pcdid – Principal components difference-in-differences

Description

pcdid implements factor-augmented difference-in-differences (DID) estimation. It is useful in situations where the user suspects that trends may be unparallel and/or stochastic among control and treated units. The data structure is similar to that in a DID setup. The estimation method is regression-based and can be considered as an extension of conventional DID regressions.

pcdid also implements a parallel trend alpha test (based on an interactive effects structure) and a recursive procedure that determines the number of factors automatically.

For further details, please see Chan and Kwok (2016, 2020) who developed the **pcdid** approach and the alpha test.

Quick start

Consider the following list of variables in a long-form panel data set:

- `id`: unit identifier
- `time`: time variable
- `y`: dependent variable
- `treated`: `=1` for treated units and `treated=0` for control (never-treated) units
- `treated_post`: `=1` for all observations from treated units after policy intervention and `=0` otherwise
- `x1, x2, …` : other covariates

You must declare your data as panel data before using `pcdid`:

```
xtset id time
```

PCDID model with the number of factors determined automatically:

```
pcdid y treated treated_post x1 x2
```

Create `yhat` containing the prediction from the previous `pcdid` command:

```
pdd yhat
```

Create `yhat0` containing the counterfactual outcomes:

```
replace treated_post=0
pdd yhat0
```

Syntax

pcdid *depvar treatvar* [*didvars indepvars*] [*if*] [, *options*]

depvar	dependent variable
treatvar	control/treated unit indicator variable (=0,1)
didvars	treatment variable(s) (discrete or continuous)
indepvar	other covariate(s)

Options	***Description***
alpha	perform the parallel trend alpha test. (Note: irrelevant if there is only one treated unit)
fproxy(#)	set number of factors used. If this option is not specified, the number of factors will be automatically determined by the recursive factor number test
stationary	advanced option: assume all factors are stationary in the recursive factor number test. (Note: irrelevant if fproxy(#) is specified)
kmax(#)	advanced option: set maximum number of factors in the recursive factor number test; default is 10. (Note: irrelevant if fproxy(#) is specified)
treatlist(string)	restrict the treated unit(s) to the one(s) specified in the string expression
nwlag(#)	set maximum lag order of autocorrelation in computing Newey-West standard errors; default is int(T^0.25). (Note: irrelevant if there is more than one treated unit)
pdall	compute coefficients needed for predicting control unit outcomes. If this option is not specified, the postestimation pdd command will set all predicted control unit outcomes to zero

The postestimation command for generating predictions is

pdd *newvar*

(Note: you must use pdd instead of predict, which is invalid in this setting.)

Remarks and examples

pcdid first uses a data-driven method (based on principal component analysis) on the control panel to compute factor proxies, which capture the unobserved trends. Then, among

treated unit(s), it runs regression(s) using the factor proxies as extra covariates. Analogous to a control function approach, these extra covariates capture the endogeneity arising from potentially unparallel trends.

pcdid also allows for inclusion of other observed time-varying covariates. (Time-invariant covariates are subsumed by fixed effects.) **pcdid** is robust to the specification of trends, e.g., it encompasses nonstationary trends.

When there are multiple treated units in the data, **pcdid** computes the mean-group (PCDID-MG) estimator for the treated units. This estimator targets the ATET casual parameter. Standard errors are obtained from a nonparametric mean-group variance formula.

When there is only one treated unit in the data, **pcdid** computes a basic (PCDID-basic) estimator for that treated unit. This estimator targets the ITET casual parameter. Standard errors are obtained from the Newey-West variance formula.

For more details about the properties of these estimators, their target causal parameters, and the parallel trend alpha test, see Chan and Kwok (2016, 2020) in the reference list. The paper also contains a recursive procedure to factor extraction, adapted from Ahn and Horenstein (2013)'s growth-ratio (GR) test.

The latest version of the materials, paper, and additional remarks/examples can be found at https://sites.google.com/site/marcchanecon/.

Additional example 1:

The following command implements pcdid estimation with 3 factor proxies and performs the parallel trend alpha test.

```
pcdid y treated treated_post x1 x2, alpha fp(3)
```

Additional example 2:

We can have more than one treatment variable. For example, suppose `treated_post2 =1` for all observations from treated units that are at least 3 periods after policy intervention, `=0` otherwise.

The following command captures a step function of treatment effects over time.

```
pcdid y treated treated_post treated_post2 x1 x2
```

Additional example 3:

Suppose id==1 is a treated unit. The following command implements pcdid-basic estimation on this treated unit, using a NW lag order of 3:

```
pcdid y treated treated_post x1 x2, tr(id==1) nwlag(3)
```

Additional example 4:

Generate predicted outcomes and residuals for all control and treated units:

```
pcdid y treated treated_post x1 x2, pdall
pdd yhat
gen resid = y - yhat
```

Generate counterfactual outcomes assuming no treatment:

```
replace treated_post = 0
pdd yhat0
```

Generate counterfactual outcomes assuming no treatment and x1=1:

```
replace treated_post = 0
replace x1 = 1
pdd yhat01
```

Then plot the outcomes for id==1:

```
line y yhat yhat0 yhat01 time if id==1
```

Stored results

Pcdid saves the factor proxies in a separate data file called `fproxy.dta (variables: fproxy1, fproxy2, ...)`. Make sure that there are no naming conflicts with your other data sets and variables.

Pcdid stores the following in `e()`:

Scalars

e(Ne)	number of treated units
e(Nc)	number of control units
e(T)	number of time periods
e(nobs)	number of observations
e(factnum)	number of factors used
e(factnum0)	number of I(0) factors determined by the recursive procedure
e(factnum1)	number of I(1) factors determined by the recursive procedure
e(alphastat)	alpha statistic
e(alphastatse)	alpha statistic standard error
e(alphastatz)	alpha statistic z-score
e(alphastatp)	alpha statistic p-value
e(kmax)	maximum number of factors set by user
e(nwlag)	maximum lag order for Newey-West standard error

e(treatlistnum) =0 if e(treatlist) is empty, =1 otherwise

Macros

e(cmd)	"pcdid"
e(id)	id variable in xtset
e(time)	time variable in xtset
e(depvar)	dependent variable
e(treatvar)	control/treated unit indicator variable
e(indeps)	treatment variable(s) and other covariates
e(treatlist)	string expression specified by the treatlist option

Matrices

e(b)	coefficient
e(V)	variance
e(mata)	unit-specific coefficients for the alpha test
e(matb)	unit-specific coefficients (treated units) for the pcdid estimator
e(matc)	unit-specific coefficients (control units; for predictions only)
e(bmgc)	number of treated units used in computing each MG coefficient

Reference

Ahn, S. and A. Horenstein (2013): Eigenvalue Ratio Test for the Number of Factors. *Econometrica*, 81, 1203-1227.

Chan, M.K. and S. Kwok (2020): The PCDID Approach: Difference-in-Differences when Trends are Potentially Unparallel and Stochastic.
(Previously circulated as: Policy Evaluation with Interactive Fixed Effects. University of Sydney working paper, 2016.)

Compatibility and known issues

This is version 1.0 of the program (date: Feb 09, 2021).

The following are required to run the program:

- Stata 14.0 or higher by default, although users may modify the version line in the ado file such that it can be run under an earlier version of Stata
- The data must be recognized as panel data by Stata with the command `xtset`

In the PCDID-basic estimator (one treated unit), the command uses the `newey` package to estimate the variance matrix. It may be sensitive to issues inherent to the `newey` command.

The latest version of the materials, paper, and additional remarks/examples can be found at https://sites.google.com/site/marcchanecon/.