`Panelview` in STATA: Visualizing Panel Data

The `panelview` package has two main functionalities: (1) it visualizes the treatment and missing-value statuses of each observation in a panel/time-series-cross-sectional (TSCS) dataset; and (2) it plots the outcome variable (either continuous or discrete) in a time-series fashion.

We develop this package in the belief that it is always a good idea to understand your raw data better before conducting statistical analyses.

Date: August 3, 2021

Version: 0.1 (Github)

Table of Contents

---

1. Syntax

The general syntax of the package can be summarized as:

```
panelview Y D X [if] [in] ,         ///
    I(varname) T(varname numeric)   ///
    [                               ///
    TYPE(string)                    ///
    discreteoutcome                 ///
    bytiming                        ///
    MYCOLor(string)                 ///
    PREpost(string)                 ///
    continuoustreat                 ///
    xlabdist(integer 1)             ///
    ylabdist(integer 1)             ///
    ignoretreat                     ///
    bytreatgroup                    ///
    linediscretreat                 ///
    allunitsplot                    ///
    *                               ///
    ]
```

Where the subcommand can be:

Subcommand	Description
`Y D X`	`varlist` of outcome variable, treatment variable, and covariates. Including covariates may change the plot because of missing values in these covariates.
`if` and `in`	If any variable not included in the `varlist` or `i()` / `t()` appears in the `if` / `in` subcommand, we should add this variable into the `varlist` following `panelview` command.
`I()` and `T()`	Specify the unit (group) and time indicators.
`TYPE()`	Use `type(treat)` to plot treatments; `type(outcome)` to plot outcomes. If not specify this option, we plot outcome and treatment against time in the same graph.
`discreteoutcome`	Plot the discrete outcome variable.
`bytiming`	Sort units by the timing of receiving the treatment (then by the total number of periods exposed to the treatment).
`MYCOLor()`	Change the color schemes; click here for sequential colors (3-9 colors). Default is `Reds`.
`PREpost(off)`	Not distinguish the pre- and post-treatment periods for treated units.
`continuoustreat`	Plot the continuous treatment variable. If it is combined with `type(outcome)`, the figure would be the same as ignoring treatment.
`xlabdist` and `ylabdist`	Change gaps between labels on the x- and y-axes. Default is 1.
`ignoretreat`	Omit the treatment indicator.
`bytreatgroup`	Put each unit into different treatment groups, then plot respectively.
`linediscretreat`	To visualize the zero level with discrete treatment, use line plot instead of bar plot.
`allunitsplot`	Plot mean D and Y against time in the same graph.

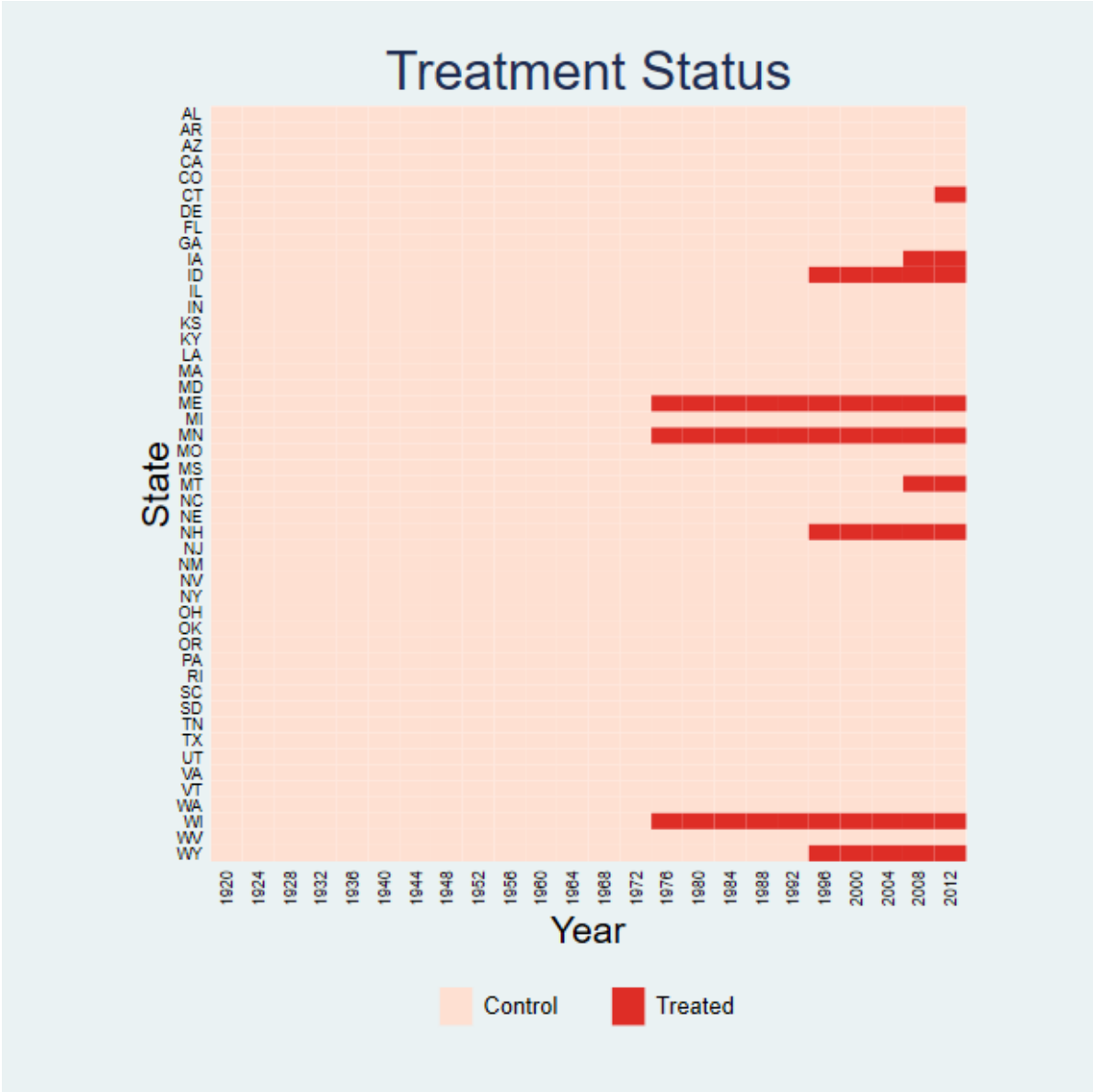
2. Plotting Treatment Conditions

First, we show how to visualize the dichotomous treatment conditions in a panel dataset. The treatment may switch on and off or have missing values.

2.1 Two Treatment Conditions

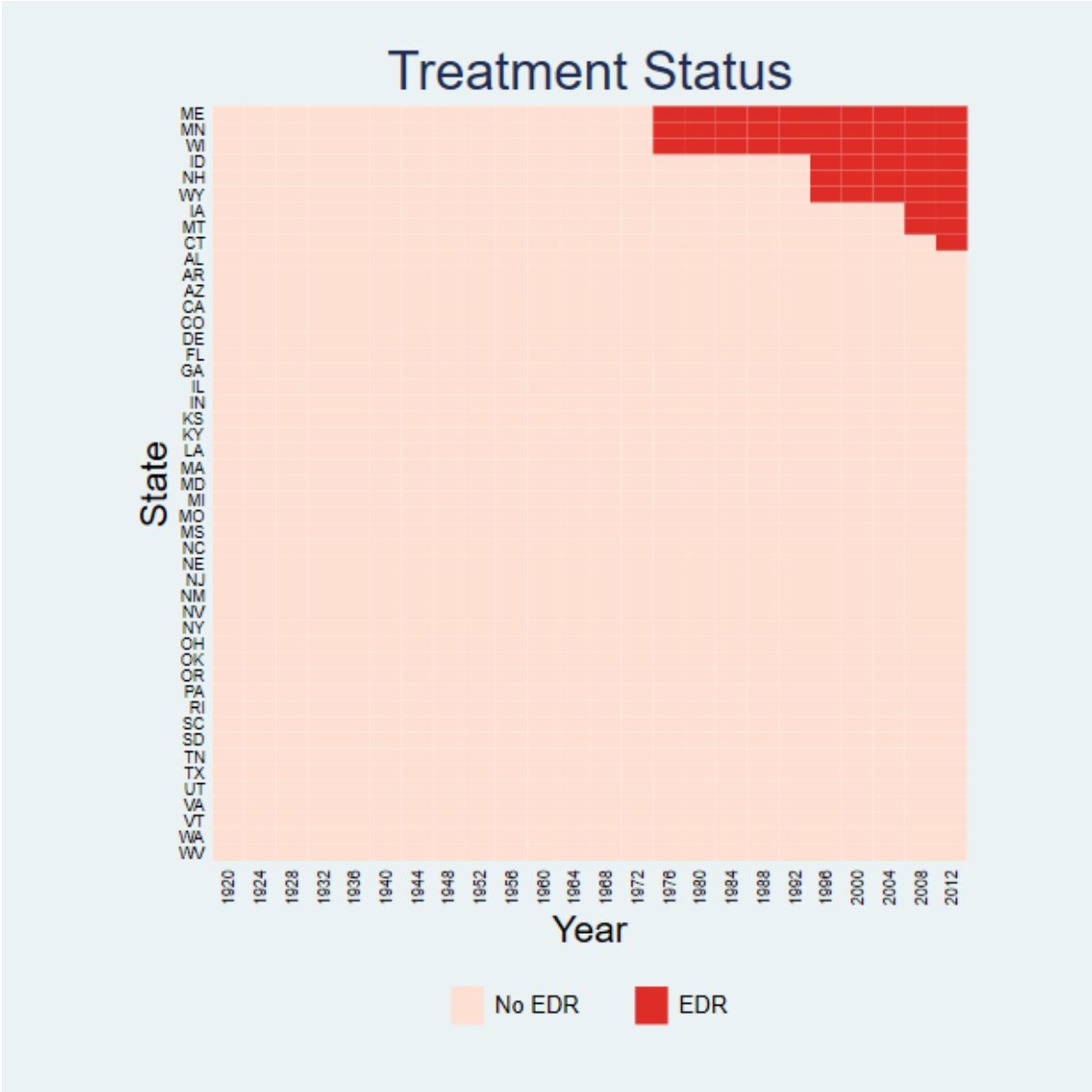
Using the `turnout` dataset (a balanced panel), we show the treatment status of Election Day Registration (EDR) in each state in a given year (Xu 2017). We can use the `title` option to change the title of the plot and change the titles of x- and y-axes through `xtitle` and `ytitle`, respectively. For DID-type TSCS data with a dichotomous treatment indicator, we can stop distinguish the pre- and post-treatment periods for treated units by specifying `prepost(off)`.

```
use turnout.dta, clear
panelview turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
type(treat) xtitle("Year") ytitle("State") title("Treatment Status")
prepost(off)
```



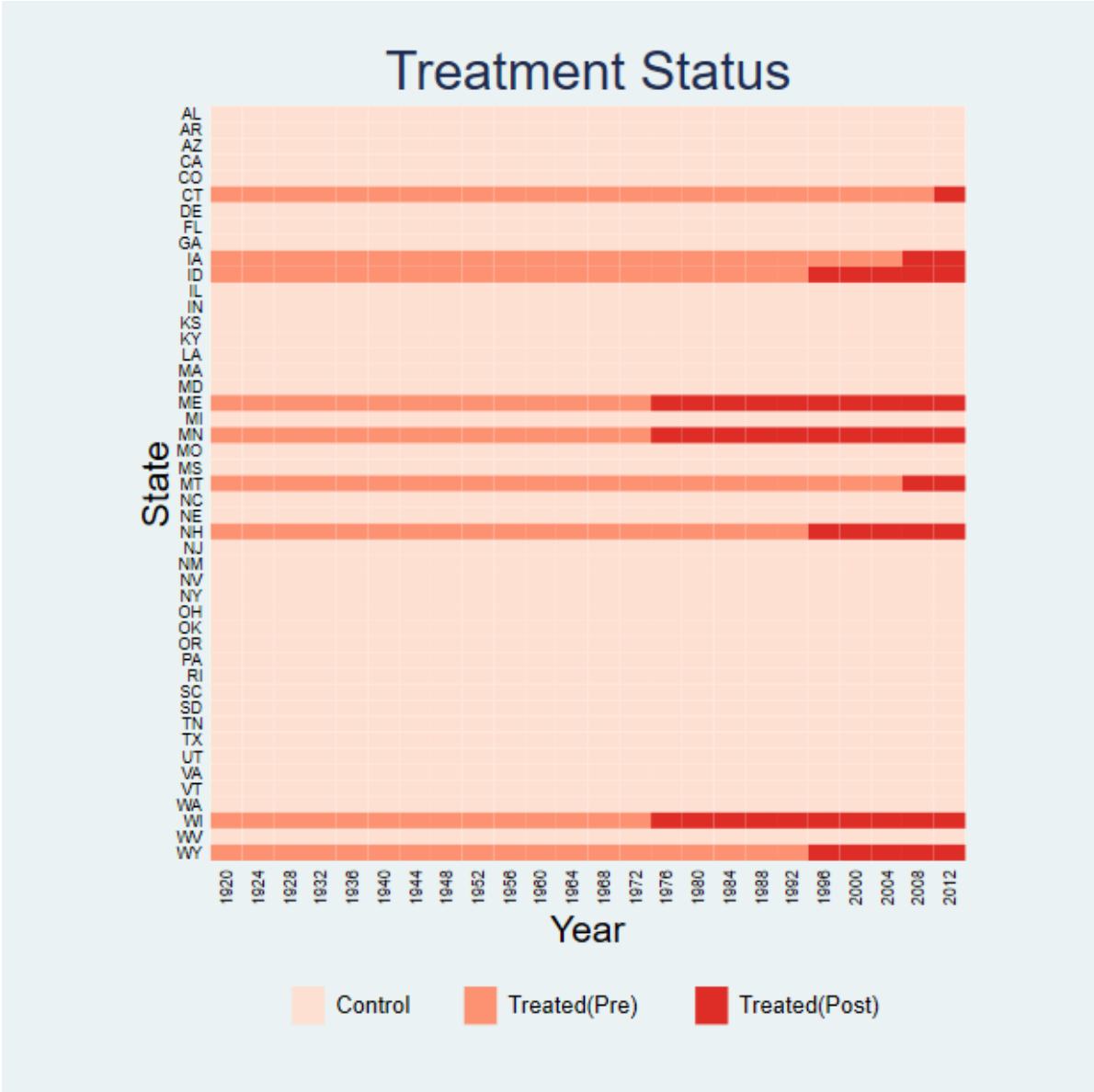
We can use the `bytiming` option to sort units by the timing of receiving the treatment and change the labels in the legend:

```
*bytiming
panelview turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
type(treat) xtitle("Year") ytitle("State") title("Treatment Status")
prepost(off) bytiming legend(label(1 "No EDR") label(2 "EDR"))
```



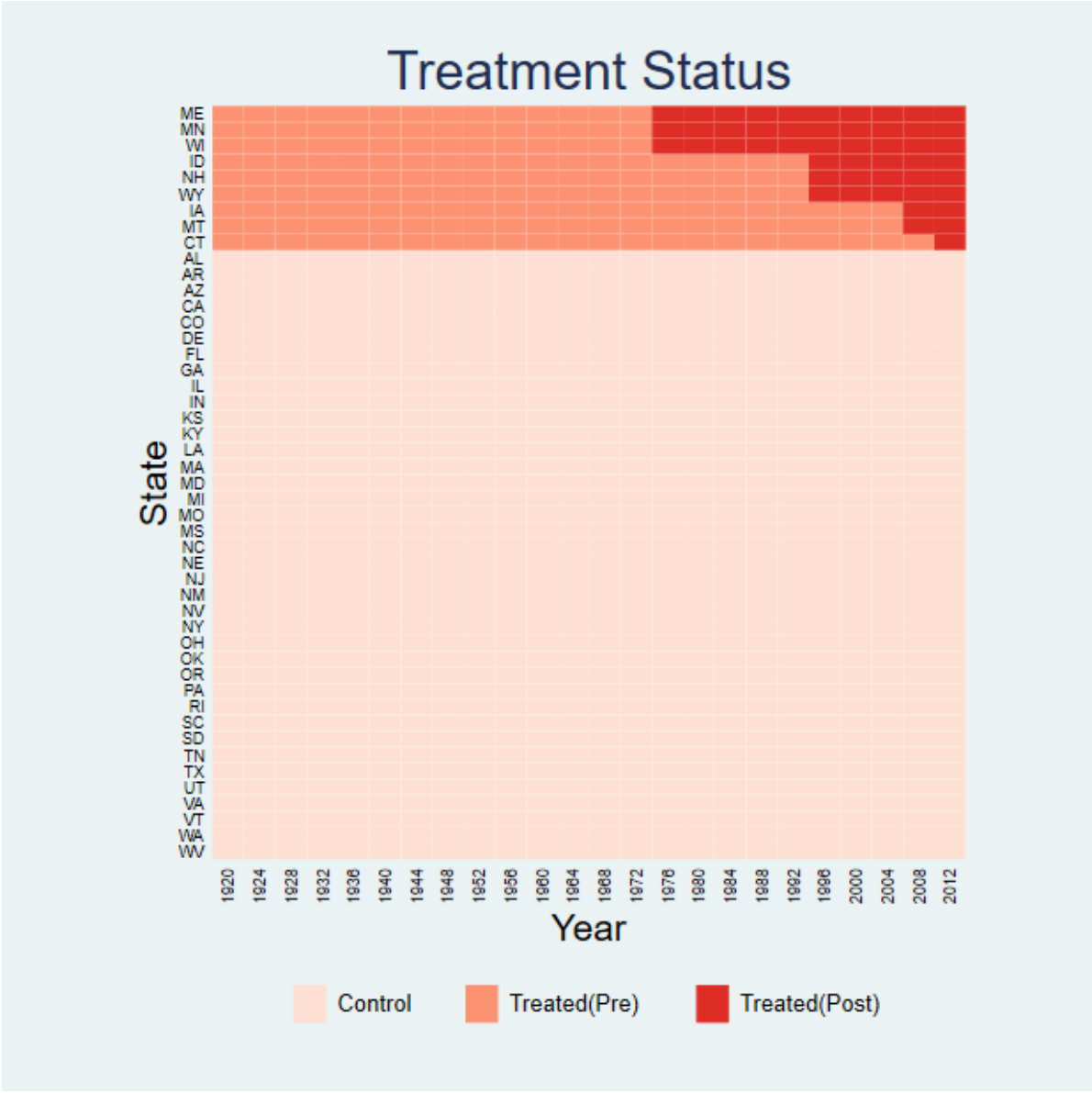
Distinguish the pre- and post-treatment periods for treated units by not specifying `prepost(off)`:

```
*prepost != off
panelview turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
type(treat) xtitle("Year") ytitle("State") title("Treatment Status")
```



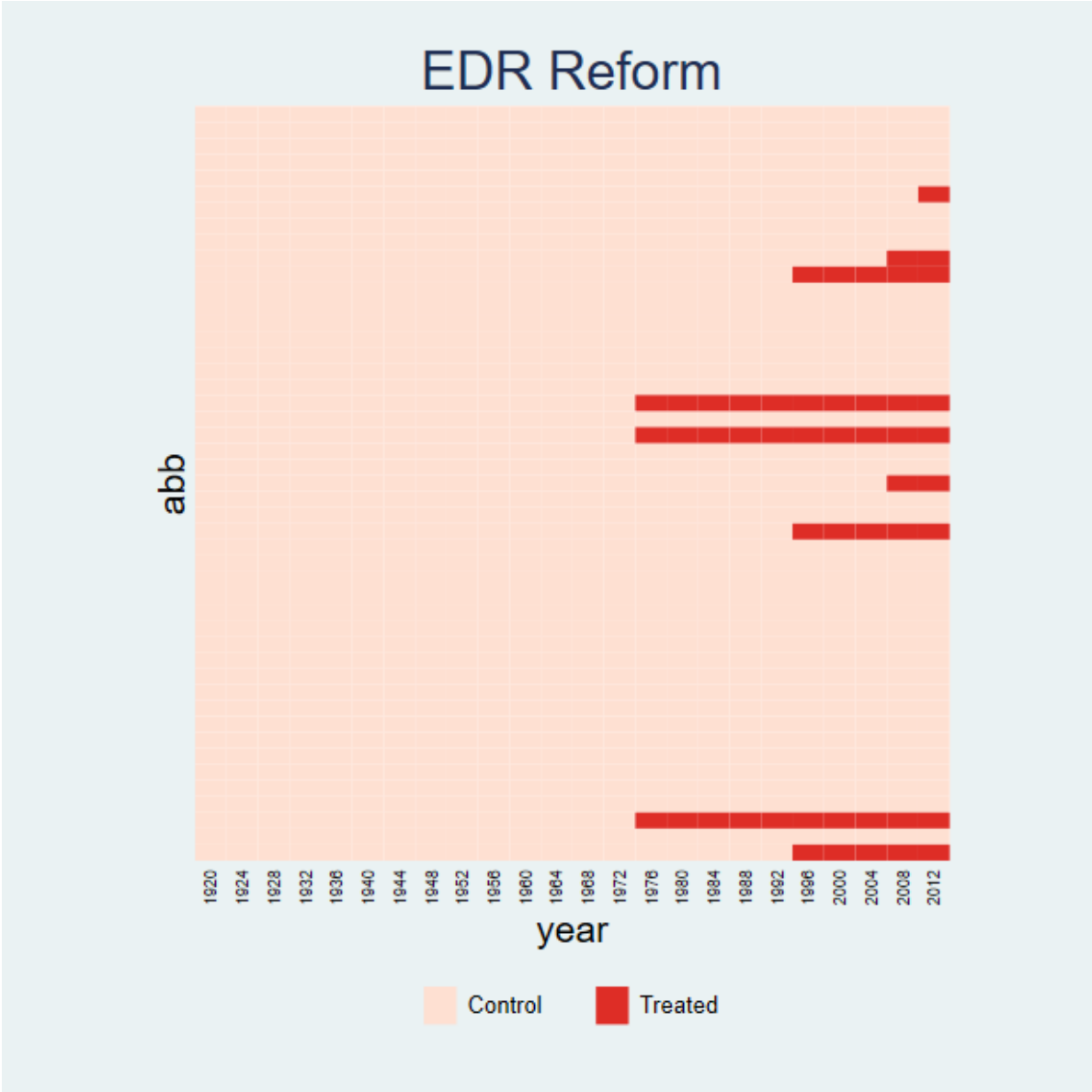
Again, sort units by the timing of receiving the treatment:

```
*bytiming
panelview turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
type(treat) xtitle("Year") ytitle("State") title("Treatment Status") bytiming
```



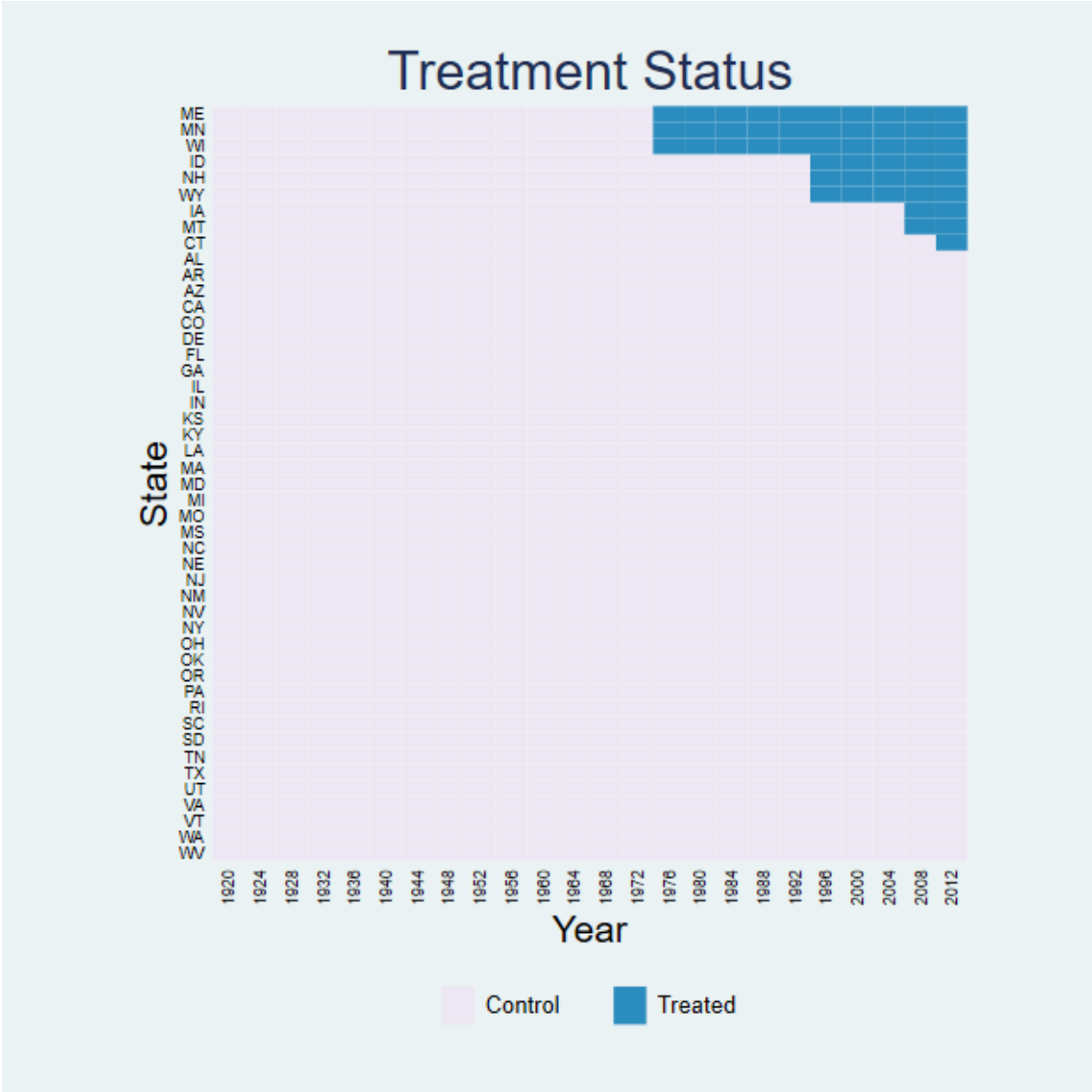
Remove the labels on the y-axis by specifying `ylabel("")` or `ylabel(none)`:

```
panelview turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
type(treat) title("EDR Reform") prepost(off) ylabel("")
```



Change the color schemes for the controls and treated using the `mycolor` option. For example, `PuBu` indicates light purple to blue. Click here for more sequential colors' choice.

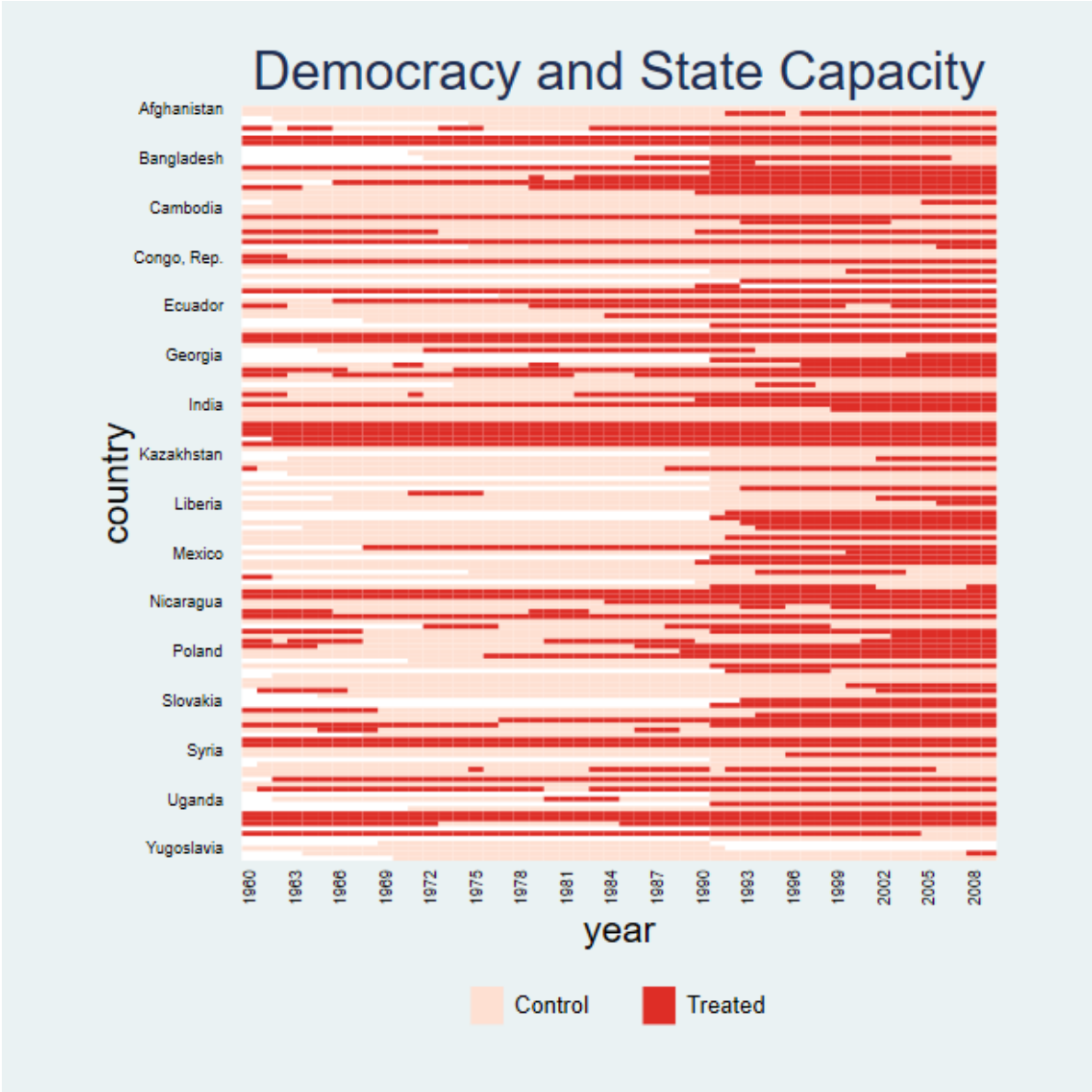
```
*mycolor(PuBu)
panelview turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
type(treat) xtitle("Year") ytitle("State") title("Treatment Status") prepost(off)
mycolor(PuBu) bytiming
```



2.2 Treatment: missing & switch on and off

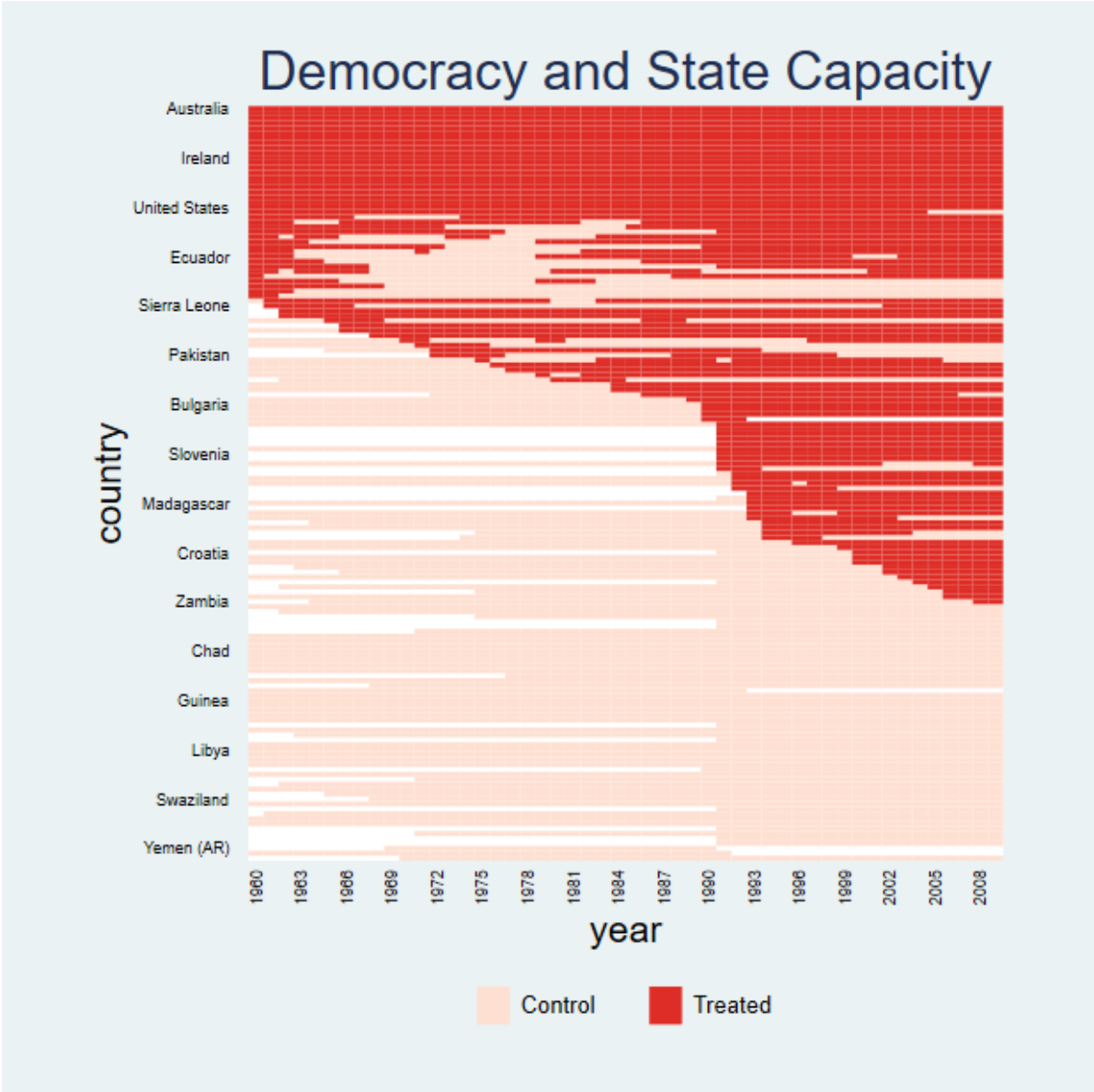
For a panel dataset in which the treatment may switch on and off, we do not differentiate between pre- and post-treatment statuses. To demonstrate how `panelview` can be used in a more general setting, the following plot uses the `capacity` dataset, which is used to investigate the effect of democracy, the treatment, on state capacity, the outcome (Wang and Xu 2018). From the figure below, we see quite a few cases of democratic reversals and that there are many missing values (the white area). We use the `xlabdist` and `ylabdist` option to change the gaps between labels on the x- and y-axes:

```
use capacity.dta, clear
panelview lnpop demo lngdp , i(country) t(year) type(treat) mycolor(Reds)
prepost(off) title("Democracy and State Capacity") xlabdist(3) ylabdist(10)
```



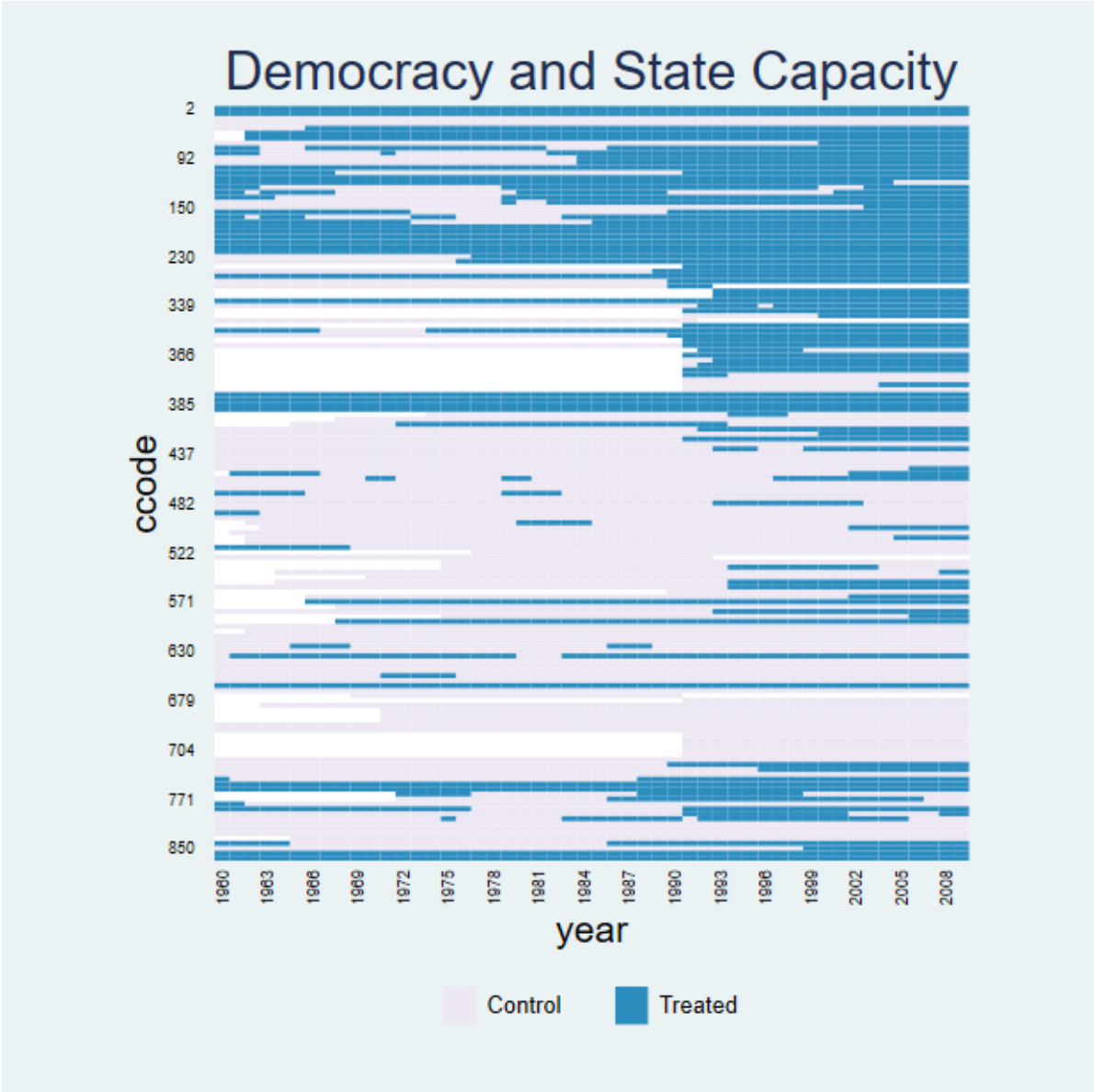
Sorting units based on the first period a unit receives the treatment gives a more appealing visual:

```
*bytiming
panelview lnpop demo lngdp, i(country) t(year) type(treat) mycolor(Reds)
prepost(off) title("Democracy and State Capacity") xlabdist(3) ylabdist(10)
bytiming
```



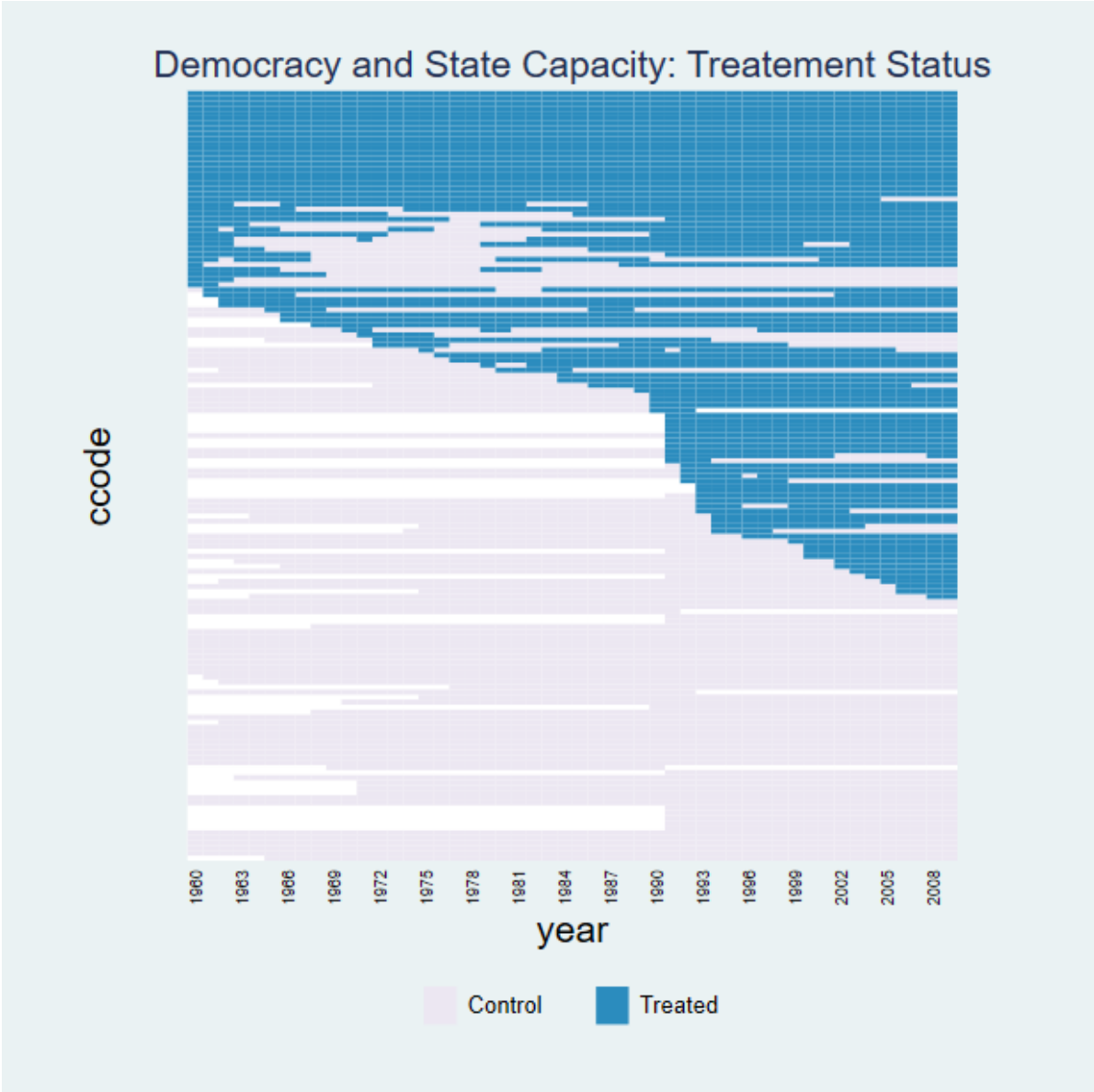
Instead of indicate `country` as units, we use `i(ccode)` to indicate country code as units, which will change the label and sequence in our figure:

```
panelview lnpop demo lngdp, i(ccode) t(year) type(treat) mycolor(PuBu)
prepost(off) title("Democracy and State Capacity") xlabdist(3) ylabdist(10) //If
we set ylabdist(11), the "155" appears at the bottom of ylabel and is hard to
remove, different with R package
```



Sort units based on the first period a unit receives the treatment and use `ylabel(none)` to remove the labels on the y-axis:

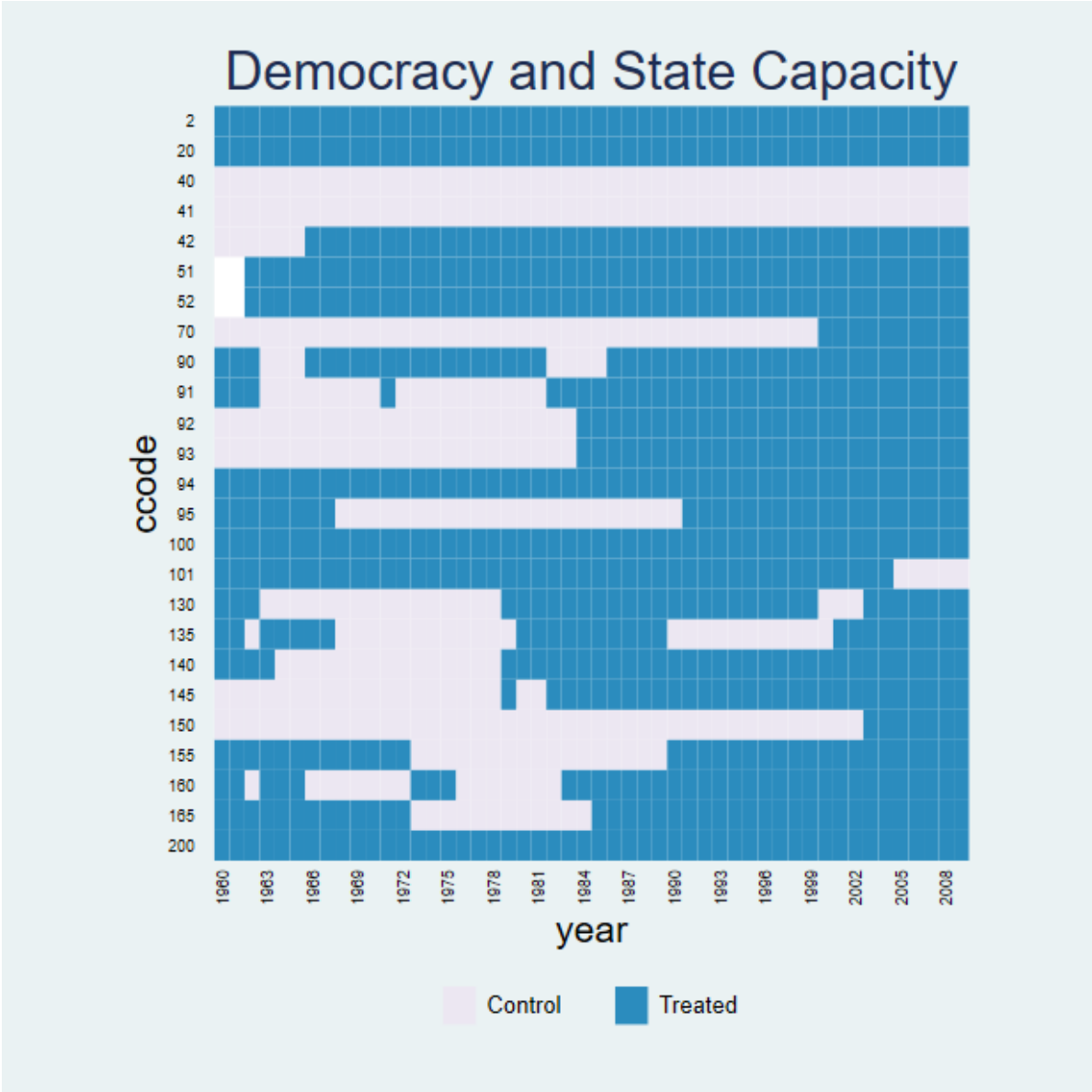
```
*bytiming
panelview lnpop demo lngdp, i(ccode) t(year) type(treat) mycolor(PuBu)
prepost(off) title("Democracy and State Capacity: Treatement Status",
size(medsmall)) bytiming xlabdist(3) ylabel(none)
```



2.3 Plotting a subset of units

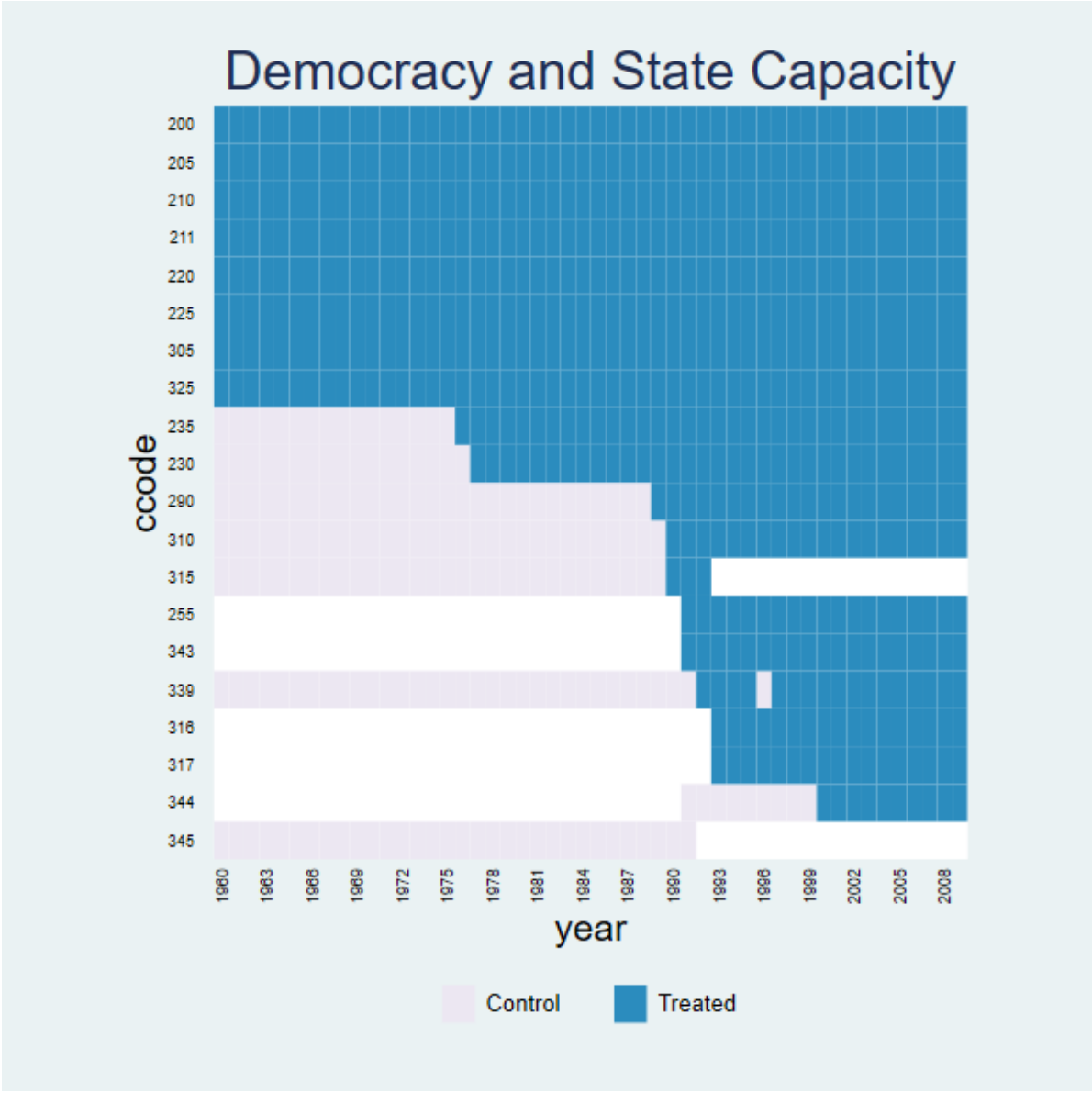
Sometimes a dataset has many units and we only want to take a peak of a subset of the units. **panelView** allows users to specify the units to be shown by the `if` subcommand. Note that if any variable not included in the `varlist` or `i()` / `t()` following `panelview` appears in the `if` or `in` command, we should add such variable into the `varlist` following `panelview`. In the following figure, we plot the treatment statuses of the first 25 units:

```
use capacity.dta, clear
egen ccodeid = group(ccode)
panelview lnpop demo lngdp ccodeid if ccodeid >= 1 & ccodeid <= 26, i(ccode)
t(year) type(treat) mycolor(PuBu) prepost(off) title("Democracy and State
Capacity") xlabdist(3)
```



Sort units based on the first period a unit receives the treatment:

```
*bytiming
panelview lnpop demo lngdp ccodeid if ccodeid >= 26 & ccodeid <= 51, i(ccode)
t(year) type(treat) mycolor(PuBu) prepost(off) title("Democracy and State
Capacity") xlabdist(3) bytiming
```

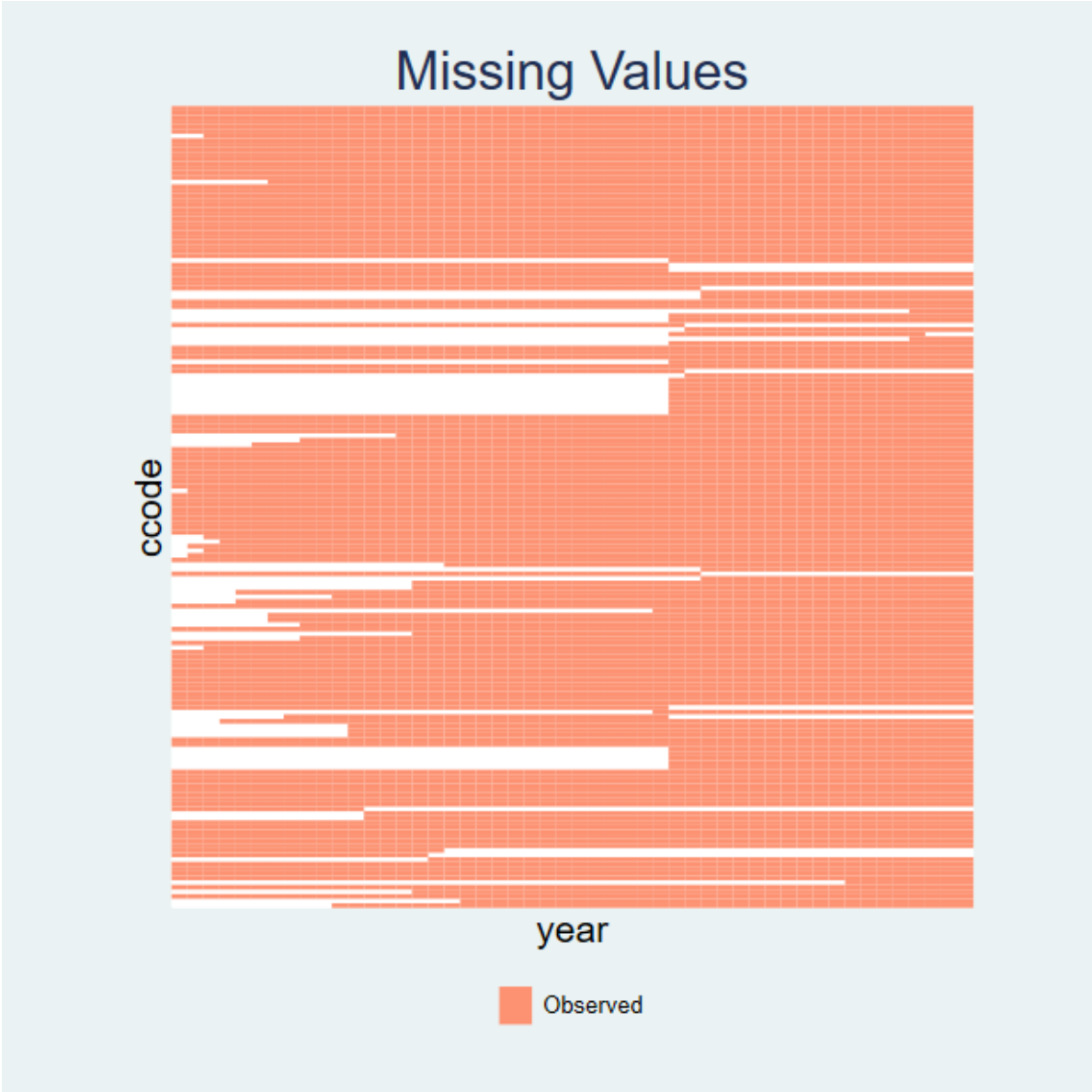


3. Ignoring Treatment Conditions

3.1 `ignoretreat` Subcommand

Omit the treatment variable in a `type(treat)` plot, in which case, the plot will show missing (the white area) and non-missing values only.

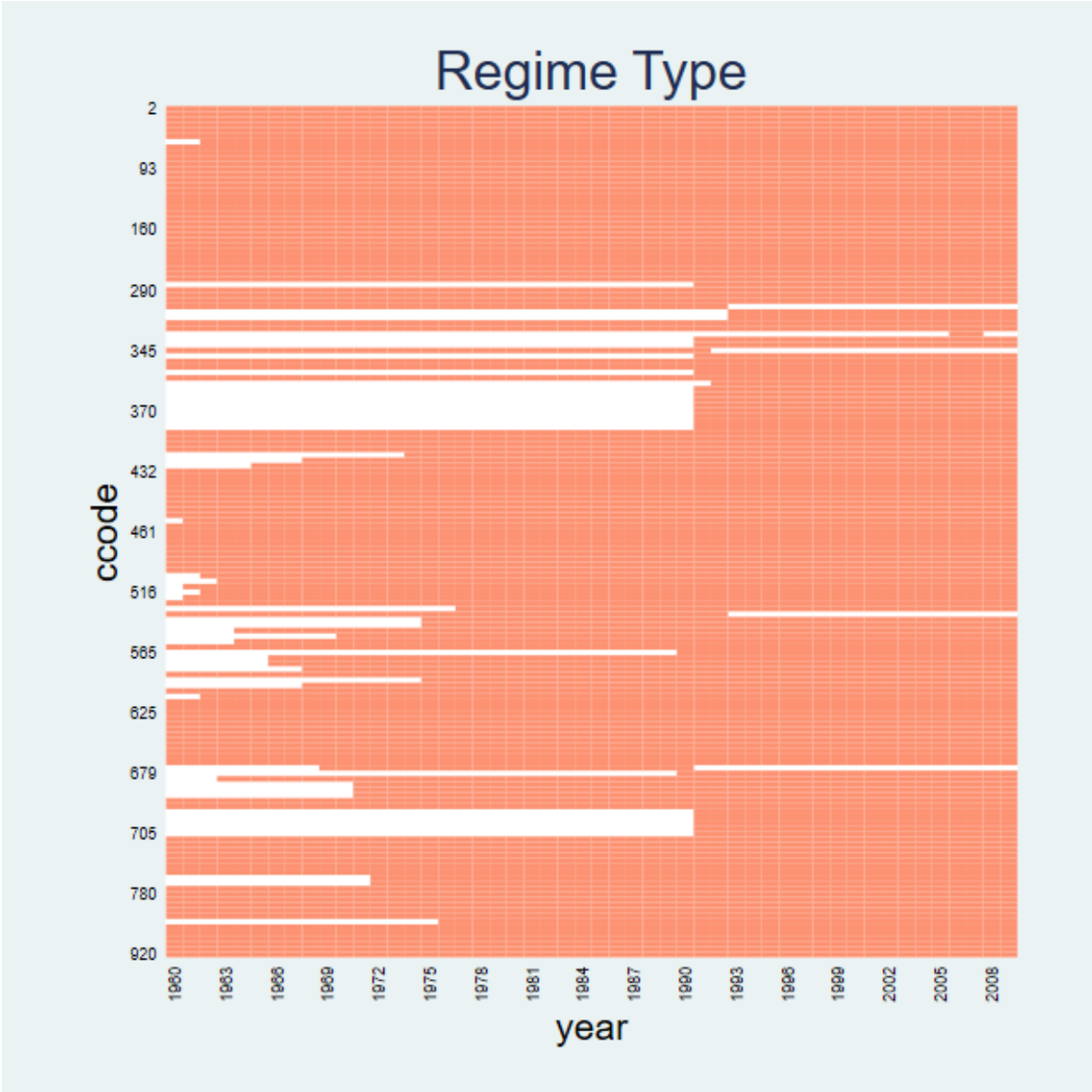
```
use capacity.dta, clear
panelview demo, i(ccode) t(year) type(treat) mycolor(Reds) title("Missing
Values") xlabel(none) ylabel(none) ignoretreat
```



3.2 Treatment level = 1 & Plotting treatment

If the treatment indicator has only 1 level, then treatment status will not be shown on the `type(treat)` plot, which is the same as `ignoretreat`:

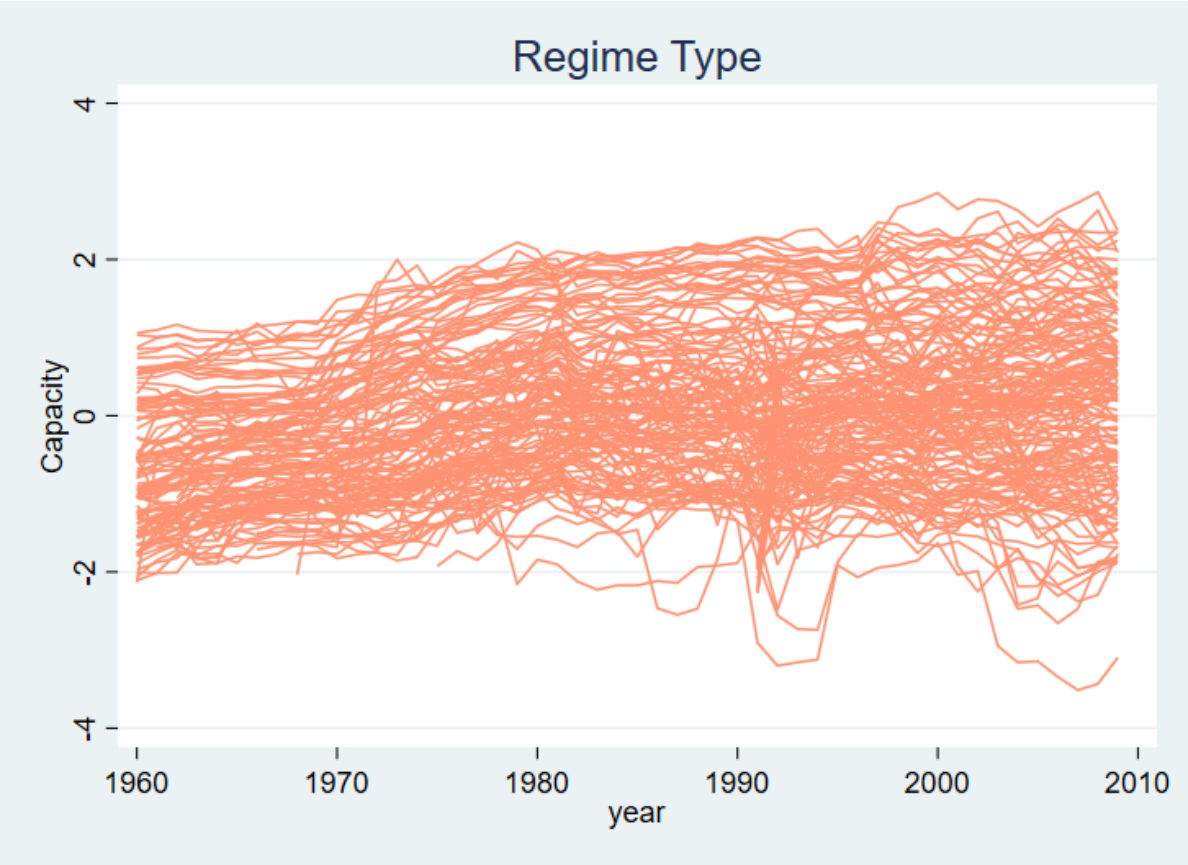
```
use capacity.dta, clear
gen demo2 = 0
panelview Capacity demo2 lngdp, i(ccode) t(year) type(treat) title("Regime Type")
xlabdist(3) ylabdist(11) legend(off) // type(treat) & number of treatment level =
1: same as ignoretreat
```



3.3 Treatment level = 1 & Plotting outcome

If the treatment indicator has only 1 level, then treatment status will not be shown on the `type(outcome)` plot, which is the same as `ignoretreat`:

```
use capacity.dta, clear
gen demo2 = 0
panelview Capacity demo2 lngdp, i(ccode) t(year) type(outcome) title("Regime
Type") legend(off) // type(outcome) & number of treatment level = 1: same as
ignoretreat
```



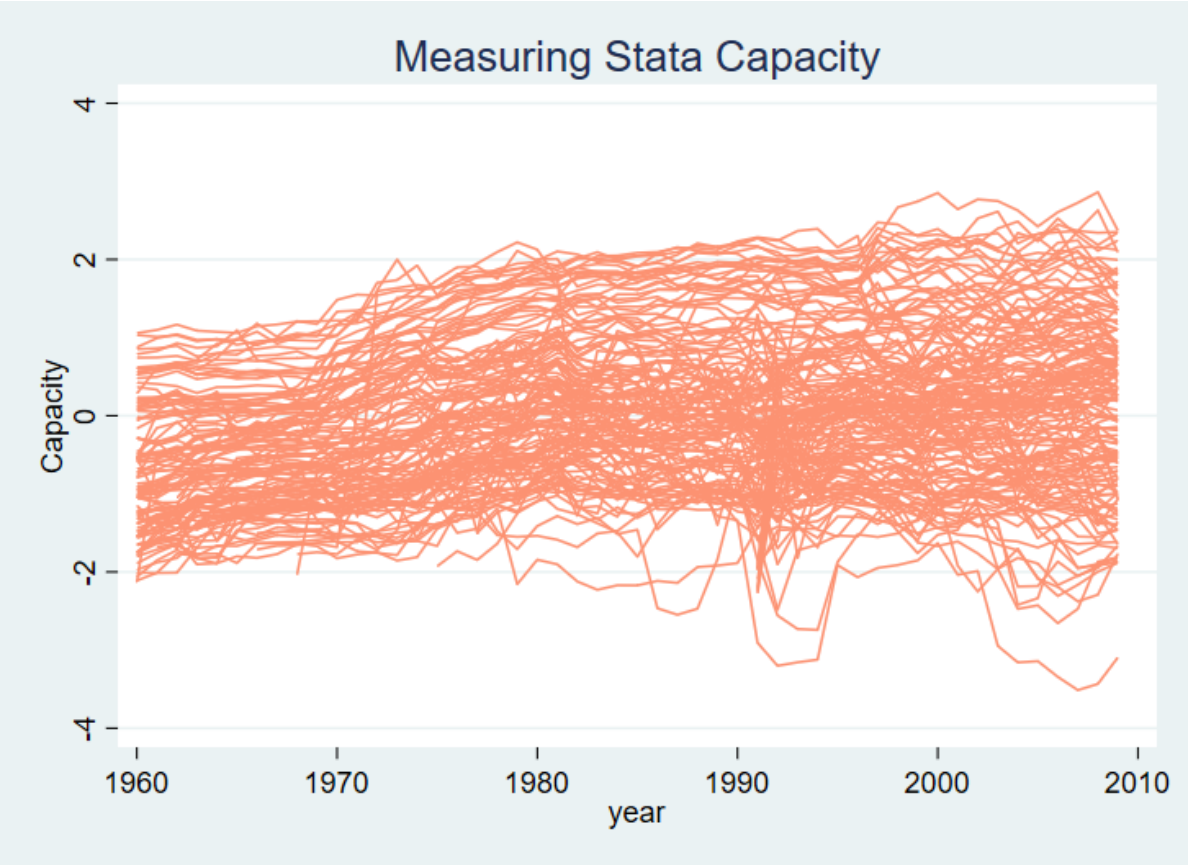
3.4 Plotting outcome & Continuous Treatment / More Than Two Treatment Levels

If the treatment indicator has more than 2 treatment levels or is a continuous variable, then treatment status will not be shown on the `type(outcome)` plot. In other words, `Type(outcome)` combined with `continuoustreat` or > 2 treatment levels is the same as `ignoretreat`.

3.4.1 Continuous Outcomes

With a continuous treatment variable (e.g. `polity2`), the treatment status will not be shown on the `type(outcome)` plot:

```
use capacity.dta, clear
* Continuous Outcome: Capacity; Continuoustreat: polity2
panelView Capacity polity2 lngdp, i(ccode) t(year) type(outcome) continuoustreat
title("Measuring Stata Capacity") legend(off)
```



Same as the following two commands:

```
use capacity.dta, clear
panelView Capacity demo lngdp, i(ccode) t(year) type(outcome) title("Measuring
Stata Capacity") ignoretreat legend(off)
```

```
* Treatment indicator has more than 2 treatment levels
* Continuous Outcome: Capacity
use capacity.dta, clear
gen demo2 = 0
replace demo2 = -1 if polity2 < -0.5
replace demo2 = 1 if polity2 > 0.5
tab demo2, m
panelView Capacity demo2 lngdp, i(ccode) t(year) type(outcome) title("Measuring
Stata Capacity") prepost(off) legend(off) // number of treatment level = 3
```

3.4.2 Discrete Outcomes

When the number of treatment levels is more than two, the treatment status will not be shown on the `type(outcome)` plot:

```
use simdata.dta, replace
replace D = 2 if time < 5
tab D, m
panelView Y D, type(outcome) i(id) t(time) mycolor(Greens) discreteoutcome
title("Raw Data") prepost(off) // number of treatment level = 3
```

Same as the following two commands:

```
use simdata.dta, replace
panelView Y D, type(outcome) i(id) t(time) mycolor(Greens) discreteoutcome
title("Raw Data") ignoretreat
```

```
use simdata.dta, replace
range x 0 1
panelView Y x, type(outcome) i(id) t(time) mycolor(Greens) discreteoutcome
title("Raw Data") prepost(off) continuoustreat // continuous treatment
```

4. More than Two Treatment Conditions

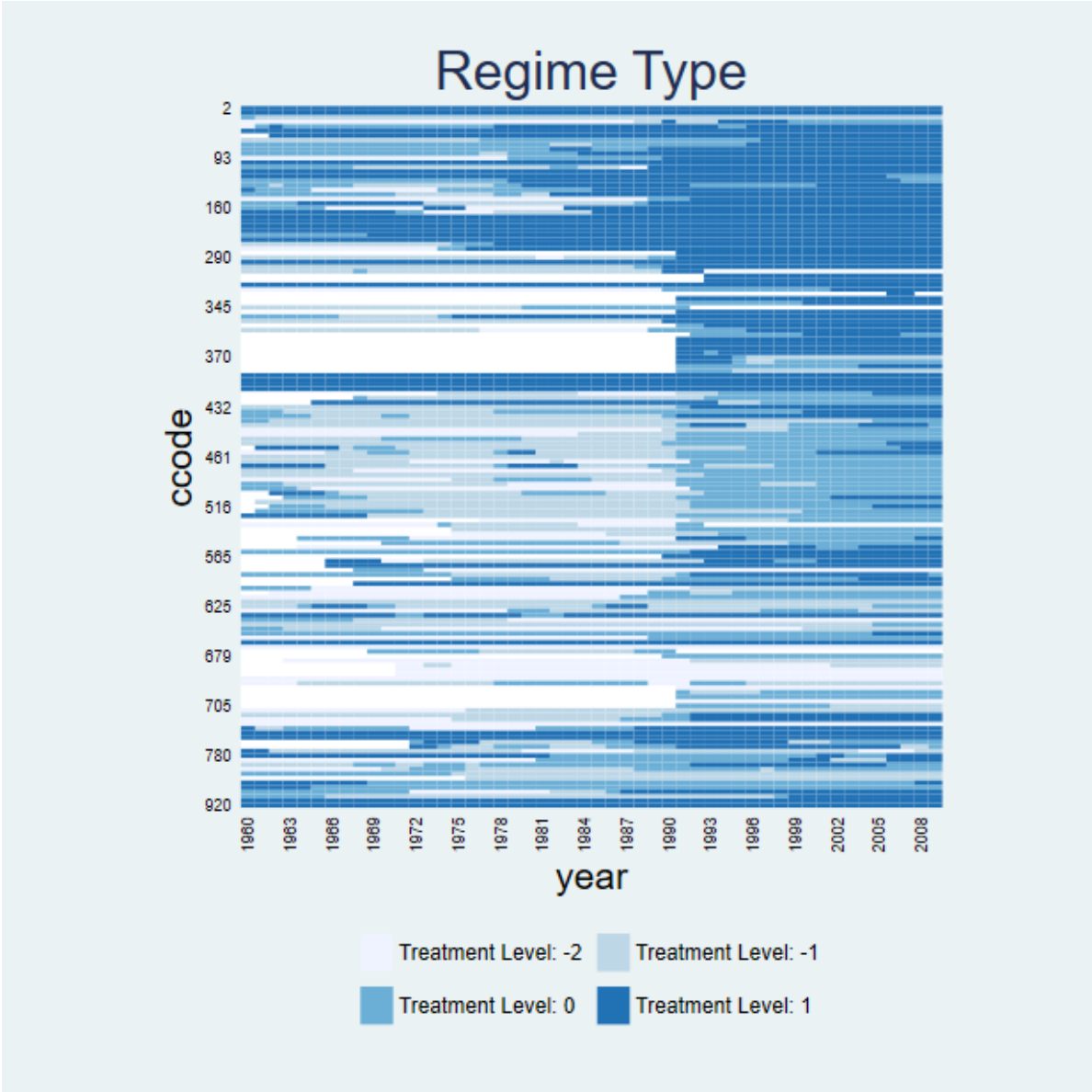
4.1 Treatment level = 3

panelView supports TSCS data with more than 2 treatment levels. For example, we create a measure of regime type with three treatment levels:

```
use capacity.dta, clear
gen demo2 = 0
replace demo2 = -1 if polity2 < -0.5
replace demo2 = 1 if polity2 > 0.5
panelView Capacity demo2 lngdp, i(ccode) t(year) type(treat) title("Regime Type")
xlabdist(3) ylabdist(11) prepost(off) mycolor(Blues) // type(treat) & number of
treatment level = 3
```

4.2 Treatment level = 4

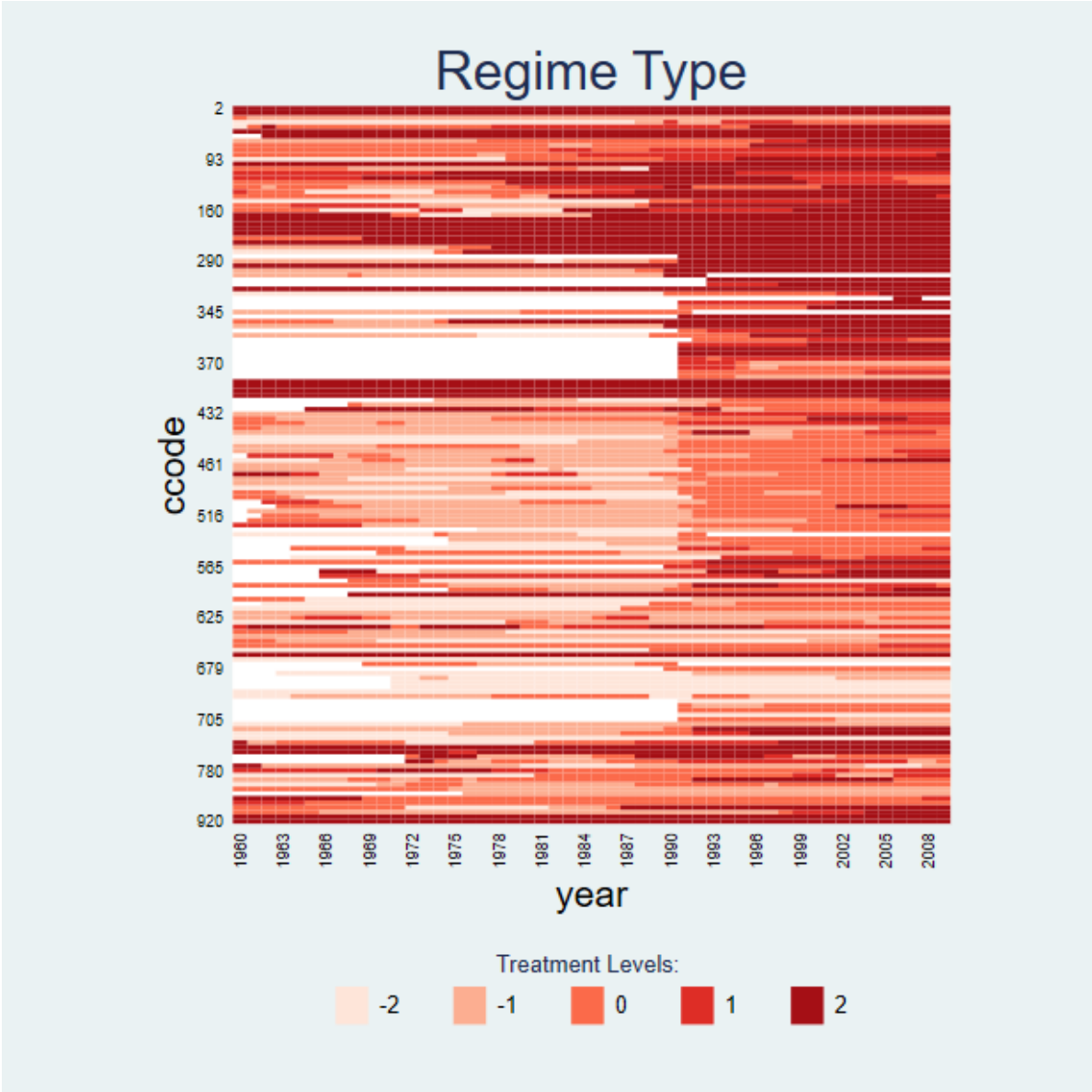
```
use capacity.dta, clear
gen demo2 = 0
replace demo2 = -2 if polity2 < -0.7
replace demo2 = -1 if polity2 < -0.5 & polity2 > -0.7
replace demo2 = 1 if polity2 > 0.5
panelview Capacity demo2 lngdp, i(ccode) t(year) type(treat) title("Regime Type")
xlabdist(3) ylabdist(11) prepost(off) mycolor(Blues) // number of treatment level
= 4
```



4.3 Treatment level >= 5

If the number of treatment levels is greater than 5, then the treatment indicator will be regarded as a continuous variable.

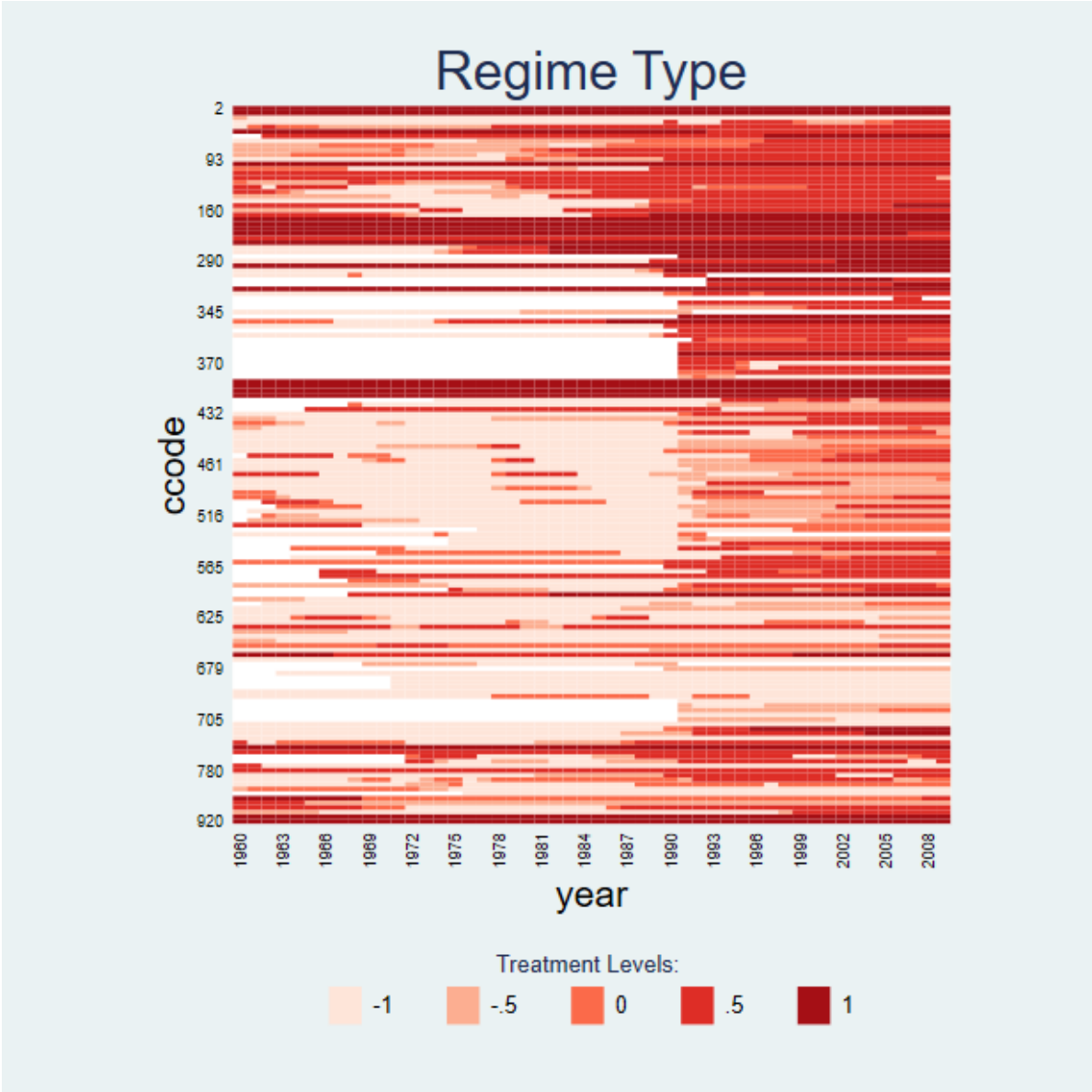
```
use capacity.dta, clear
gen demo2 = 0
replace demo2 = -2 if polity2 < -0.7
replace demo2 = -1 if polity2 < -0.5 & polity2 > -0.7
replace demo2 = 1 if polity2 > 0.5 & polity2 < 0.7
replace demo2 = 2 if polity2 > 0.7
tab demo2, m
panelview Capacity demo2 lngdp, i(ccode) t(year) type(treat) title("Regime Type")
xlabdist(3) ylabdist(11) prepost(off) continuoustreat
```



4.4 Continuous treatment

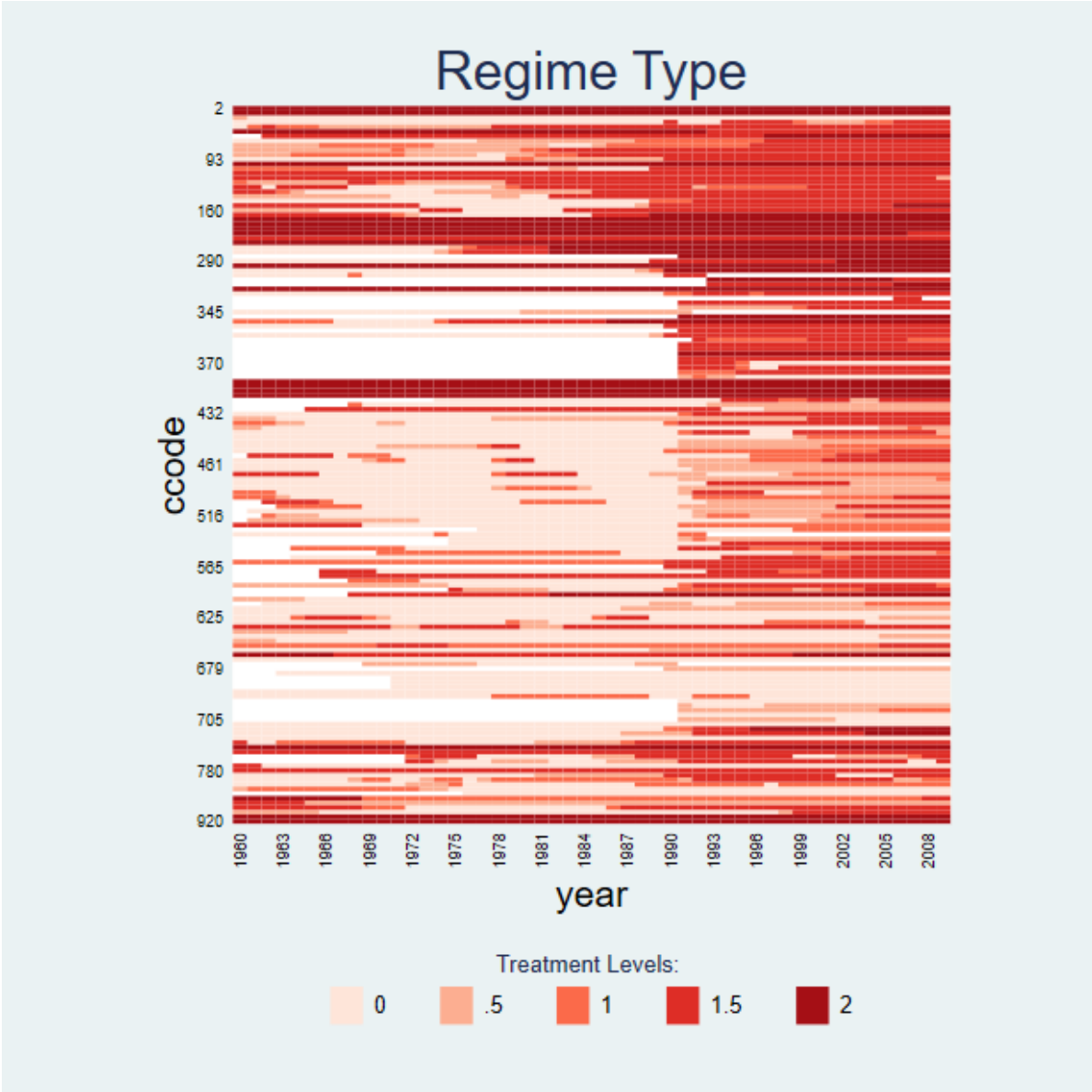
Plot the continuous treatment variable by `continuoustreat`. Note that `continuoustreat` need to combine with `prepost(off)`.

```
use capacity.dta, clear
panelview lngdp polity2, i(ccode) t(year) type(treat) continuoustreat
mycolor(Reds) prepost(off) title("Regime Type") xlabdist(3) ylabdist(11)
```



If we change the level of the continuous treatment variable, the legend will modify correspondingly:

```
use capacity.dta, clear
replace polity2 = polity2 + 1
panelview lngdp polity2, i(ccode) t(year) type(treat) continuoustreat
mycolor(Reds) prepost(off) title("Regime Type") xlabdist(3) ylabdist(11)
```



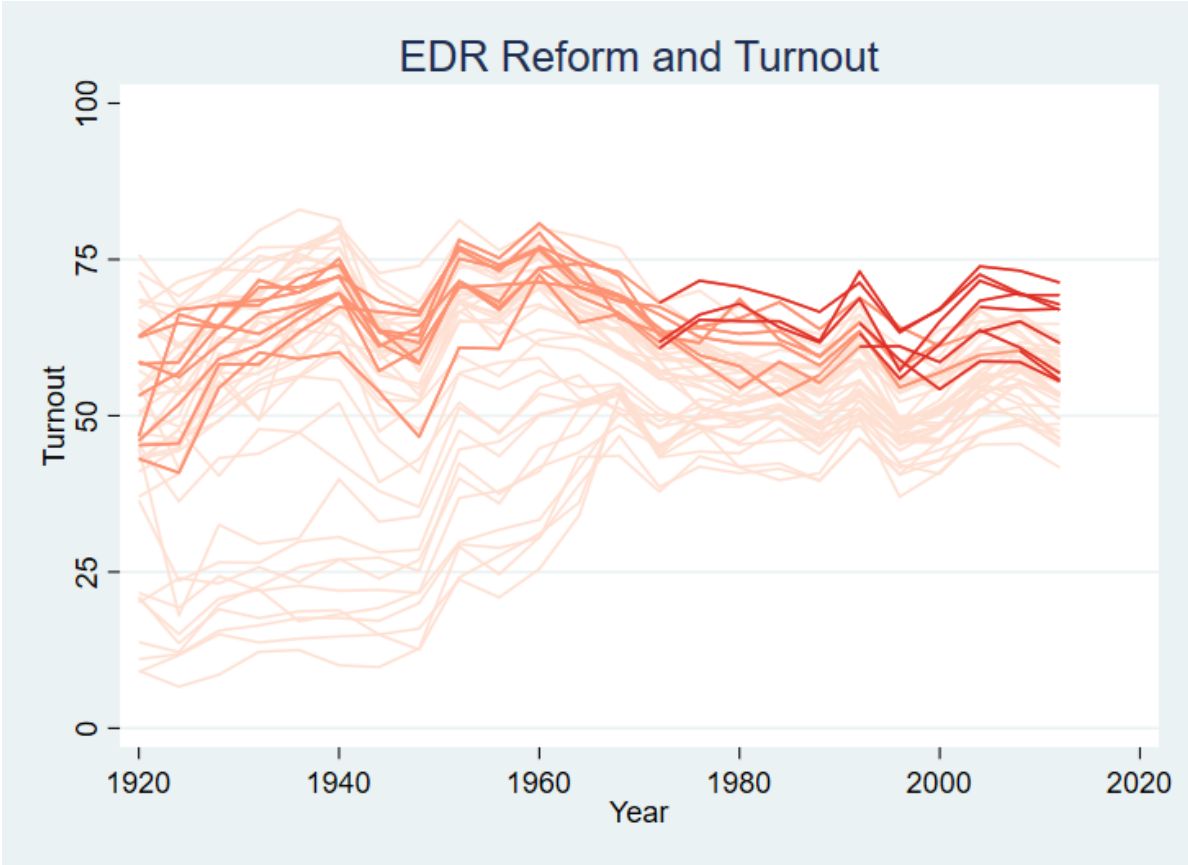
5. Continuous Outcomes

The second functionality of **panelView** is to show the raw outcome variable of a panel dataset in a time-series fashion. The syntax is very similar except that we need to specify `type(outcome)`. Different colors represent different treatment conditions.

5.1 Continuous Outcomes

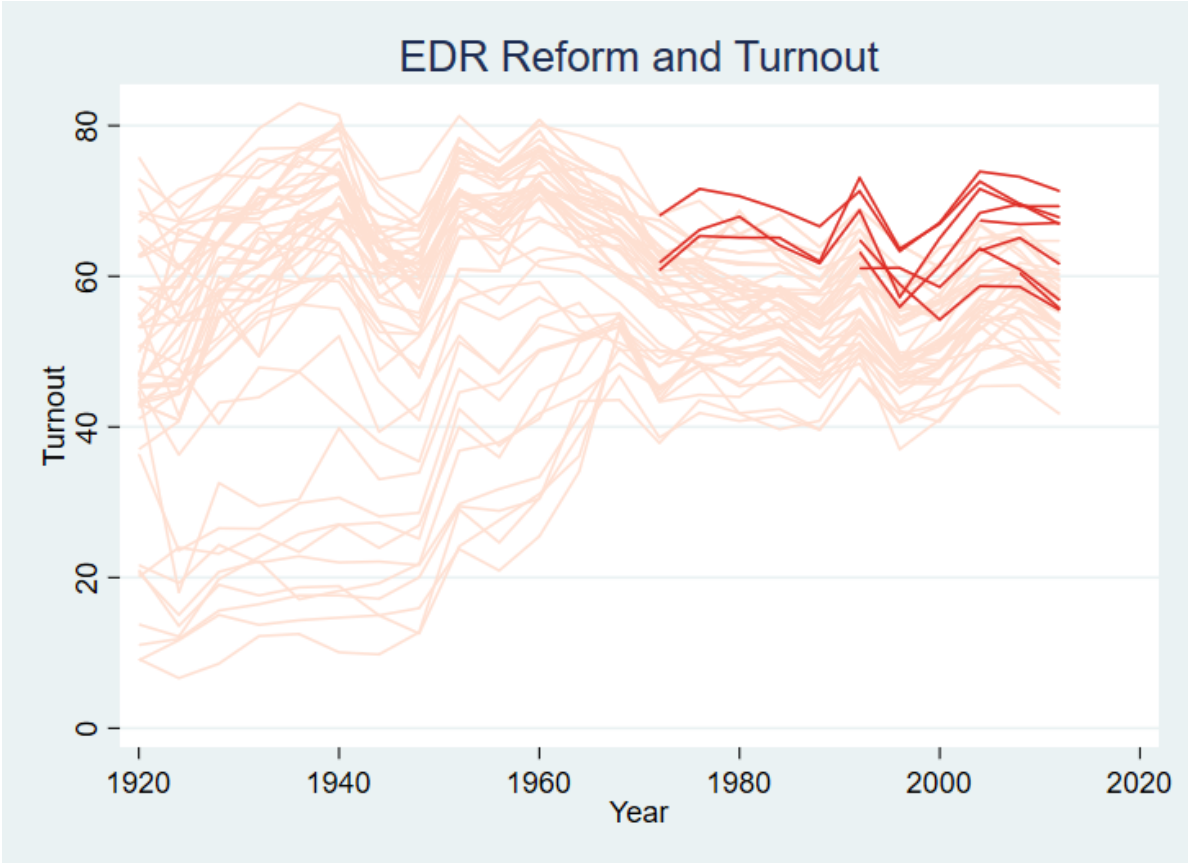
Note that we paint the period right before when the treatment begin as treated period. Different with `type(treat)`, `type(outcome)` does not need `xlabdist` and `ylabdist`. If needed, we should use `xlabel` and `ylabel` instead.

```
* Continuous outcome: turnout: 0-100; Discrete Treatment: policy_edr: 0/1
use turnout.dta, clear
panelView turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
type(outcome) xtitle("Year") ytitle("Turnout") title("EDR Reform and Turnout")
ylabel(0 (25) 100)
```



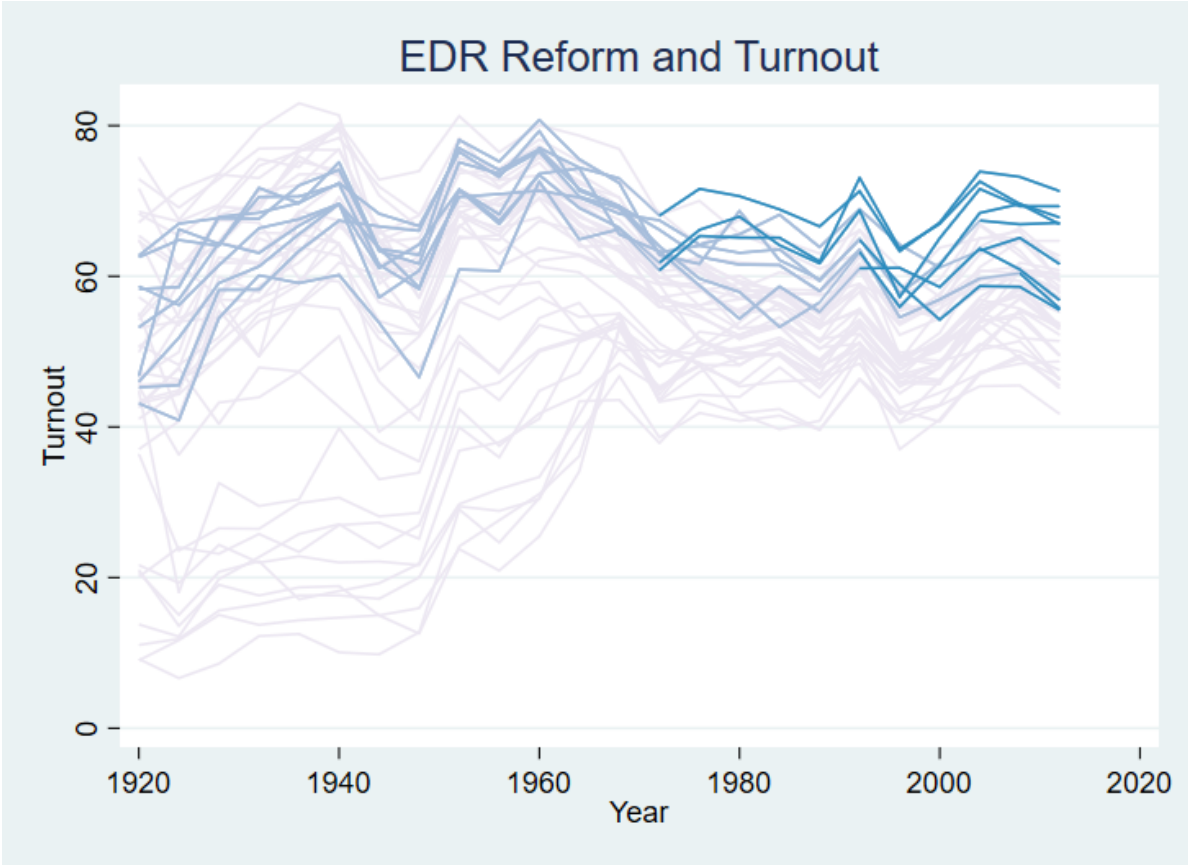
Not distinguish the pre- and post-treatment periods for treated units:

```
*prepost(off)
panelview turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
type(outcome) xtitle("Year") ytitle("Turnout") title("EDR Reform and Turnout")
prepost(off)
```



Apply the light purple to blue theme by specifying `mycolor(PuBu)`:

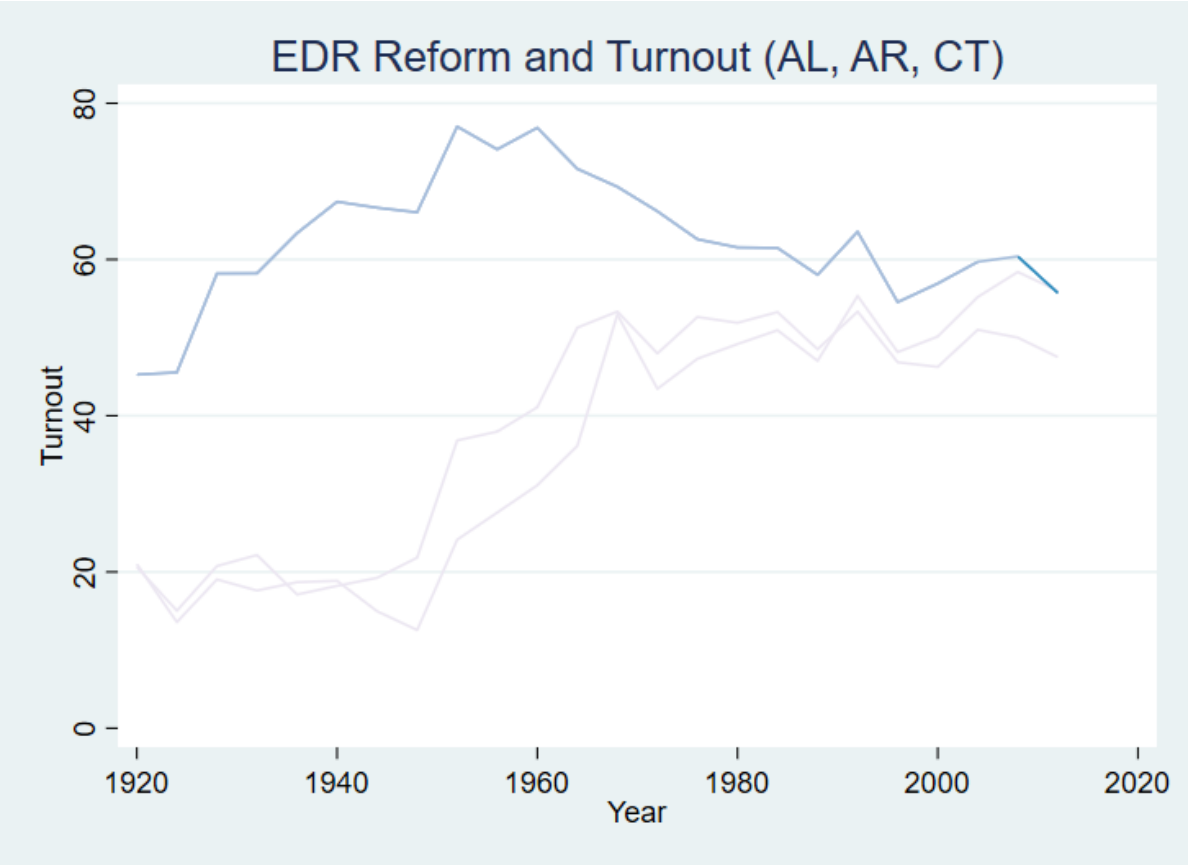
```
use turnout.dta, clear
panelview turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
type(outcome) xtitle("Year") ytitle("Turnout") title("EDR Reform and Turnout")
mycolor(PuBu)
```



5.2 Specify which unit(s) we want to take a look at

We can specify which unit(s) we want to take a look at:

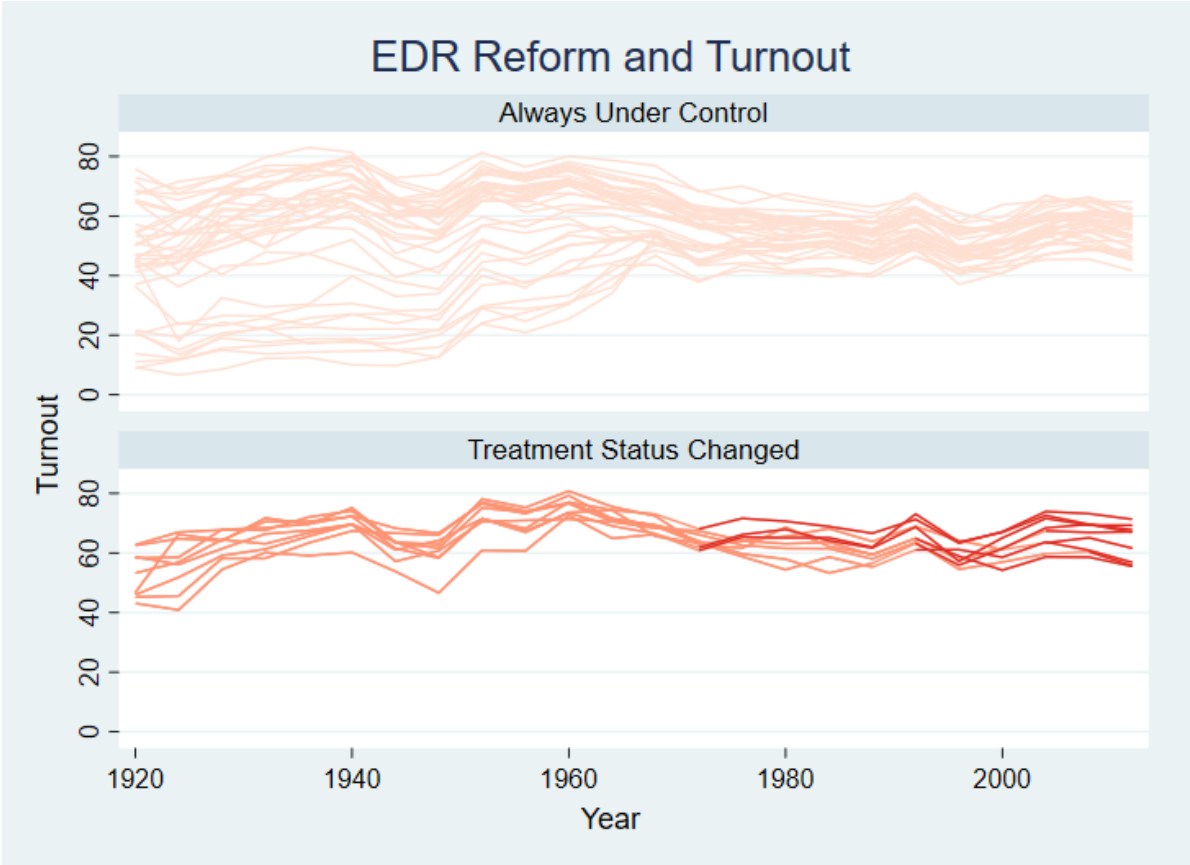
```
use turnout.dta, clear
panelview turnout policy_edr policy_mail_in policy_motor if abb == 1|abb == 2|abb
== 6, i(abb) t(year) type(outcome) xtitle("Year") ytitle("Turnout") title("EDR
Reform and Turnout (AL, AR, CT)") mycolor(PuBu)
```



5.3 Put each unit into different groups, then plot respectively

To better understand the data, sometimes we want to plot the outcome based on whether the treatment status has changed during the observed time period. We can simply add an option `bytreatgroup`. The algorithm will analyze the data and automatically put each unit into different groups, e.g. (1) Always treated, (2) always in control, (3) treatment status changed.

```
use turnout.dta, clear
panelview turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
type(outcome) xtitle("Year") ytitle("Turnout") by(, title("EDR Reform and
Turnout")) bytreatgroup xlabel(1920 (20) 2000)
```



6. Discrete Outcomes

We can accommodate discrete variables by setting `discreteoutcome`. Below is an example using the `simdata` dataset, in which the outcome variable takes three values: 0, 1, and 2.

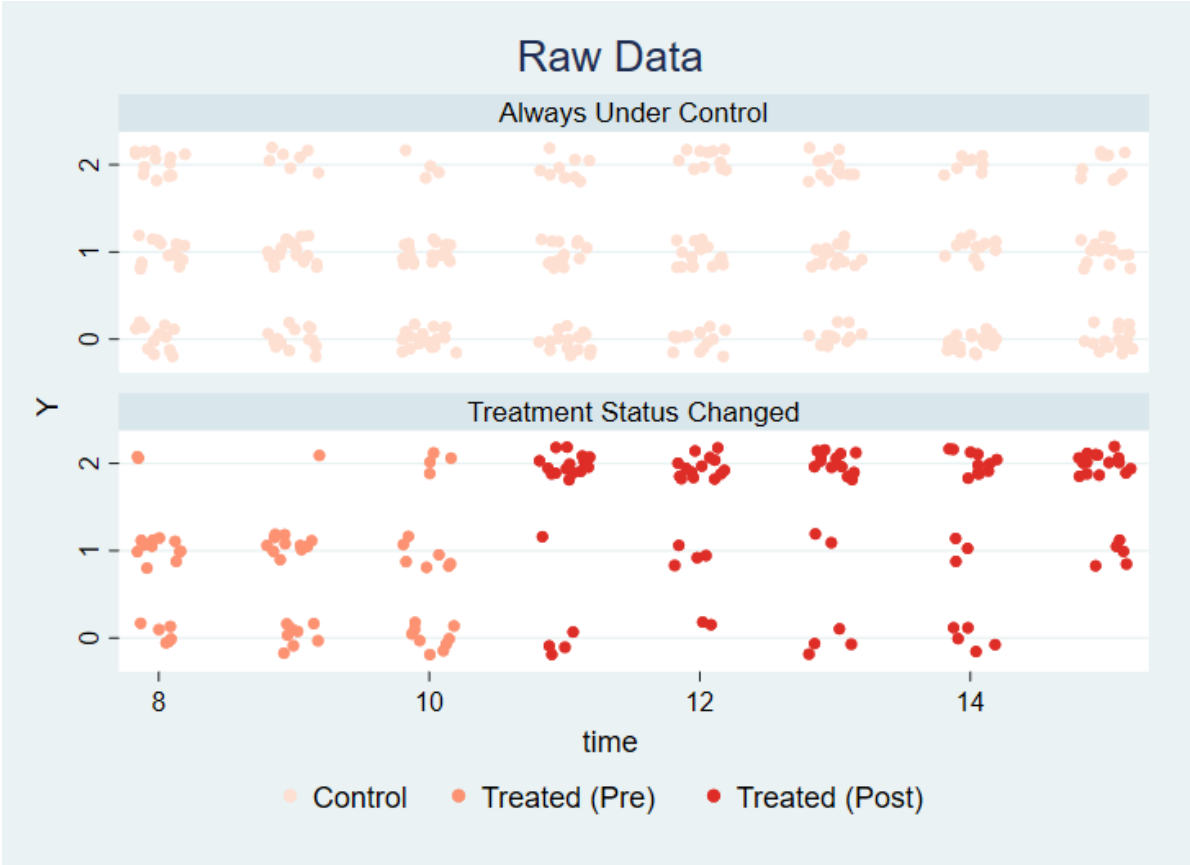
6.1 Discrete Outcomes

```
use simdata.dta, replace
panelview Y D if time >= 8 & time <= 15, type(outcome) i(id) t(time)
mycolor(Reds) discreteoutcome title("Raw Data") xlabel(8 (2) 15) ylabel(0 (1) 2)
```

6.2 Put each unit into different groups, then plot respectively

We split the sample based on changes in treatment status:

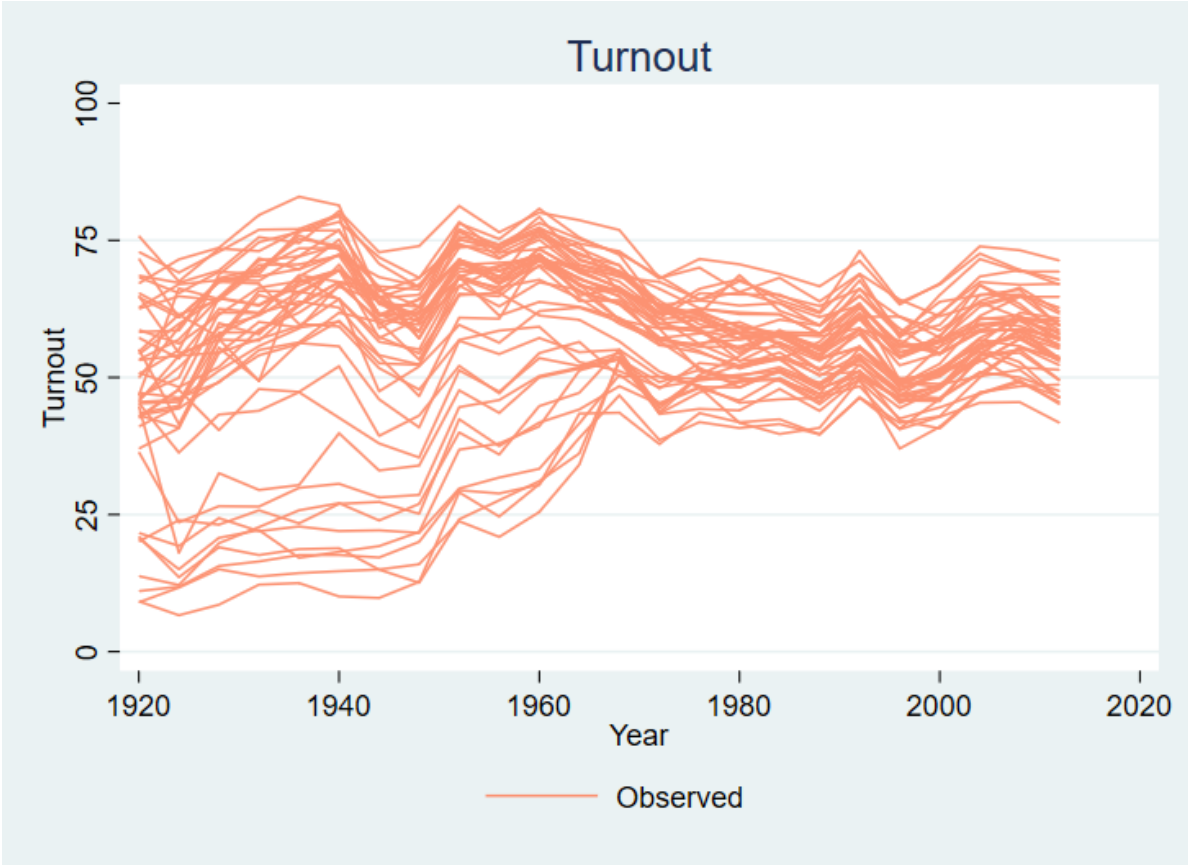
```
use simdata.dta, replace
panelview Y D if time >= 8 & time <= 15, type(outcome) i(id) t(time)
discreteoutcome by(,title("Raw Data")) xlabel(8 (2) 15) ylabel(0 (1) 2)
bytreatgroup
```



7. Plotting any variable in a panel dataset

Plot an outcome variable (or any variable) in a panel dataset by `type(outcome)` and `ignoretreat`:

```
use turnout.dta, clear
panelview turnout, i(abb) t(year) type(outcome) xtitle("Year") ytitle("Turnout")
title("Turnout") ylabel(0 (25) 100) ignoretreat
```



8. Plotting Y and D against time in the same graph

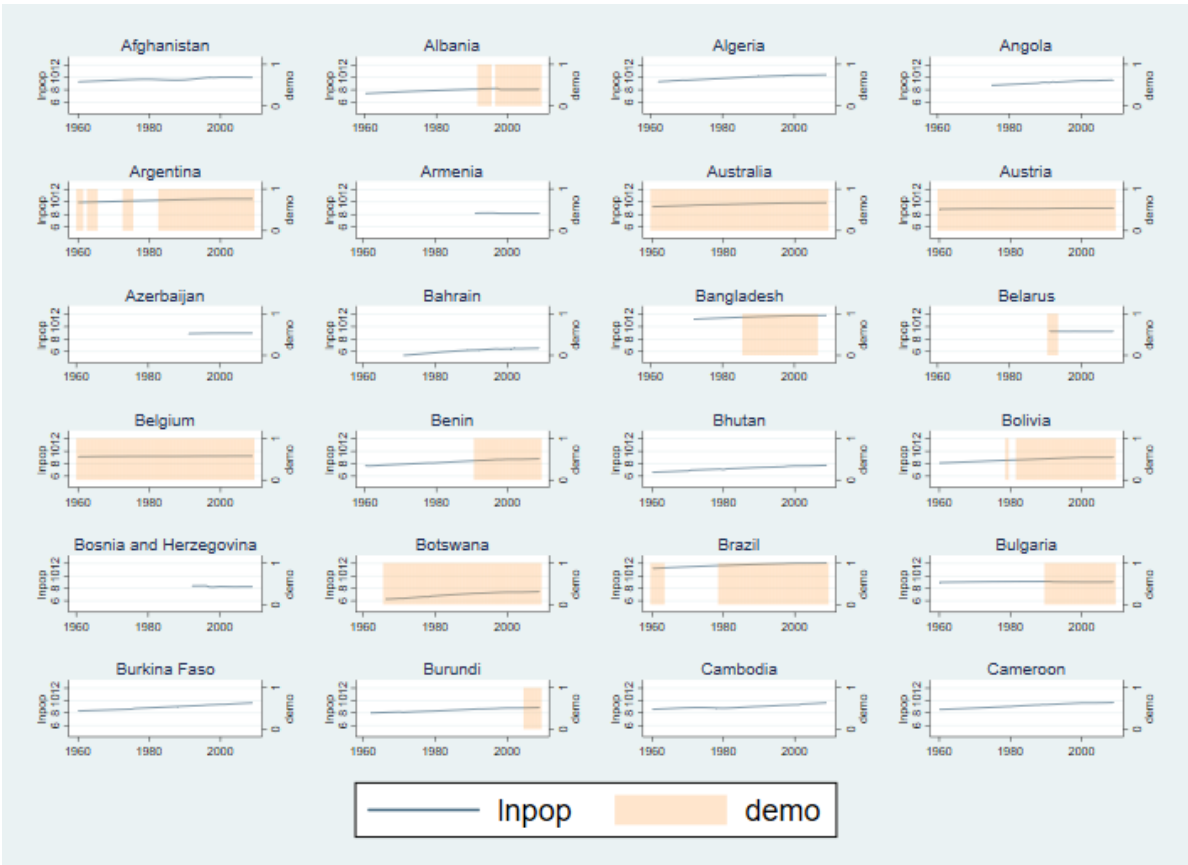
Visualize time series of the outcome and treatment for each unit in one figure by not specifying `type()`. For continuous treatment, we use line plot; for discrete treatment, we use bar plot. If one want to visualize the zero line with discrete treatment, please add `linediscretreat`. No matter the outcome is continuous or discrete, line plot is applied. The left y axis indicates outcome label, and the right y axis indicates treatment label.

8.1 Plot by each unit

Below are two examples with continuous outcome and discrete treatment variable. We arrange four subgraphs in one row:

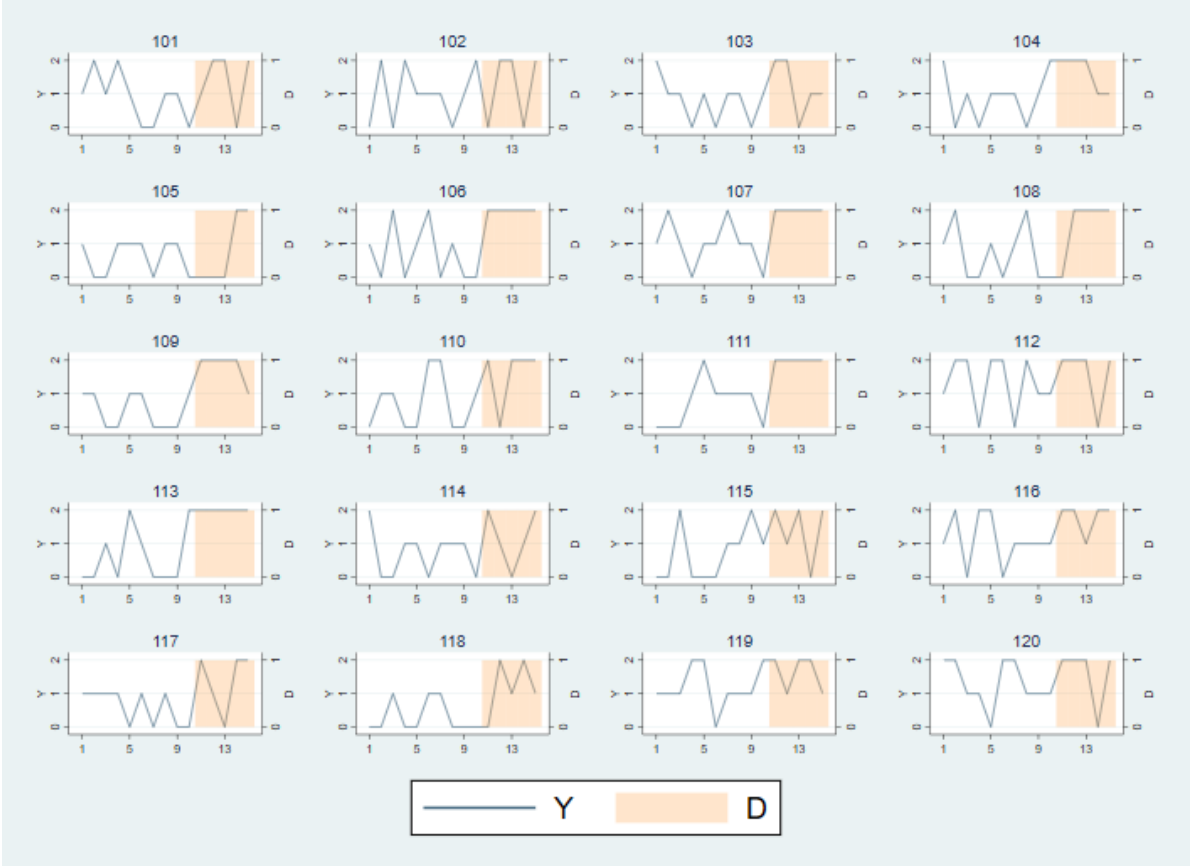
```
/***** 1. Y: continuous; D: dummy *****/
use turnout.dta, clear
panelview turnout policy_edr policy_mail_in policy_motor if abb >= 1 & abb <= 12,
i(abb) t(year) xlabdist(10)
```


```
use turnout.dta, clear
panelview turnout policy_edr policy_mail_in policy_motor if abb >= 1 & abb <= 12,
i(abb) t(year) xlabdist(10)
```



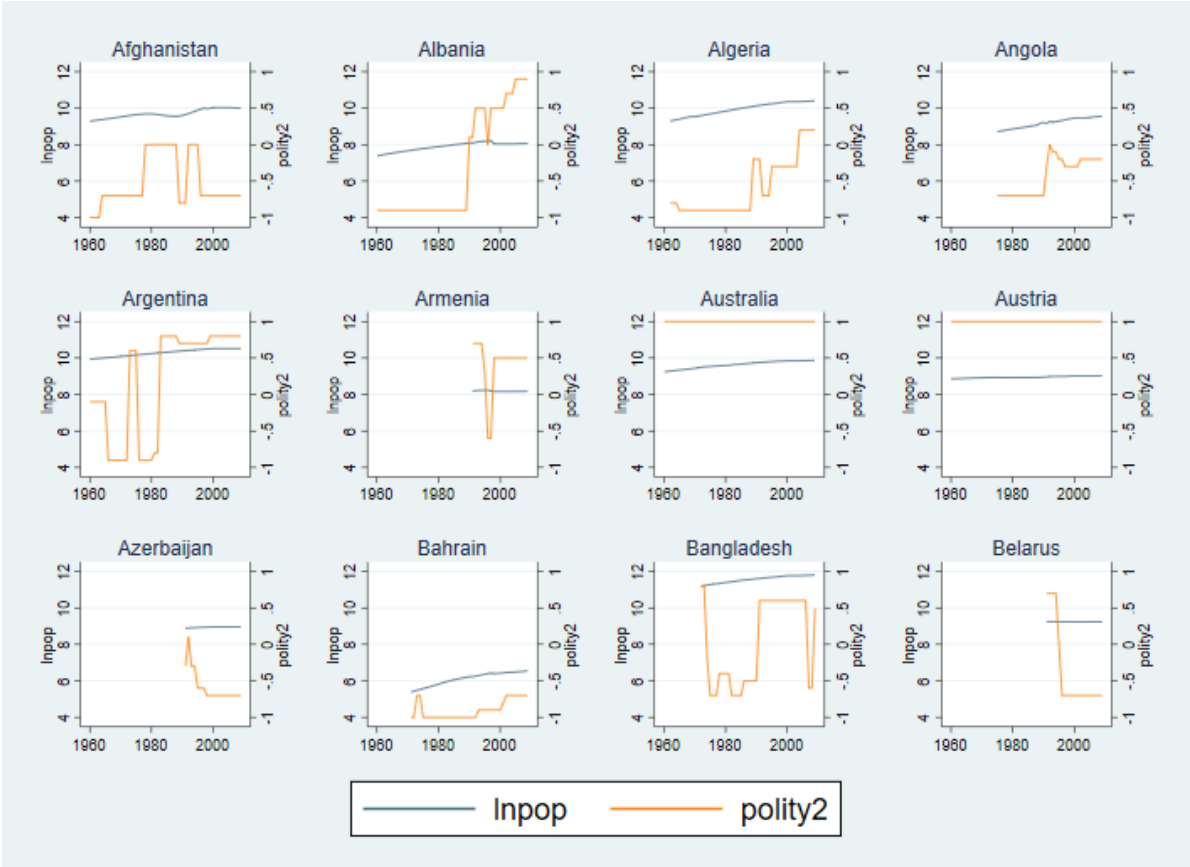
If the outcome is discrete, we can plot outcome and treatment against time in the same figure by indicating `discreteoutcome`:

```
/***** 2. Y: Discrete; D: dummy *****/
use simdata.dta, replace
panelview Y D if id >= 101 & id <= 120,i(id) t(time) discreteoutcome xlabdist(4)
```



When treatment variable is continuous, we need to add the subcommands of `continuoustreat` and `prepost(off)`:

```
/***** 3. Y: continuous; D: continuous *****/
use capacity.dta, clear
panelview lnpop polity2 if country >= 1 & country <= 12, i(country) t(year)
continuoustreat prepost(off) xlabdist(20)
```



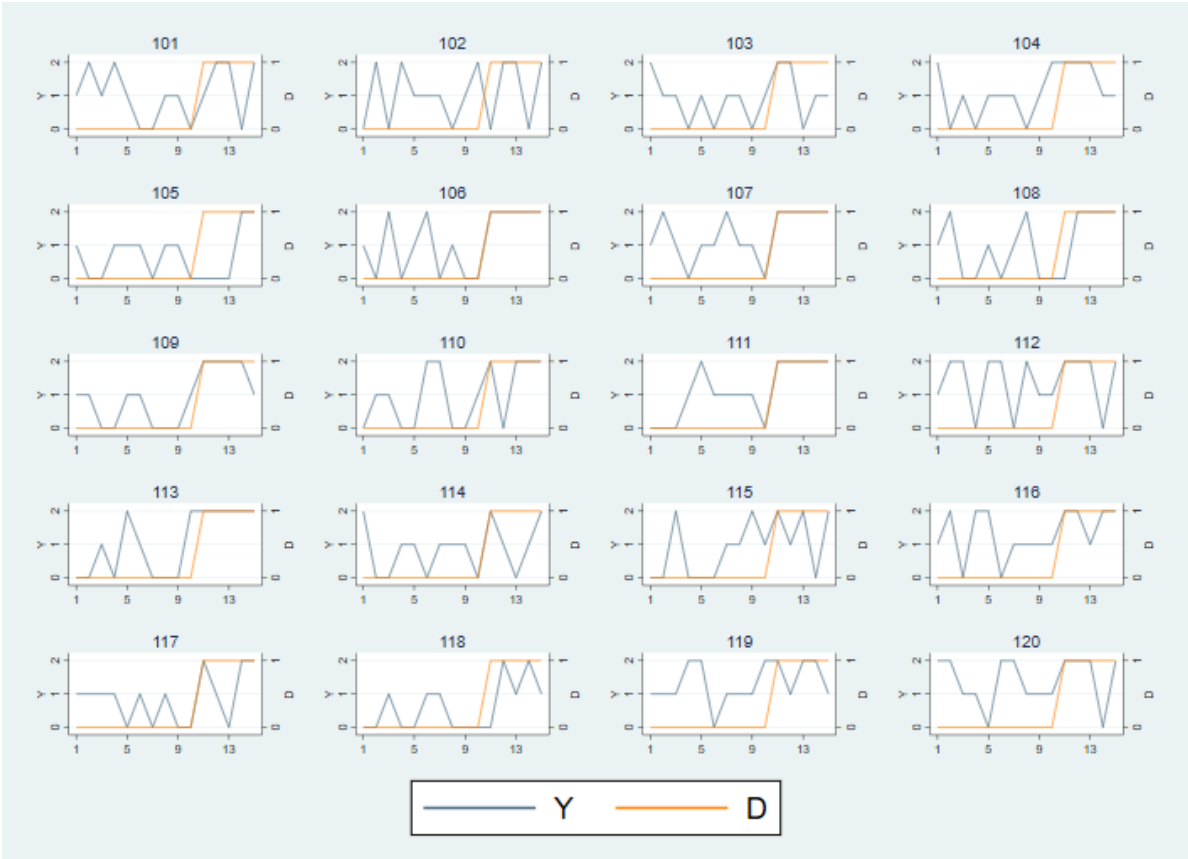
In the last situation, we plot discrete outcome and continuous treatment by options `continuoustreat` and `discreteoutcome`:

```
/***** 4. Y: Discrete; D: continuous *****/
use simdata.dta, replace
range x 0 1
panelview Y x if id >= 101 & id <= 112, i(id) t(time) prepost(off)
continuoustreat discreteoutcome xlabdist(4)
```

To visualize the zero level with discrete treatment, we add `linediscretreat` to plot treatment lines instead of bars:

```
/***** Line the discete treatment *****/
* Y: continuous; D: dummy
use turnout.dta, clear
panelview turnout policy_edr policy_mail_in policy_motor if abb >= 1 & abb <= 12,
i(abb) t(year) xlabdist(10) linediscretreat
```


```
*Y: Discrete; D: dummy
use simdata.dta, replace
panelview Y D if id >= 101 & id <= 120,i(id) t(time) discreteoutcome xlabdist(4)
linediscretreat
```

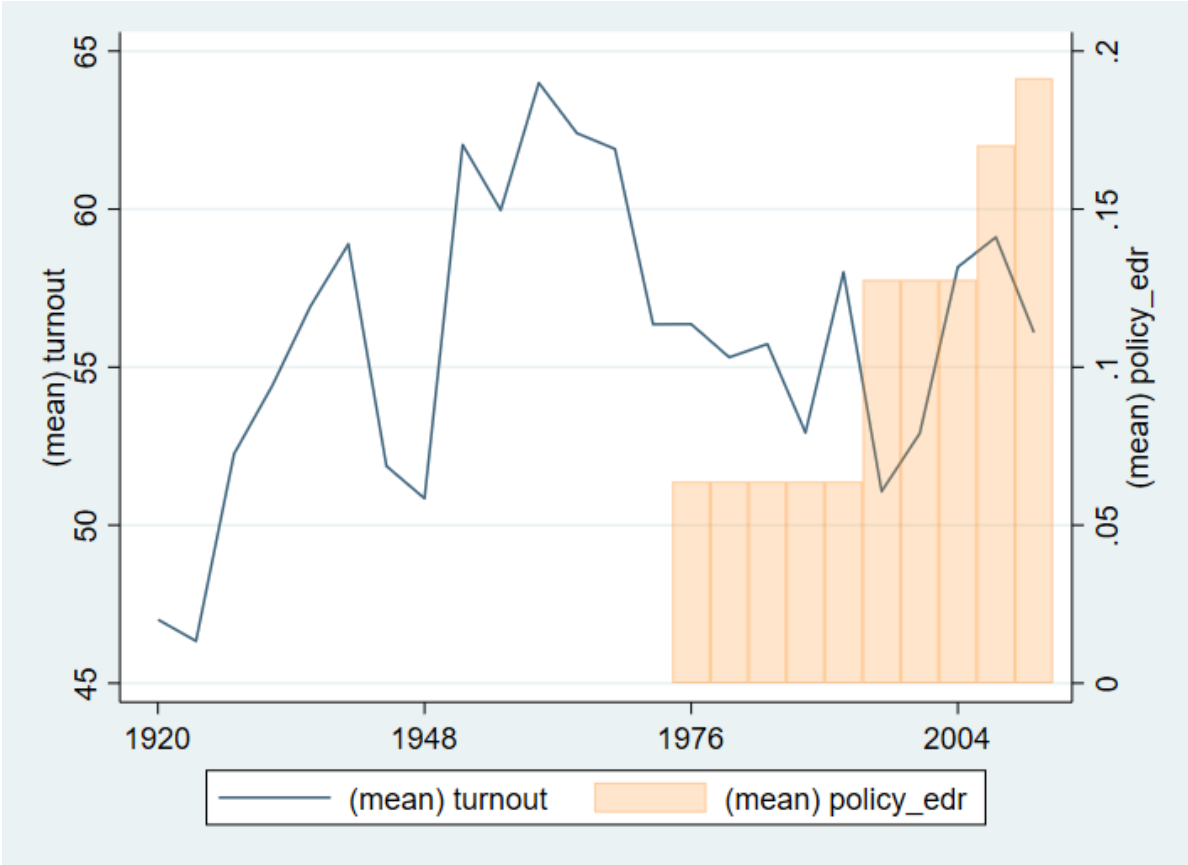


8.2 Plot average time series for all units

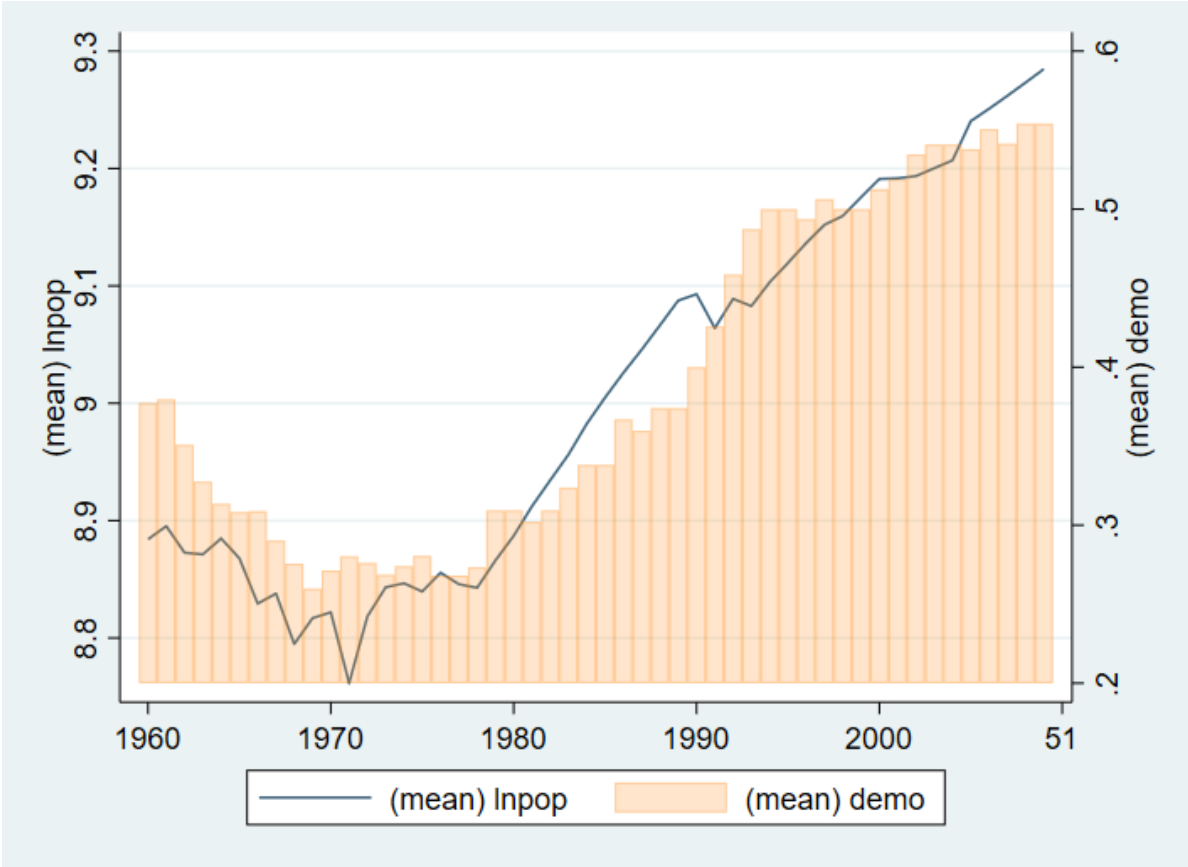
We plot mean D and Y against time in the same graph by option `allunitsplot`.

With continuous outcome and discrete treatment variable, here are two examples:

```
/***** 1. Y: continuous; D: dummy *****/
use turnout.dta, clear
panelview turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
xlabdist(7) allunitsplot
```

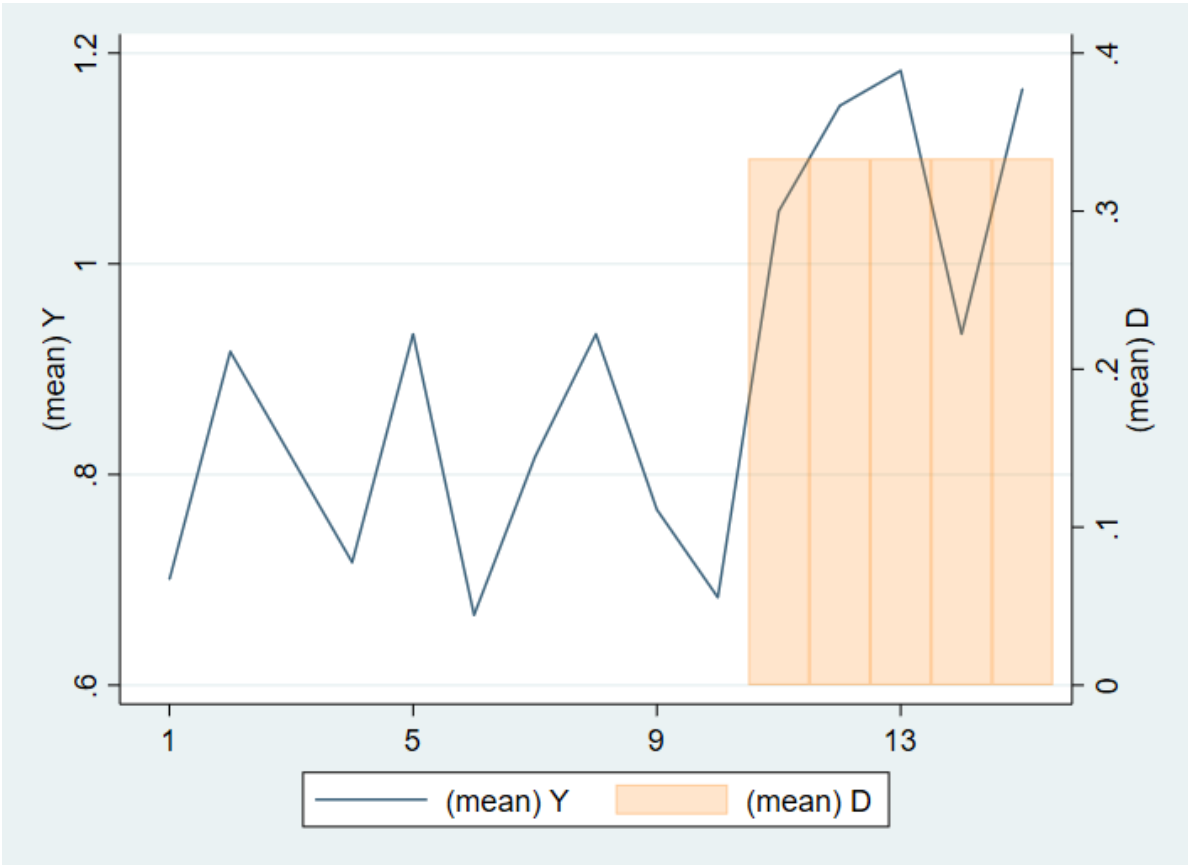



```
use capacity.dta, clear
panelview lnpop demo, i(country) t(year) xlabdist(10) allunitsplot
```



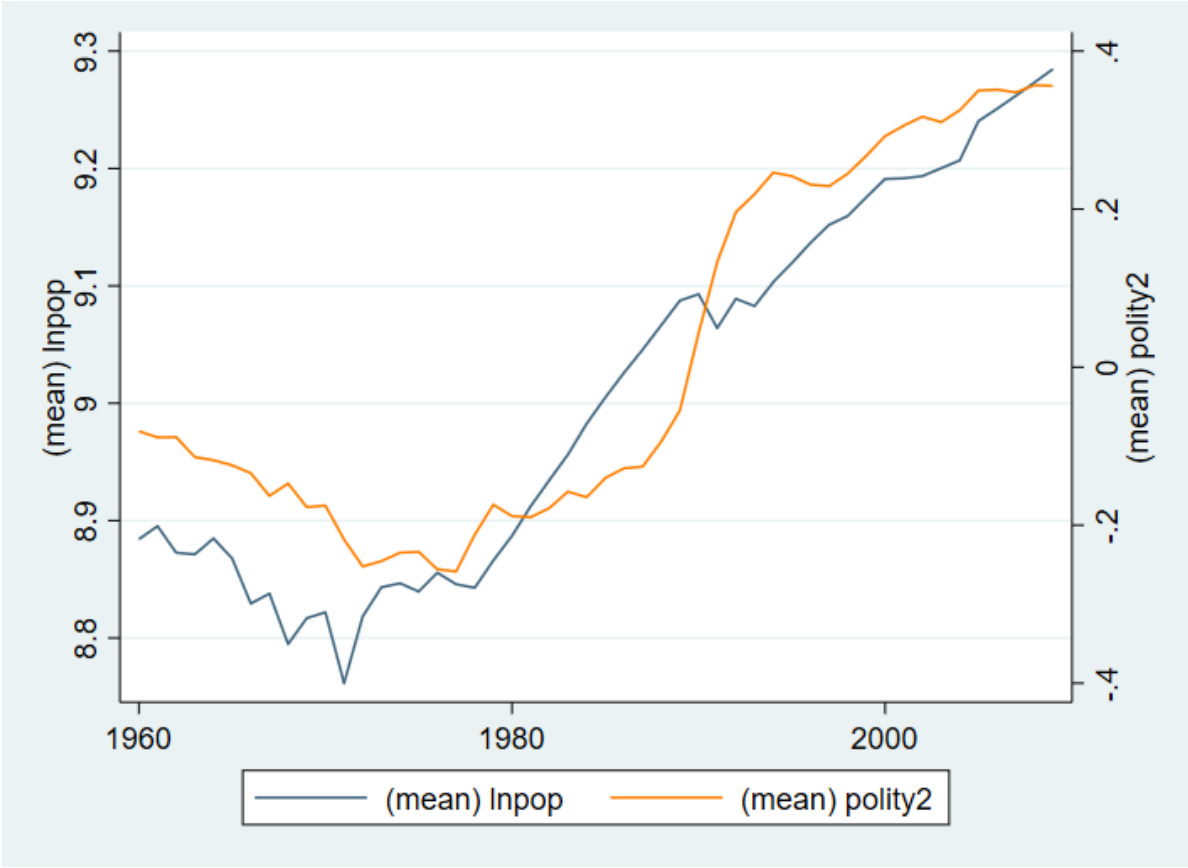
With discrete outcome and treatment:

```
/***** 2. Y: Discrete; D: dummy *****/
use simdata.dta, replace
panelview Y D,i(id) t(time) discreteoutcome xlabdist(4) allunitsplot
```



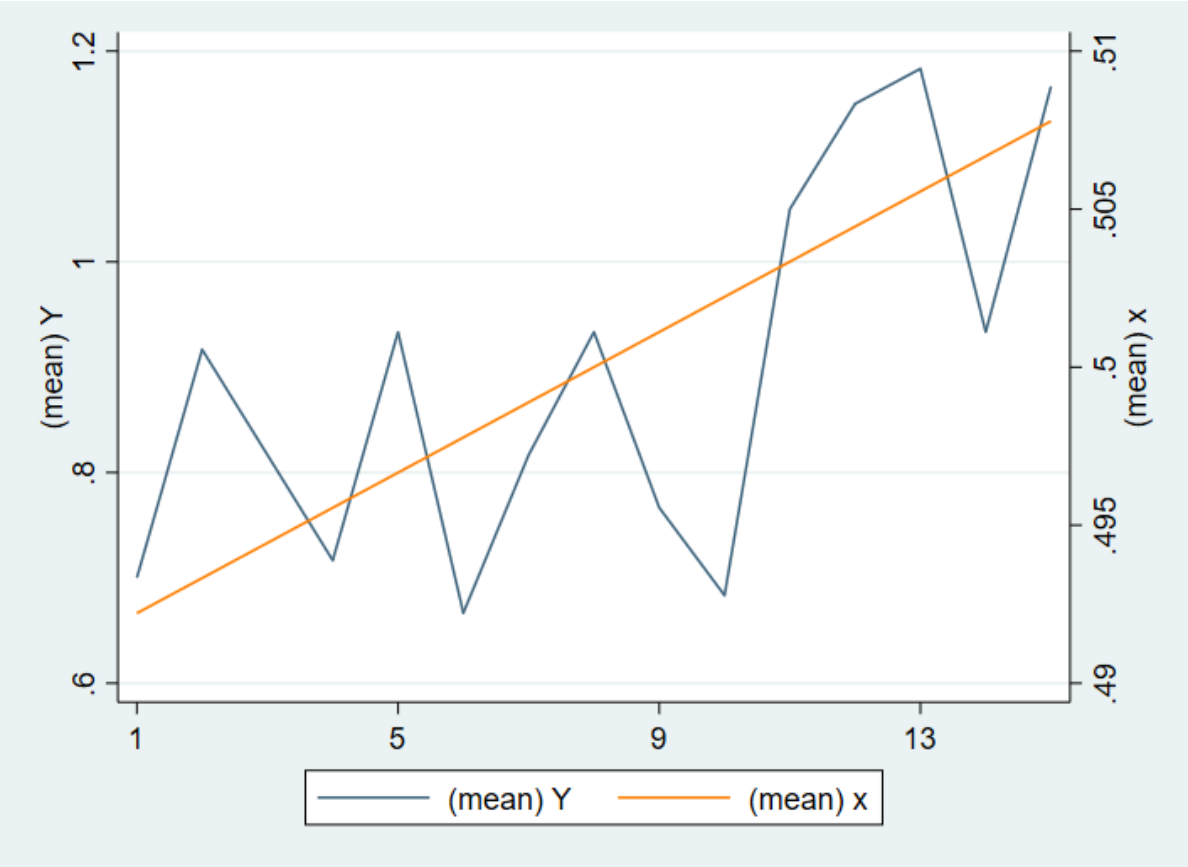
With continuous outcome and treatment:

```
/***** 3. Y: continuous; D: continuous *****/
use capacity.dta, clear
panelview lnpop polity2, i(country) t(year) continuoustreat prepost(off)
xlabdist(20) allunitsplot
```



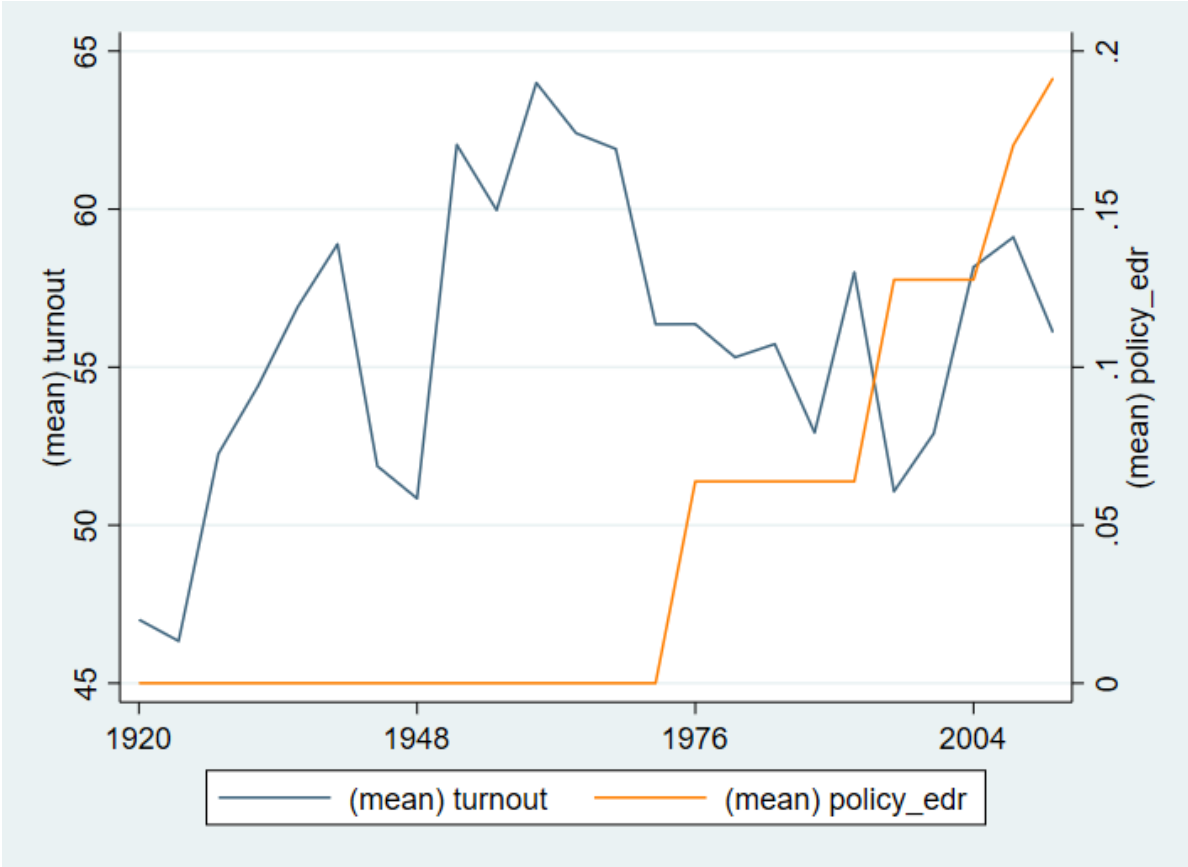
With discrete outcome and continuous treatment:

```
/***** 4. Y: Discrete; D: continuous *****/
use simdata.dta, replace
range x 0 1
panelview Y x, i(id) t(time) prepost(off) continuoustreat discreteoutcome
xlabdist(4) allunitsplot
```



Similarly, to visualize the zero level with discrete treatment, we add `linediscretreat` to plot treatment lines instead of bars:

```
/***** Line the discete treatment *****/
* Y: continuous; D: dummy
use turnout.dta, clear
panelview turnout policy_edr policy_mail_in policy_motor, i(abb) t(year)
xlabdist(7) allunitsplot linediscretreat
```




```
*Y: Discrete; D: dummy
use simdata.dta, replace
panelview Y D,i(id) t(time) discreteoutcome xlabdist(4) allunitsplot
linediscretreat
```