clear
rename countyns fips
generate year = 2020
rename fsi2020 fsi
rename wsi2020 wsi
rename np2020  np
keep statefp countyfp fips name fsi wsi np year
```

```
append using `master_lfp'
save `master_lfp', replace

*** 2021
import delimited "lfp_eligibility_2021.csv", varnames(1) clear
rename countyns fips
generate year = 2021
rename fsi2021 fsi
rename wsi2021 wsi
rename np2021  np
keep statefp countyfp fips name fsi wsi np year
append using `master_lfp'
save `master_lfp', replace

*** 2022
import delimited "lfp_eligibility_2022.csv", varnames(1) clear
rename countyns fips
generate year = 2022
rename fsi2022 fsi
rename wsi2022 wsi
rename np2022  np
keep statefp countyfp fips name fsi wsi np year
append using `master_lfp'
save `master_lfp', replace

*** 2023
import delimited "lfp_eligibility_2023.csv", varnames(1) clear
rename countyns fips
generate year = 2023
rename fsi2023 fsi
rename wsi2023 wsi
rename np2023  np
keep statefp countyfp fips name fsi wsi np year
append using `master_lfp'
save `master_lfp', replace

* Reopen appended data
use `master_lfp', clear

* Create a numeric FIPS if you like
destring statefp countyfp, replace force
gen long fips_num = statefp*1000 + countyfp

replace fips = 46102 if fips == 46113
rename fips fips_code
destring year, replace
recast int year

```

```
234 ********************************************************************************
    ************
235 Generate separate LFP Payment Dummies from np
236 ********************************************************************************
    ************
237
238 * Assuming 'np' takes integer values {1, 3, 4, 5} for the number
    of months of LFP
239 * For each value, create a distinct dummy (0/1):
240 gen byte np1 = (np == 1)
241 gen byte np3 = (np == 3)
242 gen byte np4 = (np == 4)
243 gen byte np5 = (np == 5)
244
245 save "lfp_full_appended.dta", replace
246
247 ********************************************************************************
    ************
248 * 3) DROUGHT DATA (2000droughtcounty.csv)
249 ********************************************************************************
    ************
250 import delimited "$Main/Stata/Data/2000droughtcounty.csv", clear
251 tostring mapdate, replace
252 gen date = date(mapdate,"YMD")
253 format date %td
254 gen year = year(date)
255 gen month = month(date)
256 gen day = day(date)
257 tostring fips, replace format(%05.0f)
258 rename fips fips_code
259 sort fips_code year date
260
261 * 1) MAX classification per week
262 gen byte maxcat_week = .
263 replace maxcat_week = 4 if d4 > 0
264 replace maxcat_week = 3 if maxcat_week == . & d3 > 0
265 replace maxcat_week = 2 if maxcat_week == . & d2 > 0
266 replace maxcat_week = 1 if maxcat_week == . & d1 > 0
267 replace maxcat_week = 0 if maxcat_week == . & d0 > 0
268 replace maxcat_week = -1 if maxcat_week == . & none > 0
269
270 * 2) Count total (non-consecutive) weeks in each category (MAX
    perspective)
271 by fips_code year: egen weeks_d4_max = total(maxcat_week==4)
272 by fips_code year: egen weeks_d3_max = total(maxcat_week==3)
273 by fips_code year: egen weeks_d2_max = total(maxcat_week==2)
274 by fips_code year: egen weeks_d1_max = total(maxcat_week==1)
275 by fips_code year: egen weeks_d0_max = total(maxcat_week==0)
```

```
by fips_code year: egen weeks_none_max = total(maxcat_week==-1)
by fips_code year: egen total_weeks_max = count(maxcat_week)

* 3) DOMINANT classification
gen byte domcat_week = .
replace domcat_week = 4 if d4 >= d3 & d4 >= d2 & d4 >= d1 & d4 >= d0 & d4 >= none
replace domcat_week = 3 if domcat_week == . & d3 >= d2 & d3 >= d1 & d3 >= d0 & d3 >= none
replace domcat_week = 2 if domcat_week == . & d2 >= d1 & d2 >= d0 & d2 >= none
replace domcat_week = 1 if domcat_week == . & d1 >= d0 & d1 >= none
replace domcat_week = 0 if domcat_week == . & d0 >= none
replace domcat_week = -1 if domcat_week == . & none > 0

by fips_code year: egen weeks_d4_dom = total(domcat_week==4)
by fips_code year: egen weeks_d3_dom = total(domcat_week==3)
by fips_code year: egen weeks_d2_dom = total(domcat_week==2)
by fips_code year: egen weeks_d1_dom = total(domcat_week==1)
by fips_code year: egen weeks_d0_dom = total(domcat_week==0)
by fips_code year: egen weeks_none_dom = total(domcat_week==-1)
by fips_code year: egen total_weeks_dom = count(domcat_week)

***small versus large***
* (1) For the first 3 weeks of D3:
gen d3_small = weeks_d3_max
replace d3_small = 3 if d3_small > 3

* (2) For additional D3 beyond those 3 weeks:
gen d3_large = weeks_d3_max - 3
replace d3_large = 0 if d3_large < 0


* (1) For the first 3 weeks of D3:
gen d4_small = weeks_d4_max
replace d4_small = 3 if d4_small > 3

* (2) For additional D3 beyond those 3 weeks:
gen d4_large = weeks_d4_max - 3
replace d4_large = 0 if d4_large < 0



drop if year<2000 | year>2023
```

```
321
322 ********************************************************************
    ************
323 * 7) Initialize run variables for consecutive dryness (MAX
    perspective)
324 ********************************************************************
    ************
325 generate float run_d0plus = .
326 generate float run_d1plus = .
327 generate float run_d2plus = .
328 generate float run_d3plus = .
329 generate float run_d4     = .
330
331 ********************************************************************
    ************
332 * 8) For each dryness level, count consecutive runs (MAX
    perspective)
333 ********************************************************************
    ************
334
335 by fips_code year: replace run_d0plus = 0 if _n==1
336 by fips_code year: replace run_d0plus = cond(d0plus_week==0, 0,
    cond(d0plus_week[_n-1]==1, run_d0plus[_n-1]+1, 1)) if _n>1
337 by fips_code year: replace run_d0plus = 1 if _n==1 & d0plus_week==1
338
339 by fips_code year: replace run_d1plus = 0 if _n==1
340 by fips_code year: replace run_d1plus = cond(d1plus_week==0, 0,
    cond(d1plus_week[_n-1]==1, run_d1plus[_n-1]+1, 1)) if _n>1
341 by fips_code year: replace run_d1plus = 1 if _n==1 & d1plus_week==1
342
343 by fips_code year: replace run_d2plus = 0 if _n==1
344 by fips_code year: replace run_d2plus = cond(d2plus_week==0, 0,
    cond(d2plus_week[_n-1]==1, run_d2plus[_n-1]+1, 1)) if _n>1
345 by fips_code year: replace run_d2plus = 1 if _n==1 & d2plus_week==1
346
347 by fips_code year: replace run_d3plus = 0 if _n==1
348 by fips_code year: replace run_d3plus = cond(d3plus_week==0, 0,
    cond(d3plus_week[_n-1]==1, run_d3plus[_n-1]+1, 1)) if _n>1
349 by fips_code year: replace run_d3plus = 1 if _n==1 & d3plus_week==1
350
351 by fips_code year: replace run_d4 = 0 if _n==1
352 by fips_code year: replace run_d4 = cond(d4plus_week==0, 0, cond(
    d4plus_week[_n-1]==1, run_d4[_n-1]+1, 1)) if _n>1
353 by fips_code year: replace run_d4 = 1 if _n==1 & d4plus_week==1
354
355 ********************************************************************
    ************
356 * 9) Similarly, create run variables for the DOM perspective
```

```
    ************
358 generate float run_d0plus_dom = .
359 generate float run_d1plus_dom = .
360 generate float run_d2plus_dom = .
361 generate float run_d3plus_dom = .
362 generate float run_d4_dom     = .
363
364 by fips_code year: replace run_d0plus_dom = 0 if _n==1
365 by fips_code year: replace run_d0plus_dom = cond(d0plus_week_dom==0
    , 0, cond(d0plus_week_dom[_n-1]==1, run_d0plus_dom[_n-1]+1, 1)) if
    _n>1
366 by fips_code year: replace run_d0plus_dom = 1 if _n==1 &
    d0plus_week_dom==1
367
368 by fips_code year: replace run_d1plus_dom = 0 if _n==1
369 by fips_code year: replace run_d1plus_dom = cond(d1plus_week_dom==0
    , 0, cond(d1plus_week_dom[_n-1]==1, run_d1plus_dom[_n-1]+1, 1)) if
    _n>1
370 by fips_code year: replace run_d1plus_dom = 1 if _n==1 &
    d1plus_week_dom==1
371
372 by fips_code year: replace run_d2plus_dom = 0 if _n==1
373 by fips_code year: replace run_d2plus_dom = cond(d2plus_week_dom==0
    , 0, cond(d2plus_week_dom[_n-1]==1, run_d2plus_dom[_n-1]+1, 1)) if
    _n>1
374 by fips_code year: replace run_d2plus_dom = 1 if _n==1 &
    d2plus_week_dom==1
375
376 by fips_code year: replace run_d3plus_dom = 0 if _n==1
377 by fips_code year: replace run_d3plus_dom = cond(d3plus_week_dom==0
    , 0, cond(d3plus_week_dom[_n-1]==1, run_d3plus_dom[_n-1]+1, 1)) if
    _n>1
378 by fips_code year: replace run_d3plus_dom = 1 if _n==1 &
    d3plus_week_dom==1
379
380 by fips_code year: replace run_d4_dom = 0 if _n==1
381 by fips_code year: replace run_d4_dom = cond(d4plus_week_dom==0, 0,
     cond(d4plus_week_dom[_n-1]==1, run_d4_dom[_n-1]+1, 1)) if _n>1
382 by fips_code year: replace run_d4_dom = 1 if _n==1 &
    d4plus_week_dom==1
383
384 ************************************************************************
    ************
385 * 10) Create annual max runs (both MAX & DOM)
386 ************************************************************************
    ************
387 * -- MAX perspective
388 by fips_code year: egen max_run_d0plus    = max(run_d0plus)
```

```
by fips_code year: egen max_run_d1plus    = max(run_d1plus)
by fips_code year: egen max_run_d2plus    = max(run_d2plus)
by fips_code year: egen max_run_d3plus    = max(run_d3plus)
by fips_code year: egen max_run_d4        = max(run_d4)

* -- DOM perspective
by fips_code year: egen max_run_d0plus_dom = max(run_d0plus_dom)
by fips_code year: egen max_run_d1plus_dom = max(run_d1plus_dom)
by fips_code year: egen max_run_d2plus_dom = max(run_d2plus_dom)
by fips_code year: egen max_run_d3plus_dom = max(run_d3plus_dom)
by fips_code year: egen max_run_d4_dom     = max(run_d4_dom)

*****************************************************************************
* 11) LFP Payment Flags (ensuring alignment with USDA table)
*****************************************************************************
* Per your table:
*  1 month = "8 or more WEEKS continuous D2"
*  3 months= "At least 1 WEEK D3" (continuous or not)
*  4 months= "4 or more WEEKS (not necessarily continuous) of D3
OR at least 1 WEEK of D4"
*  5 months= "4 or more WEEKS (not necessarily continuous) of D4"

* A) 1-month LFP:
generate byte pay1_flag = (max_run_d2plus >= 8)

* B) 3-month LFP:
generate byte pay3_flag = (max_run_d3plus >= 1)

* C) 4-month LFP:
*  - "4 or more WEEKS of D3 or at least 1 WEEK of D4"
*    We'll interpret that as (weeks_d3_max >= 4) OR (weeks_d4_max
>=1).
generate byte pay4_flag = (weeks_d3_max >= 4 | weeks_d4_max >= 1)

* D) 5-month LFP:
*  - "4 or more weeks (not necessarily continuous) of D4"
generate byte pay5_flag = (weeks_d4_max >= 4)


* Step A) If pay5=1, all lower flags = 0
replace pay4_flag = 0 if pay5_flag==1
replace pay3_flag = 0 if pay5_flag==1
replace pay1_flag = 0 if pay5_flag==1

* Step B) If pay4=1 (and pay5=0), set pay3=0 and pay1=0
replace pay3_flag = 0 if pay4_flag==1
```

```
433  replace pay1_flag = 0 if pay4_flag==1
434
435  * Step C) If pay3=1 (and pay4=0 & pay5=0), set pay1=0
436  replace pay1_flag = 0 if pay3_flag==1
437
438
439  * 1) 1-month LFP
440  generate byte pay1_dom = (max_run_d2plus_dom >= 8)
441
442  * 2) 3-month LFP
443  generate byte pay3_dom = (max_run_d3plus_dom >= 1)
444
445  * 3) 4-month LFP
446  generate byte pay4_dom = (weeks_d3_dom >= 4 | weeks_d4_dom >= 1)
447
448  * 4) 5-month LFP
449  generate byte pay5_dom = (weeks_d4_dom >= 4)
450
451  * If pay5_dom == 1, zero out pay4_dom, pay3_dom, pay1_dom
452  replace pay4_dom = 0 if pay5_dom==1
453  replace pay3_dom = 0 if pay5_dom==1
454  replace pay1_dom = 0 if pay5_dom==1
455
456  * If pay4_dom == 1 (and pay5_dom==0), zero out pay3_dom, pay1_dom
457  replace pay3_dom = 0 if pay4_dom==1
458  replace pay1_dom = 0 if pay4_dom==1
459
460  * If pay3_dom == 1 (and pay4_dom==0 & pay5_dom==0), zero out
     pay1_dom
461  replace pay1_dom = 0 if pay3_dom==1
462
463
464
465  save "drought_measures.dta", replace
466
467  ************************************************************************
     ************
468  * 4) CONTROLS
469  ************************************************************************
     ************
470  * 4a) Unemployment
471  import delimited "Unemployment2023.csv", clear
472  gen str4 str_year = substr(attribute, length(attribute) - 3, 4)
473  gen year = real(str_year)
474  drop str_year
475  gen measure = substr(attribute, 1, length(attribute) - 5)
476  drop attribute
477  keep if value=="Unemployment_rate"
```

```
save "unemploymentcontrols.dta", replace

***Total land
import delimited "TotalAcres.csv", clear

* 4b) ANSI code
gen state_fips = substr("00" + string(stateansi), -2, 2)
gen county_fips = substr("000" + string(countyansi), -3, 3)
gen fips_code = state_fips + county_fips
drop if county == "OTHER (COMBINED) COUNTIES" | county == "OTHER
COUNTIES"

destring year, replace
rename value AgLand
keep fips_code county year AgLand
sort fips_code year
duplicates drop fips_code year, force
save "TotalAgLand.dta", replace

gen is_numeric = regexm(AgLand, "^[0-9,\\.]+$")
by fips_code, sort: egen all_numeric = min(is_numeric)
drop if all_numeric == 0
drop is_numeric all_numeric

replace AgLand = subinstr(AgLand, ",", "", .)
destring AgLand, replace
destring fips_code, replace

tsset fips_code year, yearly
tsfill, full
tsappend, add(1)

* Carry-forward approach: create a "step" variable for AgLand
bysort fips_code (year): gen AgLand_step = AgLand
bysort fips_code (year): replace AgLand_step = AgLand_step[_n-1] if
 missing(AgLand_step) & _n > 1

drop if year < 2000 | year > 2023
tostring fips_code, replace format(%05.0f)
save "TotalAgLand.dta", replace


****Irrigation****
import delimited "IrrigationUSDA.csv", clear
gen state_fips = substr("00" + string(stateansi), -2, 2)
gen county_fips = substr("000" + string(countyansi), -3, 3)
gen fips_code = state_fips + county_fips
drop if county == "OTHER (COMBINED) COUNTIES" | county == "OTHER
```

```
    COUNTIES"
525
526 destring year, replace
527 rename value IrrigationAgLand
528 keep fips_code county year IrrigationAgLand
529 sort fips_code year
530 duplicates drop fips_code year, force
531 save "IrrigationUSDA.dta", replace
532
533 gen is_numeric = regexm(IrrigationAgLand, "^[0-9,\\.]+$")
534 by fips_code, sort: egen all_numeric = min(is_numeric)
535 drop if all_numeric == 0
536 drop is_numeric all_numeric
537
538 replace IrrigationAgLand = subinstr(IrrigationAgLand, ",", "", .)
539 destring IrrigationAgLand, replace
540 destring fips_code, replace
541
542 tsset fips_code year, yearly
543 tsfill, full
544 tsappend, add(1)
545
546 * Carry-forward approach: create a "step" variable for AgLand
547 bysort fips_code (year): gen AgLand_step = AgLand
548 bysort fips_code (year): replace AgLand_step = AgLand_step[_n-1] if
     missing(AgLand_step) & _n > 1
549
550 drop if year < 2000 | year > 2023
551 tostring fips_code, replace format(%05.0f)
552 save "IrrigationUSDA.dta", replace
553
554 use "TotalAgLand.dta", clear
555 merge 1:1 fips_code year using "IrrigationUSDA.dta"
556 keep if _merge == 3
557 drop _merge county
558
559 gen float fraction = .
560 replace fraction = Irrigation / AgLand if AgLand > 0 &
    IrrigationAgLand >= 0
561
562 bysort fips_code (year): gen fraction_step = fraction
563 bysort fips_code (year): replace fraction_step = fraction_step[_n-1
    ] if missing(fraction_step) & _n>1
564 bysort fips_code (year): ipolate fraction year, gen(fraction_lin)
    epolate
565
566
567 save "AgLand_Irrigation_merged.dta", replace
```

```
568
569 use "unemploymentcontrols.dta", clear
570 tostring fips_code, replace format(%05.0f)
571 sort fips_code year
572
573 merge 1:1 fips_code year using "AgLand_Irrigation_merged.dta"
574 keep if _merge == 3
575 drop _merge county
576
577 save "controls.dta", replace
578
579
580 ********************************************************************
    ************
581 * 5) MERGE EVERYTHING TOGETHER
582 ********************************************************************
    ************
583
584 * 5a) Merge Cattle + LFP
585 use "county_cattle_data.dta", clear
586 merge 1:1 fips_code year using "lfp_full_appended.dta"
587 keep if _merge == 3
588 drop _merge
589
590 * 5b) Merge in Drought
591 merge 1:1 fips_code year using "drought_measures.dta"
592 keep if _merge == 3
593 drop _merge
594
595 * 5c) Merge in Controls
596 merge 1:1 fips_code year using "Controls.dta"
597 keep if _merge == 3
598 drop _merge
599
600 ********************************************************************
    ************
601 * 6) Final Save
602 ********************************************************************
    ************
603 save "final_merged_dataset.dta", replace
604
605
606 ***Payment Check****
607 ************************************************************
608 * For actual coverage = np1 => do they also have pay1_flag?
609 ************************************************************
610 preserve
611 keep if np4==1 & year>=2014
```

```
612  count
613  local total = r(N)
614  count if pay4_flag1==4
615  local matched = r(N)
616  local unmatched = `total' - `matched'
617  local matched_pct   = 100*`matched'/`total'
618  local unmatched_pct = 100*`unmatched'/`total'
619  di "========================================="
620  di " Among np1=1 => total: `total'"
621  di "   # pay4_flag1=1: `matched'  (" %5.2f matched_pct "%)"
622  di "   no pay4_flag1 : `unmatched' (" %5.2f unmatched_pct "%)"
623  di "========================================="
624  restore
625  ************************************************************
626  * For actual coverage = np1 => do they also have pay1_flag?
627  ************************************************************
628  preserve
629  keep if np4==1 & year>=2014
630  count
631  local total = r(N)
632  count if pay4_flag1==1
633  local matched = r(N)
634  local unmatched = `total' - `matched'
635  local matched_pct   = 100*`matched'/`total'
636  local unmatched_pct = 100*`unmatched'/`total'
637  di "========================================="
638  di " Among np1=1 => total: `total'"
639  di "   # pay4_flag1=1: `matched'  (" %5.2f matched_pct "%)"
640  di "   no pay4_flag1 : `unmatched' (" %5.2f unmatched_pct "%)"
641  di "========================================="
642  restore
643  ************************************************************
644  * For actual coverage = np1 => do they also have pay1_flag?
645  ************************************************************
646  preserve
647  keep if np5==1 & year>=2014
648  count
649  local total = r(N)
650  count if pay5_flag1==1
651  local matched = r(N)
652  local unmatched = `total' - `matched'
653  local matched_pct   = 100*`matched'/`total'
654  local unmatched_pct = 100*`unmatched'/`total'
655  di "========================================="
656  di " Among np1=1 => total: `total'"
657  di "   # pay5_flag1=1: `matched'  (" %5.2f matched_pct "%)"
658  di "   no pay5_flag1 : `unmatched' (" %5.2f unmatched_pct "%)"
659  di "========================================="
```

```
restore


****Adding in Regions*****
import delimited using "reglink.csv", clear
drop v3 v4 v5 v6 v7 v8 v9
sort fips_code
tostring fips_code, replace format(%05.0f)
sort fips_code year
sort fips_code
save "regionmap.dta", replace
 use "final_merged_dataset.dta", clear
tostring fips_code, replace format(%05.0f)
  sort fips_code
merge 1:1 fips_code using "regionmap.dta"
drop _merge
merge 1:1 fips_code using "regionmap.dta"
merge m:1 fips_code using "regionmap.dta"
save "final_merged_dataset.dta", replace
label list region_num
  destring fips_code, replace


**********************************************************************
**********
*2. Regressions*
**********************************************************************
**********
xtset fips_code year
****4.1*****
xtreg ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_d2_max#
i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.weeks_d4_max#i.
np4 c.weeks_d4_max#i.np5 unemployment_rate Irrigation) i.year, fe
vce(cluster fips_code)

xtreg ln_cattle L.ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.
weeks_d2_max c.weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5
c.weeks_d2_max#i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.
weeks_d4_max#i.np4 c.weeks_d4_max#i.np5 unemployment_rate
Irrigation) i.year, fe vce(cluster fips_code)

xtreg ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_d2_max#
i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.weeks_d4_max#i.
np4 c.weeks_d4_max#i.np5 unemployment_rate Irrigation) i.year [aw=
avg_herd_l1], fe vce(cluster fips_code)

```

```
694  xtreg ln_cattle L.ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.
     weeks_d2_max c.weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5
     c.weeks_d2_max#i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.
     weeks_d4_max#i.np4 c.weeks_d4_max#i.np5 unemployment_rate
     Irrigation) i.year [aw=avg_herd_l1], fe vce(cluster fips_code)
695
696
697
698  ****4.2****
699  xtreg ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
     d3_small c.d3_large c.d4_small c.d4_large i.np1 i.np3 i.np4 i.np5 c
     .weeks_d2_max#i.np1 c.d3_small#i.np3 c.d3_large#i.np4 c.d4_small#i.
     np4 c.d4_large#i.np5 unemployment_rate Irrigation) i.year, fe vce(
     cluster fips_code)
700
701  xtreg ln_cattle L.ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.
     weeks_d2_max c.d3_small c.d3_large c.d4_small c.d4_large i.np1 i.
     np3 i.np4 i.np5 c.weeks_d2_max#i.np1 c.d3_small#i.np3 c.d3_large#i.
     np4 c.d4_small#i.np4 c.d4_large#i.np5 unemployment_rate Irrigation)
      i.year, fe vce(cluster fips_code)
702
703  xtreg ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
     d3_small c.d3_large c.d4_small c.d4_large i.np1 i.np3 i.np4 i.np5 c
     .weeks_d2_max#i.np1 c.d3_small#i.np3 c.d3_large#i.np4 c.d4_small#i.
     np4 c.d4_large#i.np5 unemployment_rate Irrigation) i.year [aw=
     avg_herd_l1], fe vce(cluster fips_code)
704
705  xtreg ln_cattle L.ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.
     weeks_d2_max c.d3_small c.d3_large c.d4_small c.d4_large i.np1 i.
     np3 i.np4 i.np5 c.weeks_d2_max#i.np1 c.d3_small#i.np3 c.d3_large#i.
     np4 c.d4_small#i.np4 c.d4_large#i.np5 unemployment_rate Irrigation)
      i.year [aw=avg_herd_l1], fe vce(cluster fips_code)
706
707
708  ***4.3***
709  xtreg ln_cattle L.(c.max_run_d0plus c.max_run_d1plus c.
     max_run_d2plus c.max_run_d3plus c.max_run_d4 i.np1 i.np3 i.np4 i.
     np5 c.max_run_d2plus#i.np1 c.max_run_d3plus#i.np3 c.max_run_d3plus#
     i.np4 c.max_run_d4#i.np4 c.max_run_d4#i.np5 Irrigation
     unemployment_rate) i.year, fe vce(cluster fips_code)
710
711  xtreg ln_cattle L.ln_cattle L.(c.max_run_d0plus c.max_run_d1plus c.
     max_run_d2plus c.max_run_d3plus c.max_run_d4 i.np1 i.np3 i.np4 i.
     np5 c.max_run_d2plus#i.np1 c.max_run_d3plus#i.np3 c.max_run_d3plus#
     i.np4 c.max_run_d4#i.np4 c.max_run_d4#i.np5 Irrigation
     unemployment_rate) i.year, fe vce(cluster fips_code)
712
713  xtreg ln_cattle L.(c.max_run_d0plus c.max_run_d1plus c.
```

```
    max_run_d2plus c.max_run_d3plus c.max_run_d4 i.np1 i.np3 i.np4 i.
    np5 c.max_run_d2plus#i.np1 c.max_run_d3plus#i.np3 c.max_run_d3plus#
    i.np4 c.max_run_d4#i.np4 c.max_run_d4#i.np5 Irrigation
    unemployment_rate) i.year [aw=avg_herd_l1], fe vce(cluster
    fips_code)
714
715 xtreg ln_cattle L.ln_cattle L.(c.max_run_d0plus c.max_run_d1plus c.
    max_run_d2plus c.max_run_d3plus c.max_run_d4 i.np1 i.np3 i.np4 i.
    np5 c.max_run_d2plus#i.np1 c.max_run_d3plus#i.np3 c.max_run_d3plus#
    i.np4 c.max_run_d4#i.np4 c.max_run_d4#i.np5 Irrigation
    unemployment_rate) i.year [aw=avg_herd_l1], fe vce(cluster
    fips_code)
716
717 ***4.4***
718 xtreg ln_cattle L.(c.max_run_d0plus c.max_run_d1plus c.
    max_run_d2plus c.d3_small c.d3_large c.d4_small c.d4_large i.np1 i.
    np3 i.np4 i.np5 c.max_run_d2plus#i.np1 c.d3_small#i.np3 c.d3_large#
    i.np4 c.d4_small#i.np4 c.d4_large#i.np5 Irrigation
    unemployment_rate) i.year, fe vce(cluster fips_code)
719
720 xtreg ln_cattle L.(c.max_run_d0plus c.max_run_d1plus c.
    max_run_d2plus c.d3_small c.d3_large c.d4_small c.d4_large i.np1 i.
    np3 i.np4 i.np5 c.max_run_d2plus#i.np1 c.d3_small#i.np3 c.d3_large#
    i.np4 c.d4_small#i.np4 c.d4_large#i.np5 Irrigation
    unemployment_rate) i.year [aw=avg_herd_l1], fe vce(cluster
    fips_code)
721
722 xtreg ln_cattle L.ln_cattle L.(c.max_run_d0plus c.max_run_d1plus c.
    max_run_d2plus c.d3_small c.d3_large c.d4_small c.d4_large i.np1 i.
    np3 i.np4 i.np5 c.max_run_d2plus#i.np1 c.d3_small#i.np3 c.d3_large#
    i.np4 c.d4_small#i.np4 c.d4_large#i.np5 Irrigation
    unemployment_rate) i.year, fe vce(cluster fips_code)
723
724 xtreg ln_cattle L.ln_cattle L.(c.max_run_d0plus c.max_run_d1plus c.
    max_run_d2plus c.d3_small c.d3_large c.d4_small c.d4_large i.np1 i.
    np3 i.np4 i.np5 c.max_run_d2plus#i.np1 c.d3_small#i.np3 c.d3_large#
    i.np4 c.d4_small#i.np4 c.d4_large#i.np5 Irrigation
    unemployment_rate) i.year [aw=avg_herd_l1], fe vce(cluster
    fips_code)
725
726 ****Robustness Tests***
727 preserve
728 drop if year==2014
729 xtreg ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
    weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_d2_max#
    i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.weeks_d4_max#i.
    np4 c.weeks_d4_max#i.np5 Irrigation unemployment_rate) i.year, fe
    vce(cluster fips_code)
```

```
734 xtreg ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
    weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_d2_max#
    i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.weeks_d4_max#i.
    np4 c.weeks_d4_max#i.np5 unemployment_rate) i.year, fe vce(cluster
    fips_code)
735 restore
736
737 xtreg ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
    weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_d2_max#
    i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.weeks_d4_max#i.
    np4 c.weeks_d4_max#i.np5 unemployment_rate) i.year, fe vce(cluster
    fips_code)
738
739 xtreg ln_cattle L.(c.weeks_d0_dom c.weeks_d1_dom c.weeks_d2_dom c.
    weeks_d3_dom c.weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_d2_dom#
    i.np1 c.weeks_d3_dom#i.np3 c.weeks_d3_dom#i.np4 c.weeks_d4_dom#i.
    np4 c.weeks_d4_dom#i.np5 Irrigation unemployment_rate) i.year, fe
    vce(cluster fips_code)
740
741
742 ****Appendix****
743 ***lowdrought***
744 gen week_low_max = weeks_d0_max + weeks_d1_max + weeks_d2_max
745
746 xtreg ln_cattle L.(c.weeks_low_max c.weeks_d3_max c.weeks_d4_max i.
    np1 i.np3 i.np4 i.np5 c.weeks_low_max#i.np1 c.weeks_d3_max#i.np3 c.
    weeks_d3_max#i.np4 c.weeks_d4_max#i.np4 c.weeks_d4_max#i.np5
    Irrigation unemployment_rate) i.year, fe vce(cluster fips_code)
747
748 xtreg ln_cattle L.(c.weeks_low_max c.weeks_d3_max c.weeks_d4_max i.
    np1 i.np3 i.np4 i.np5 c.weeks_low_max#i.np1 c.weeks_d3_max#i.np3 c.
    weeks_d3_max#i.np4 c.weeks_d4_max#i.np4 c.weeks_d4_max#i.np5
    Irrigation unemployment_rate) i.year [aw=avg_herd_l1], fe vce(
    cluster fips_code)
749
750 xtreg ln_cattle L.ln_cattle L.(c.weeks_low_max c.weeks_d3_max c.
    weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_low_max#i.np1 c.
    weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.weeks_d4_max#i.np4 c.
    weeks_d4_max#i.np5 Irrigation unemployment_rate) i.year, fe vce(
    cluster fips_code)
751
752 xtreg ln_cattle L.ln_cattle L.(c.weeks_low_max c.weeks_d3_max c.
    weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_low_max#i.np1 c.
    weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.weeks_d4_max#i.np4 c.
    weeks_d4_max#i.np5 Irrigation unemployment_rate) i.year [aw=
    avg_herd_l1], fe vce(cluster fips_code)
753
754
```

```
***splines***

mkspline ss_d3_1 8 ss_d3_2 26 ss_d3_3 = weeks_d3_max
mkspline sss_d3_1 8 sss_d3_2 26 sss_d3_3 = weeks_d3_max
mkspline sss_d4_1 8 sss_d4_2 26 sss_d4_3 = weeks_d4_max
xtreg ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
    sss_d3_1 c.sss_d3_2 c.sss_d3_3 c.sss_d4_1 c.sss_d4_2 c.sss_d4_3 i.
    np1 i.np3 i.np4 i.np5 c.weeks_d2_max#i.np1 c.sss_d3_1#i.np3 c.
    sss_d3_2#i.np4 c.sss_d3_3#i.np4 c.sss_d4_1#i.np4 c.sss_d4_2#i.np5 c
    .sss_d4_3#i.np5 Irrigation unemployment_rate) i.year, fe vce(
    cluster fips_code)

xtreg ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
    sss_d3_1 c.sss_d3_2 c.sss_d3_3 c.sss_d4_1 c.sss_d4_2 c.sss_d4_3 i.
    np1 i.np3 i.np4 i.np5 c.weeks_d2_max#i.np1 c.sss_d3_1#i.np3 c.
    sss_d3_2#i.np4 c.sss_d3_3#i.np4 c.sss_d4_1#i.np4 c.sss_d4_2#i.np5 c
    .sss_d4_3#i.np5 Irrigation unemployment_rate) i.year [aw=
    avg_herd_l1], fe vce(cluster fips_code)

xtreg ln_cattle  L.ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.
    weeks_d2_max c.sss_d3_1 c.sss_d3_2 c.sss_d3_3 c.sss_d4_1 c.sss_d4_2
     c.sss_d4_3 i.np1 i.np3 i.np4 i.np5 c.weeks_d2_max#i.np1 c.sss_d3_1
    #i.np3 c.sss_d3_2#i.np4 c.sss_d3_3#i.np4 c.sss_d4_1#i.np4 c.
    sss_d4_2#i.np5 c.sss_d4_3#i.np5 Irrigation unemployment_rate) i.
    year, fe vce(cluster fips_code)

xtreg ln_cattle  L.ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.
    weeks_d2_max c.sss_d3_1 c.sss_d3_2 c.sss_d3_3 c.sss_d4_1 c.sss_d4_2
     c.sss_d4_3 i.np1 i.np3 i.np4 i.np5 c.weeks_d2_max#i.np1 c.sss_d3_1
    #i.np3 c.sss_d3_2#i.np4 c.sss_d3_3#i.np4 c.sss_d4_1#i.np4 c.
    sss_d4_2#i.np5 c.sss_d4_3#i.np5 Irrigation unemployment_rate) i.
    year [aw=avg_herd_l1], fe vce(cluster fips_code)



****lags***
**no lag**
xtreg ln_cattle (c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
    weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_d2_max#
    i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.weeks_d4_max#i.
    np4 c.weeks_d4_max#i.np5 Irrigation unemployment_rate) i.year, fe
    vce(cluster fips_code)

***lagged 2***
xtreg ln_cattle L2.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
    weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_d2_max#
    i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.weeks_d4_max#i.
    np4 c.weeks_d4_max#i.np5 Irrigation unemployment_rate) i.year, fe
```

```
    vce(cluster fips_code)
776
777 ****lagged 1 and 2******
778 xtreg ln_cattle L(1/2).(c.weeks_d0_max c.weeks_d1_max c.
    weeks_d2_max c.weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5
    c.weeks_d2_max#i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.
    weeks_d4_max#i.np4 c.weeks_d4_max#i.np5 Irrigation
    unemployment_rate) i.year, fe vce(cluster fips_code)
779
780
781 ****clustering by state****
782 xtreg ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
    weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_d2_max#
    i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.weeks_d4_max#i.
    np4 c.weeks_d4_max#i.np5 Irrigation unemployment_rate) i.year, fe
    vce(cluster state)
783
784 ****including an extra ln(cattle)****
785 xtreg ln_cattle L(1/2).ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c
    .weeks_d2_max c.weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5
     c.weeks_d2_max#i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.
    weeks_d4_max#i.np4 c.weeks_d4_max#i.np5 Irrigation
    unemployment_rate) i.year, fe vce(cluster state)
786
787 ***pre 2014***
788 preserve
789 keep if year<2014
790 xtreg ln_cattle (c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
    weeks_d3_max c.weeks_d4_max Irrigation unemployment_rate) i.year,
    fe vce(cluster fips_code)
791 restore
792
793 ***post 2014****
794 preserve
795 keep if year>2013
796 xtreg ln_cattle (c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
    weeks_d3_max c.weeks_d4_max i.np1 i.np3 i.np4 i.np5 c.weeks_d2_max#
    i.np1 c.weeks_d3_max#i.np3 c.weeks_d3_max#i.np4 c.weeks_d4_max#i.
    np4 c.weeks_d4_max#i.np5 Irrigation unemployment_rate) i.year, fe
    vce(cluster fips_code)
797 restore
798
799 ***Placebo Test***
800 xtreg ln_cattle L.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
    weeks_d3_max c.weeks_d4_max) F.(i.np1 i.np3 i.np4 i.np5) L.c.
    weeks_d2_max#F.i.np1 L.c.weeks_d3_max#F.i.np3 L.c.weeks_d3_max#F.i.
    np4 L.c.weeks_d4_max#F.i.np4 L.c.weeks_d4_max#F.i.np5 L.(Irrigation
     unemployment_rate) i.year, fe vce(cluster fips_code)
```

```
801
802  xtreg ln_cattle F.(c.weeks_d0_max c.weeks_d1_max c.weeks_d2_max c.
     weeks_d3_max c.weeks_d4_max) L.(i.np1 i.np3 i.np4 i.np5) F.c.
     weeks_d2_max#L.i.np1 F.c.weeks_d3_max#L.i.np3 F.c.weeks_d3_max#L.i.
     np4 F.c.weeks_d4_max#L.i.np4 F.c.weeks_d4_max#L.i.np5 L.(Irrigation
      unemployment_rate) i.year, fe vce(cluster fips_code)
803
804
805
```