didunit: a unit-level multi-period differences-in-differences estimator in R

Ransi Clark[a], Jonathan N. Katz[b], R. Michael Alvarez[c]

[a] *Caltech, CA 91125, ransi@caltech.edu*
[b] *Caltech, CA 91125, jkatz@caltech.edu*
[c] *Caltech, CA 91125, rma@hss.caltech.edu*

Abstract

This R package estimates multi-period differences-in-differences at the level of the treated unit. This allows more flexible aggregation over estimators whose most granular differences-in-differences estimate is at the treated time and is useful in applications where there is considerable heterogeneity within the treated time group or when units treated at the same time receive somewhat different treatments. For example, when units are treated with different doses, they can be aggregated on the basis of dose to derive a dose-response function. Regional heterogeneity, as illustrated by a cross-country study on democratization, is another example. The software's calls has the same syntax as the did package and agrees with those estimates when panels are balanced and covariates are not relevant.

Keywords: differences-in-differences, dynamic effects, dose-response, staggered adoption

Metadata

See Table 1.

1. Motivation and significance

Differences-in-differences (DiD) estimates in a panel data setting with treatment occurring at various times have been commonly derived using two-way fixed effects. This regression recovers DiD estimates by imposing fixed effects for the unit and time period. They have been shown to be biased in applications where the treatment effect is heterogeneous in treated time [1, 2]. Since it is impossible to know a priori if treatment effects are heterogeneous, alternative estimation routines are needed to implement the design.

Nr.	Code metadata description	
C1	Current code version	1.1.0
C2	Permanent link to code/repository used for this code version	`https://github.com/ransiw/didunit`
C3	Permanent link to Reproducible Capsule	
C4	Legal Code License	GPLv2
C5	Code versioning system used	git
C6	Software code languages, tools, and services used	R
C7	Compilation requirements, operating environments & dependencies	`fastglm`
C8	If available Link to developer documentation/manual	`ransiw.github.io/didunit/`
C9	Support email for questions	ransiclark@gmail.com

Table 1: Code metadata (mandatory)

The most widely used of such routines is that of the `did` package built in `R` [3]. The `did` algorithm calculates differences-in-differences for each treated time group. However, units within a treated time group[1] can be heterogeneous. For one, although all units are treated at the same time, they can be treated with different doses. Moreover, units differ in their attributes and can experience the same treatment differently.

For example, in an application where the outcome is per-capita GDP and treatment is democratization, countries in Europe, Latin America, and Africa may democratize the same year. If the most granular estimates are calculated at the treated time, the user cannot isolate regional heterogeneity in treatment effects, unless all units within the treatment group are of a single region. Our extension allows for such aggregation by calculating a more granular estimate, i.e. at the level of the treated unit.

The algorithm operate in two major steps. First, the 2×2 differences-in-differences are estimated separately for every treated unit at every time period. Then, the first-step estimates are aggregated according to user specification. A simple aggregation averages all post-treatment effects to an overall effect. The first-step post-treatment effects can also be aggregated within the treated time group, dynamic time, or calendar time.

Implementing the first-step estimates for each treated unit instead of the

[1]Each row in the panel dataset is a unit-time cell.

group has several advantages. For one, it allows aggregation across a wider spectrum of attributes, including treatment attributes such as dosage, which generalizes the `did` algorithm to non-binary treatment settings. Secondly, the more disaggregated algorithm can help diagnose the source of outliers and overlap violations within the treated time group. In balanced panels, the aggregates produced from either the `did` or `didunit` algorithm are equal.[2]

An important difference between the two algorithms is in how they handle sample attrition. The `didunit` algorithm handles unbalanced panels by ensuring compositional balance between the pre-treatment and post-treatment. This ensures that units that had exited the sample or their treatment status since being observed pre-treatment does not contribute to the estimates. This is in contrast to the `did`'s repeated cross section (RCS) implementation for unbalanced panels, which assumes that were the units observed only post-treatment were observed pre-treatment, their pre-treatment outcomes would be similar to the pre-treatment outcomes of the units that were observed[4].

While disaggregation of the treated group to the unit-level offers advantages with flexible aggregation, the individual treatment effects require a stronger unit-level parallel trend assumption for causal identification, compared to the average of the treated time group-level parallel trend assumption. That is, a treated-time group causal estimate assumes that the average of the treated units' counterfactual trend is the same as average of the control units' trend. But a unit-level causal estimate assumes that the single treated unit's trend is the same as the average of the control units' trend. However, under aggregation, the parallel trend assumption is weakened. As noted previously, the aggregates of `did` and `didunit` agree in several cases.

Several other features of the `did` package are retained in `didunit`, including weighting schemes for pre-treatment covariates, all aggregation schemes, choices over control groups (between "never-treated" or "not-yet-treated"), restrictions to the lag length for dynamic aggregation, and additional sample weights. Extensions are in unit-level aggregations, custom aggregations, and further restriction to control group based on a custom variable.

The inferential strategy of `didunit` is that of conformal inference, which is recommended for use in synthetic controls that also produces unit-level counterfactuals [5]. Conformal inference techniques provide interval estimates for a new observation drawn from the control set. By taking the treated unit to be the new observation, a prediction interval is produced for the unit's

[2]This equality is provided that weighting is not used. Even when panels are balanced the inverse propensity weights differ for units in the same treated-time group. Overlap violations are also more numerous at the unit-level. This, however, has diagnostic advantages.

counterfactual under no treatment, and then subtracted against the treated unit's actual outcome to arrive at an interval for the treatment effect [6].

The influence function approach of `did` requires a sizable treated time group to achieve coverage (heuristically, ten or more). In cross-country applications, the treated time group sizes can be as small as one. The conformal strategy alleviates under-coverage in small groups. However, it requires an additional assumption that the treated unit's counterfactual residual and control units' residuals are exchangeable.[3] If covariates are available, the exchangeability assumption can be relaxed to weighted exchangeability. The algorithms for producing these intervals and their aggregates are described in detail in the Appendix.

The `didunit` package is most useful when there is a high level of heterogeneity between units, such as in cross-country studies, or cross-state studies. However, while it accommodates unbalanced panels, it cannot be used in settings where the unit composition changes every time period such as in the Current Population Survey (CPS) or the American National Election Study (ANES). Software best suited for such applications is recommended below.

2. Software description

We begin with the setup of the algorithm, and then demonstrate its core functionalities with examples. The implementation requires a panel dataset with each row indexed by a unit identifier and a time identifier.

2.1. Description of algorithm

Let $Y_{i,t}$ denote the outcome at time t for unit i. Index control units from $1, \cdots, N$ and treated units from $N+1, \cdots, N+N_T$. There are N_T treated units. The time index t is in $1, \ldots, T$. Each unit has an associated vector G_i which records its time of treatment, and an associated vector of attributes $X_{i,t}$. If covariates are not relevant, let $X_{i,t} = 1$. Denote the control group at any time as C_t.

For notational simplicity, we assume a balanced panel where the data $\{(Y_{i,t}, X_{i,t})\}$ is observed for each unit i at every time t. Assume the control group is the never treated group, and that once treated always treated. Also assume that the immediate pre-treatment time for a treated unit is $G_i - 1$. All these assumptions can be relaxed.

[3]Consider a data generating model where $Y_i = \alpha_i + \tau D_i + \epsilon_i$. For our intervals to guarantee coverage, we must assume that the variance of the residual ϵ_i is the same for all i regardless of treatment status D_i, $\sigma_\epsilon(D_i = 1) = \sigma_\epsilon(D_i = 0)$. Violation of this will result in under- or over-coverage.

Let $\Delta Y_{i,t}$ of a treatment unit be the outcome's difference from the pre-treatment outcome Y_{i,G_i-1}. Also, let w_i be the weights of the form $e(X_{i,G_i-1})/1-e(X_{i,G_i-1})$ where $e(X_{i,G_i-1})$ is the predicted probability of treatment for a unit i based on its pre-treatment information.

2.1.1. First step

The first-step $DID^{2\times2}$ estimates for the treated unit j from a doubly robust model assumption (`dr`) is:

$$DID^{2\times2}_{j,G_j,t} = (\Delta Y_{j,t} - \hat{f}_0(X_{j,G_j-1})) - \sum_{i \in C_t} w_i(\Delta Y_{i,G_j+(t-G_j)} - \hat{f}_0(X_{i,G_j-1})) \quad (1)$$

Here $\hat{f}_0$ is a shorthand for the outcome regression of $\Delta Y_{i,t}$ on X_{i,G_j-1} in $i \in C_t$. If only an outcome regression model is assumed (`reg`), the algorithm sets all $w_i = 0$. If only the inverse propensity score model is assumed (`ipw`) set $\hat{f}_0 = 0$ in formula (1) above.

We use the Jackknife+/CV+ conformal inference method to produce confidence intervals for the $DiD^{2\times2}_{j,G_i,t}$ estimates (See [7] for theoretical details). The complete algorithm for these weights are in the Appendix, as Algorithm 1. Here we describe it in brief. Having fixed a treated unit j, for every $i \in C_t$, reestimate a new $\hat{f}_0^{(-i)}$ without that particular i, record its new counterfactual prediction as $\hat{f}_0^{(-i)}$, and a residual term $r_i = |\Delta Y_{i,t} - \hat{f}_0^{(-i)}|$. Produce a $1-\alpha$ interval estimate for the counterfactual prediction by taking the α quantile of the $\hat{f}_0^{(-i)} - |r_i|$, and the $1-\alpha$ quantile of $\hat{f}_0^{(-i)} + |r_i|$. To get the interval estimate for $DiD^{2\times2}_{j,G_j,t}$, subtract the prediction interval from the realized $\Delta Y_{j,t}$. The algorithm can be sped up by increasing the size of the holdout set, in which case it would become a CV+.

2.1.2. Second step

One type of second-step aggregation is that of all post-treatment effects of a treated unit i (`unit`).

$$ATT(j) = \frac{1}{T - G_j + 1} \sum_{t=G_j}^{T} DID^{2\times2}_{j,G_j,t} \quad (2)$$

Another option, `dynamic`, aggregates to the lag $e := t - G_j$.

$$ATT_e = \frac{1}{N_T} \sum_{j=N+1}^{N+N_T} DID^{2\times2}_{j,G_j,G_j+e} \quad (3)$$

To aggregate to calendar time t (`calendar`), assume that$\mathcal{T}_t = \{j : G_j >= t\}$, and then implement the following.

$$ATT_t = \frac{1}{|\mathcal{T}_t|} \sum_{j \in \mathcal{T}_t} DID^{2\times 2}_{j,G_j,t} \tag{4}$$

Another type of aggregation is the further aggregation of objects calculated by Eq. (2) into the treatment time group (`group`). Supposing that $\mathcal{G}_g$ is the set of treated units for which $G_j = g$, and $|\mathcal{G}_g|$ is that set's cardinality, calculate the group g's post-treatment aggregate with Eq. (5) below.

$$ATT_g = \frac{1}{|\mathcal{G}_g|} \sum_{j \in \mathcal{G}_g} ATT(j) \tag{5}$$

For overall aggregation (`simple`), take the set $\mathcal{G}_g$ to be all treated units. If there are custom variables, replace the set $\mathcal{G}_g$ with all treated units that have the same custom value. This could be those that were treated with a particular dose.

The $(1-\alpha)$-interval for unit-level aggregates in Eq. (2) is made by aggregating over the counterfactual predictions $\hat{f}_0^{(-i)}$ and residuals r_i and forming the quantiles of those aggregates. Algorithm 2 in the Appendix details this procedure.

The interval estimates for aggregates in Eq.s (3)-(5) can be produced either by taking a Minkowski aggregate of the intervals produced for $ATT(j)$ in Eq. (2) or by estimating standard errors for each $ATT(j)$, out of $\hat{f}_0^{(-i)}$ and residuals r_i and then aggregating these standard errors. The latter method assumes independence across treated units j and asymptotic normality of aggregates and therefore lacks the same finite sample guarantees that Algorithm 1 gives. Minkowski aggregates do not assume independence, are distribution-free, and have better small sample coverage, but are too conservative, under independence. Algorithm 3 details both procedures.

As remarked above, conformal intervals require an additional assumption of exchangeability between the treated unit's counterfactual residuals and the control unit's residuals. If this assumption is true, these intervals provide better coverage when there are fewer than 10 treated units in each treated time group. For example, in our cross-country democratization example, 32 treated-time groups have fewer than 5 democratizing countries, and only 1 group has 10 or more democratizing countries. When treated groups are small `did`'s interval estimates greatly undercover.

Table 2 demonstrates this with the simulation results for data from `did`'s internal simulator for different treatment time group sizes and software. The columns record coverage for two intervals from `did`, and two intervals from

didunit (the independence assumed conformal interval and the Minkowski interval).

Group Size	did	did (unif)	didunit (indep)	didunit (Minkow)
1	5.9	5.9	95.5	95.5
2	59.2	59.5	94.1	99.6
5	86.6	86.7	95.2	100
10	92.7	92.7	94.9	100
20	92.6	92.6	94.2	100
30	93.6	93.6	94.2	100
50	94.5	94.7	94.4	100

Table 2: Coverage of 95pc CIs (%) by size of group across software

The Appendix reports additional tables for when the exchangeability assumption is violated.

2.2. Software architecture

The first-step estimates as in (1) are derived using the `att_it()` function. The second-step of aggregation as in (2)-(4) is done with the `aggite()` function. The `aggite2()` function can aggregate first-step estimates into within-group dynamics.

The related functions for `att_it()` and `aggite()` in the `did` package are `att_gt()` and `aggte()`. There is no comparable function for `aggite2()`. In what follows, unless noted otherwise, the syntax and default options preserve those of `did`, so that only a change in the function name is needed to compare the results for the overlapping aggregation types `group`, `dynamic`, `calendar`, and `simple`.

To make comparisons between the two packages easier, the `mpdta` dataset in the original `did` package is also included in the new package. This dataset, `mpdta`, is a subset of counties in the United States along with their log teen employment rate (`lemp`), the year the minimum wage was raised above the federal minimum wage (`first.treat`), and the log of the population (`lpop`). The county is `countyreal`, and the time period is `year`.

All code examples and relevant output are reported in a separate supplementary file.

2.2.1. First-step

An example of a first-step implementation for unconditional differences-in-differences is:

```
library(didunit)
mpdta <- didunit::mpdta
fsobj <- att_it(yname = "lemp",
                gname = "first.treat",
                idname = "countyreal",
                tname = "year",
                data = mpdta,
                panel = TRUE,
                control_group = "nevertreated")
```

By default, `panel` is set to `TRUE`. If the data is unbalanced, the `TRUE` option forces the data to a balanced panel by removing units that are not observed in all time periods. If this is not desired, the `FALSE` option should be specified.

The default `control_group` is `"nevertreated"`. For a dynamically updating control group, use the `"notyettreated"` option.

The following code stubs pertain to the use of covariates for weighting:

```
fsobjx <- att_it(yname = "lemp",
                gname = "first.treat",
                idname = "countyreal",
                tname = "year",
                data = mpdta,
                panel = TRUE,
                control_group = "nevertreated",
                xformla = ~lpop,
                est_method = "ipw",
                overlap = "trim")
```

The `xformla` inputs the pre-treatment covariates that will be used by the weighting schemes, which are specified using the `est_method`. This option allows for three types of weighting of which doubly-robust (`"dr"`) weighting is the default. Other options for weighting are outcome regression (`"reg"`) and inverse propensity score weighting (`"ipw"`). Since both `"ipw"` and `"dr"` require calculation of propensity scores, this maximum of the propensity score is reported in the `ipwqual` field in the `att_it()` output.

Weighting units using either `"ipw"` or `"dr"` assume overlap. To illustrate the failure of overlap, imagine a single covariate such as county population `lpop`. If all control units have a population less than that of the treated unit, overlap is violated. With more covariates, overlap failures are less intuitive, but more likely to occur. Such failures can cause numerical problems such as non-convergence of the optimization algorithm and produce inflated standard errors.

The input `overlap` specifies how overlap failure is dealt with when using either `"dr"` or `"ipw"`. The default option is to remove (`"trim"`) first-step

estimates that do not have sufficient overlap[4]. The use of "trim" drops the estimate before aggregation, but the estimated value is recorded in the field attcalc in att_it()'s output. Users who do not wish to remove estimates with overlap failures can use the option "retain". The related att_gt() of the did package does not accept the overlap input. Its default is to "trim" if panel is set to TRUE.

Users who need to view the results in tabular form can apply the attit_table() function to the output of att_it(). This function can also be applied to objects produced by did's att_gt().

Interval estimates are produced by the Jackknife+ algorithm, where the holdout set is one unit. To speed up the algorithm the size of the holdout set can be increased using the conformal_split parameter. If weighted conformal intervals are preferred, set weighted_conformal to TRUE.

2.2.2. Second-step

The function aggite() aggregates these first-step post-treatment effects within "unit", "group", "calendar" time, or a custom variable (more details below). Another useful aggregation is one on "dynamic" time, which can be used for pre-treatment placebo testing.

Following the previous example, a treated time group aggregation is:

```
ssgroup <- aggite(fsobj, type="group", na.rm = TRUE)
```

The default for na.rm input is FALSE. If the panel is unbalanced, this default option will error. The error informs the user that some first-step estimates could not be calculated. In a balanced panel, this could happen if overlap failures are trimmed. Setting na.rm=TRUE overrides this error message.

To see these sub-aggregation results in tabular form, feed the output of aggite() through the aggite_table() function. This function also works for output from aggte() in the original package.

Overall results and confidence intervals can be obtained by the call;

```
ssgroup$overall.att
ssgroup$overall.lci
ssgroup$overall.uci
```

For aggregates the default intervals assume independence. Setting indep to FALSE recovers the Minkowski aggregate of the intervals.

2.3. Additional functionalities of ***didunit***

The first extension to did is aggregation on a custom variable. This variable must be specified in the att_it() call under customnames. Multiple variables

[4]Specifically, if the maximum propensity score from a logit model exceeds 0.999.

can be specified. We illustrate this using a built-in simulator.

```
simdata <- sim_data(dosage = rep(c(1,2),each=5))
```

The above code stub produces a dataset with 30 units, 5 of which are treated at time period 10, another 5 treated at time period 15, and the rest never treated. The input `dosage` in the simulator varies the dose received by the treated units. Units treated in the same group can receive one or two doses.

To implement the first step, run the following;

```
attobject <- att_it(yname = "y",
                    gname = "treatg",
                    idname = "unit",
                    tname = "time",
                    data = simdata,
                    customnames = "dosage")
```

In a second step, a dose-response function can be recovered with;

```
agtobject = aggite(attobject,type="dosage")
```

Any variable can be used for aggregation as long as the variable is specified in the initial `att_it` call under `customnames`. The same variables that are used within the `xformla` can also be specified there. However, custom aggregation variables cannot vary over time. Relatedly, a user can specify a `cohort`, which keeps comparisons within a specified sub-group, such as a regions, or an age-group.

The second functionality is aggregation across more than one dimension. The user can aggregate to "group-dynamic" or "custom-dynamic". Continuing with the simulated example:

```
agtobject2 = aggite2(attobject,
                     type="dosage",
                     type2="dynamic")
```

This type of aggregation is useful if dynamic effects markedly differ based on dosage, or another custom variable of interest.

3. Illustrative example with countries as units

The differences between the two packages are more apparent with small sample sizes, such as in cross-country studies. We illustrate these differences with application to data in [8], where treatment is democratization, and the outcome of interest is gross domestic product per capita. When treated units are countries, there is considerable heterogeneity among the units. Countries

differ in population size, territorial extent, and regions. So, heterogeneous effects within the treatment time group are of interest.

Countries can also suddenly enter and leave the sample, as territorial extents change due to conflict, treaties, or independence from colonial rule. So, it is necessary to ensure that the recovered effects are not an artifact of compositional changes between the baseline and the post-treatment period. Moreover, treatment time groups based on the year of democratization are generally very small (11 groups have only one democratizing country) and inferential methods such as in `did` can undercover for small groups, as our simulation exercise demonstrates.

The example dataset for the exercise is included in the package as `demgdp`. It is derived from the replication material for [8], which analyzed the effect of democratization on growth in a panel of countries from 1960 to 2010.

The data is an annual cross-country panel for gross domestic product per capita (`gdppercapitaconstant2000us`). The variable `YearFirstDemocracy` is the first year that a country is observed as a democracy since 1960. The algorithm eliminates countries that democratized before 1960 since their pre-democratic outcome ($Y_{i,g-1}$ in Eq (1)) precedes the sample's start in 1960. We pre-process the data as follows to accommodate the non-reversal of treatment assumption in the algorithm:

```
demgdp <- didunit::demgdp
demgdp[is.na(demgdp$YearFirstDemocracy),"
   YearFirstDemocracy"] <- 0
demgdp <- demgdp[demgdp$breakdown==0,]
demgdp$wbnum <- as.numeric(as.factor(demgdp$wbcode))
```

There is considerable regional heterogeneity even among countries that democratize in the same year.

```
unique(demgdp[,c("wbcode","regionnum","
   YearFirstDemocracy")])
```

Eleven countries democratize in 1993. These are Cambodia, Central African Republic, Czech Republic, Lesotho, Lithuania, Latvia, Madagascar, Mongolia, Paraguay, Russia, and the Slovak Republic. Cambodia appears in the data for the first time in 1993, which `did` includes in the 1993 group. In contrast, the `didunit` algorithm would drop Cambodia since it was not observed before 1993 and only consider the 10 remaining democracies.

The following code stub combines the first and second step estimates to get treated time group-level aggregates:

```
out_it <- att_it(yname =
"gdppercapitaconstant2000us",
```

```
                gname = "YearFirstDemocracy",
                idname = "wbnum",
                tname = "year",
                customnames = "regionnum",
                xformla = ~1,
                data = demgdp,
                panel = FALSE,
                control_group = "notyettreated")
group_effects_it <- aggite(out_it,
                           type ="group",
                           na.rm = TRUE)
```

Swapping out the `did::att_gt` for `att_it` and removing the `customnames` gives the results produced from the `did` algorithm. We compare the overall effects and the 1970 group effects to demonstrate the consequential difference.

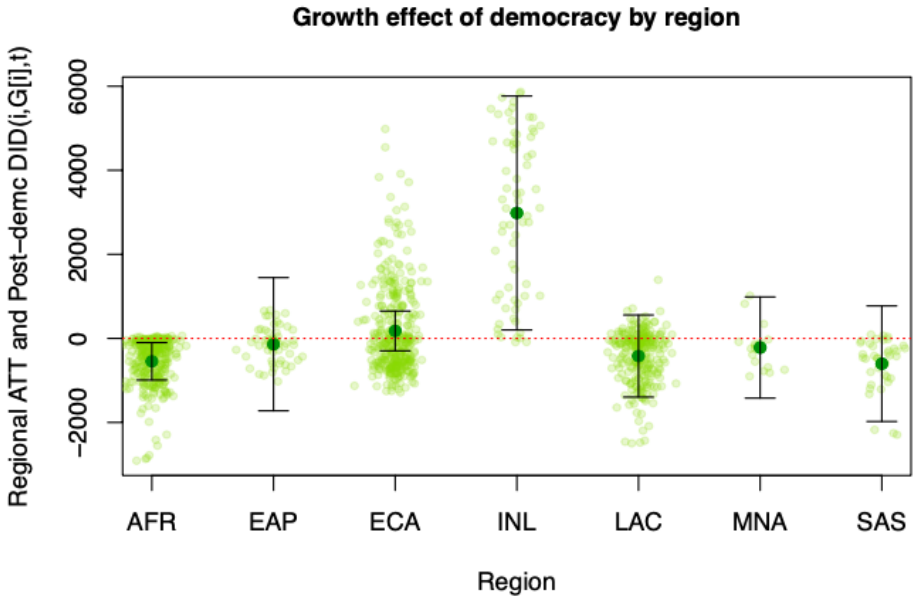
The overall effect produced by `didunit` is -46.1. The overall effect from the `did` package is 756.9. Since we do not use covariates in the call (`xformla = 1`), the differences in these estimates come from how compositional balance is imposed in `didunit`. The supplementary file demonstrates that once the panels are balanced (i.e. when `panel = TRUE`) both packages give the same group aggregates for Eq. (5) and the same overall estimate for `simple` (810.8), but drops a number of groups while balancing.

To diagnose the source of these large discrepancies, we compare the outputs of `didunit::aggite` and `did::aggte`. The largest discrepancy is in the group of countries that democratized in 1970 (-1720.5 (`didunit`) vs 8795.2 `did`). There are 5 countries in this group: West Germany, Fiji, Ghana, Ireland, and Malta. Of these, only Ghana is observed in 1969, the year that pre-treatment information is drawn from. All other countries first enter the sample in 1970. The `didunit` algorithm only considers Ghana in the 1970 group and eliminates all other countries due to lack of pre-treatment information. This applies to the control group as well, where two countries are eliminated for the lack of pre-treatment information.

Lastly, we estimate the regional treatment effect. Country-level ATTs are weighted uniformly unless a sample weighting is specified in the call for `att_it()`.

```
region_effects_it <- aggite(out_it,
                            type ="regionnum",
                            na.rm = TRUE)
```

The figure shows the results (in dark green) with their confidence intervals. The post-treatment $DiD_{i,g,t}^{2\times2}$'s (in light-green) are aligned with the relevant i's region.



The region codes are: Africa (AFR), East Asia Pacific (EAP), Eastern Europe and Central Asia (ECA), Industrialized nations (INL), Latin America (LAC), Middle East and North Africa (MNA), and South Asia (SAS).

Growth effects are different across regions. Industrialized countries ("INL") experienced faster growth on average than the control group, while Africa ("AFR") experienced slower growth than the contemporary control group. For all other regions, the effect is ambiguous.

4. Related software

Other `R` packages that implement multi-period difference-in-difference estimators are `did_multiplegt` based on [9], `etwfe` package based on [10], and `fixest::sunab` based on [11]. None of the above packages calculate effects at the unit-level. `did_multiplegt` handles non-binary treatments. The package `didimputation` based on [12] calculates effects at the unit-level but does not output these estimates. They control for covariates using linear regression. The package `PanelMatch` based on [13] performs unit-level matching. Only `did` allows for choice of control set between never-treated and not-yet-treated units. A summary is in table 3.

The package `didunit` cannot handle repeated cross sections where the unit composition changes in every time period. The `did` software handle these if selection effects can be assumed to be minor. The package `cdid` from [14]

accommodates repeated cross section data where selection effects might be more serious. The package `didunit` is also more computationally expensive because more granular estimates are produced. For very large datasets, the packages `did` and an even faster `fastdid` could more appropriate.

Estimator	Dynamics	Non-Binary	Control set choice	Matching
`didunit`	✓	✓	✓	✓
`did`	✓	×	✓	✓
`did_multiplegt`	✓	✓	×	×
`fixest::sunab`	✓	×	×	×
`didimputation`	✓	×	×	×
`PanelMatch`	✓	×	×	✓

Table 3: Comparison of existing estimators

5. Conclusions

Alternatives to panel-data regressions for varying treatment times have seen strong uptake. At the time of writing, the `did` package has been downloaded 153,000 times from the CRAN repository. This package extends its functionality by allowing for:

- sub-aggregation to treatment intensity, and other time-invariant characteristics of the treated units,
- sub-aggregation along more than one dimension,
- treated-unit-level diagnostic capability,
- greater flexibility with managing overlap violations.
- superior coverage when treated time groups are small.

A drawback of this algorithm is that it is more computationally demanding than the `did` package, particularly as the number of treated units exceed 5000. It also requires a stronger treated unit level parallel trends assumption for causal identification of the first-step estimates, although in aggregating to second-step estimates the assumption is weakened to that level of aggregation.

The package is most useful in applications where panels are highly unbalanced and there are a few but highly heterogeneous treated units. Such applications are commonplace in social sciences where comparisons are made between countries, or between administrative regions within a country.

Acknowledgements

The authors thank the three anonymous referees who provided valuable feedback.

References

[1] A. Goodman-Bacon, Difference-in-differences with variation in treatment timing, Journal of econometrics 225 (2) (2021) 254–277.

[2] J. Roth, P. H. Sant'Anna, A. Bilinski, J. Poe, What's trending in difference-in-differences? a synthesis of the recent econometrics literature, Journal of Econometrics 235 (2) (2023) 2218–2244.

[3] B. Callaway, P. H. Sant'Anna, Difference-in-differences with multiple time periods, Journal of econometrics 225 (2) (2021) 200–230.

[4] A. Abadie, Semiparametric difference-in-differences estimators, The review of economic studies 72 (1) (2005) 1–19.

[5] M. D. Cattaneo, Y. Feng, F. Palomba, R. Titiunik, Uncertainty quantification in synthetic controls with staggered treatment adoption, Review of Economics and Statistics (2025) 1–46.

[6] L. Lei, E. J. Candès, Conformal inference of counterfactuals and individual treatment effects, Journal of the Royal Statistical Society Series B: Statistical Methodology 83 (5) (2021) 911–938.

[7] R. F. Barber, E. J. Candes, A. Ramdas, R. J. Tibshirani, Predictive inference with the jackknife+, The Annals of Statistics 49 (1) (2021) 486–507.

[8] D. Acemoglu, S. Naidu, P. Restrepo, J. A. Robinson, Democracy does cause growth, Journal of political economy 127 (1) (2019) 47–100.

[9] C. De Chaisemartin, X. d'Haultfoeuille, Two-way fixed effects estimators with heterogeneous treatment effects, American economic review 110 (9) (2020) 2964–2996.

[10] J. M. Wooldridge, Two-way fixed effects, the two-way mundlak regression, and difference-in-differences estimators, Available at SSRN 3906345 (2021).

[11] L. Sun, S. Abraham, Estimating dynamic treatment effects in event studies with heterogeneous treatment effects, Journal of econometrics 225 (2) (2021) 175–199.

[12] K. Borusyak, X. Jaravel, J. Spiess, Revisiting event-study designs: robust and efficient estimation, Review of Economic Studies (2024) 3253–3285.

[13] K. Imai, I. S. Kim, E. H. Wang, Matching methods for causal inference with time-series cross-sectional data, American Journal of Political Science 67 (3) (2023) 587–605.

[14] C. Bellégo, D. Benatia, V. Dortet-Bernadet, The chained difference-in-differences, Journal of Econometrics 248 (2025) 105783.

6. Appendix

Algorithm 1: CV+/Jacknife+ Prediction Interval (two-sided, level $1-\alpha$)

---

1. **Inputs:** Data $\{(X_i, D_i, \Delta Y_{i,t})\}_{i=1}^{N+1}$, total miscoverage $\alpha \in (0,1)$, number of folds $K \geq 2$, and a black-box regressor $f(\cdot)$. This is for any arbitary t, which is suppressed in the rest of algorithm.
2. **Calculation of effect $DID^{2\times 2}_{N+1,G_{N+1},t}$ for unit $N+1$ at time t:**
   a Calculate a propensity score $\hat{e}(X_i)$ using a logistic regression of $D_{i,t}$ on X_i. If $D_{i,t}=1$, set w_i to be 1. If $D_{i,t}=0$, $w_i = \frac{\hat{e}(X_i)}{1-\hat{e}(X_i)}$.
   b Calculate an outcome regression $\hat{f}_0$ on $\Delta Y_{i,t}$ and X_i for all $D_{i,t}=0$.
   c Now calculate $DID^{2\times 2}_{N+1,G_{N+1},t}$ as in Eq.(1).[5]
3. **Calculation of conformal interval for $DID^{2\times 2}_{N+1,G_{N+1},t}$:**
   3.1 Partition the untreated units $1,\cdots,N$ into K disjoint folds $F_1,\ldots,F_K$. For each i, k_i is the fold it is left out of. Then for each fold $k=1,\ldots,K$:
      i. Repeat 2a-2c on all but fold k, obtaining $f_0^{(-k)}$.
      ii. For each $i' \in F_k$, compute the out-of-fold prediction for the treated unit's X_{N+1},

$$\hat{f}_0^{(-k_i)}(X_{N+1})$$

      and out-of-fold residual

$$r^{(k_i)} = Y_{i'} - \hat{f}_0^{(-k_i)}(X_i).$$

   3.2 Candidate bounds at treated unit's X_{N+1}, for each $i=1,\cdots,N$ at each i's holdout set k_i,

$$L_i(X_{N+1}) = \hat{f}_0^{(-k_i)}(X_{N+1}) - |r^{(k_i)}|, \qquad U_i(X_{N+1}) = \hat{f}_0^{(-k_i)}(X_{N+1}) + |r^{(k_i)}|$$

   3.3 Take interval across these bounds,

$$L(X_{N+1}) = Q_\alpha\big(\{L_i(X_{N+1})\}_{i=1}^N\big), \qquad U(X_{N+1}) = Q_{1-\alpha}\big(\{U_i(X_{N+1})\}_{i=1}^N\big).$$

   3.4 Take the $1-\alpha$ conformal interval for $DID^{2\times 2}_{N+1,G_{N+1},t}$ to be

$$(\Delta Y_{N+1} - U(X_{N+1}), \Delta Y_{N+1} - L(X_{N+1}))$$

---

[5] If the estimation method is set to inverse propensity weighting only, $\hat{f}_0 = 0$. If the estimation method is set to outcome regression only, set $w_i = 0$.

Algorithm 2: Aggregation of intervals for a single treated unit across time (two-sided, level $1-\alpha$)

1. **Inputs:** The counterfactual predictions for each post-treatment time period $\{\{\hat{f}^{(-k_i)}_{0\,[t]}(X_{_{N+1}})\}_{i=1}^{N}\}_t$, and the residuals $\{\{r^{(k_i)}_{[t]}\}_{i=1}^{N}\}_t$. The post-treatment periods are $t = G_{_{N+1}}, \cdots, T$.
2. **Calculation of estimate** $ATT(N+1)$**:** The estimate is the overtime mean for a treated unit.[6]

$$ATT(N+1) = \frac{1}{T - G_{_{N+1}} + 1} \sum_{t=G_j}^{T} DID^{2\times 2}_{N+1,G_{_{N+1}},t}$$

3. **Calculation of the conformal interval**:
   3.1 Take the across time means for $\hat{f}^{(-k_i)}_{0\,[t]}(X_{_{N+1}})$ and $r_{[t]}$

$$\tilde{f}_0(X_{_{N+1}}) = \frac{1}{T - G_{_{N+1}} + 1} \sum_{t=G_{_{N+1}}}^{T} \hat{f}^{(-k_i)}_{0\,[t]}(X_{_{N+1}})$$

$$\tilde{r}^{(k_i)}(X_{_{N+1}}) = \frac{1}{T - G_{_{N+1}} + 1} \sum_{t=G_{_{N+1}}}^{T} r^{(k_i)}_{[t]}(X_{_{N+1}})$$

   3.2 Candidate bounds at treated unit's $X_{_{N+1}}$, for each $i = 1, \cdots, N$ at each i's holdout set k_i,

$$L_i(X_{_{N+1}}) = \tilde{f}^{(-k_i)}(X_{_{N+1}}) - |\tilde{r}^{(k_i)}|, \qquad U_i(X_{_{N+1}}) = \tilde{f}^{(-k_i)}(X_{_{N+1}}) + |\tilde{r}^{(k_i)}|$$

   3.3 Take interval across these bounds,

$$L(X_{_{N+1}}) = Q_\alpha\big(\{L_i(X_{_{N+1}})\}_{i=1}^{N}\big), \qquad U(X_{_{N+1}}) = Q_{1-\alpha}\big(\{U_i(X_{_{N+1}})\}_{i=1}^{N}\big).$$

   3.4 Take the $1-\alpha$ conformal interval to be

$$(\overline{\Delta Y_{_{N+1}}} - U(X_{_{N+1}}), \overline{\Delta Y_{_{N+1}}} - L(X_{_{N+1}}))$$

   where

$$\overline{\Delta Y_{_{N+1}}} = \frac{1}{T - G_{_{N+1}} + 1} \sum_{t=G_{_{N+1}}}^{T} \Delta Y_{N+1,t}$$

[6] Weighted means can be used, but is not indicated here to keep notation simple.

Algorithm 3: Aggregation of intervals for a several treated unit across time (two-sided, level $1-\alpha$)

1. **Inputs:** The counterfactual predictions for each post-treatment time period $j = N+1, \cdots, N+N_T$ $\{\{\tilde{f}_{[t]}^{(-k_i)}(X_j)\}_{i=1}^{N}\}_j$ the residuals $\{\{\tilde{r}_{[t]}^{(k_i)}\}_{i=1}^{N}\}_j$ and total miscoverage $\alpha \in (0,1)$,
2. **Calculation of estimate ATT_g:** The estimate is the aggregate ATT for the set of units treated at time g, $\mathcal{G}_g$.

$$ATT_g = \frac{1}{|\mathcal{G}_g|} \sum_{j \in \mathcal{G}_g} ATT(j)$$

3. **Calculation of interval estimate: Minkowski mean**
   - 3.1 Calculate the $1-\alpha/|\mathcal{G}_g|$ conformal interval as in Algorithm 1 and 2.
   - 3.2 Take the Minkowski mean of these confidence intervals,

$$|\mathcal{G}_g|^{-1} \sum_j \Big(\overline{\Delta Y_j} - U(X_j), \overline{\Delta Y_j} - L(X_j) \Big)$$

4. **Calculation of interval estimate: Assuming independence**
   - 4.1 Take standard errors of each $j = N+1, \cdots, N+N_T$ associated $\tilde{f}_{[t]}^{(-k_i)}(X_j)$ and $\tilde{r}_{[t]}^{(k_i)}$ as σ_j^f and σ_j^r, respectively.
   - 4.2 Calculate a normal approximated confidence interval where $c_{\alpha/2}$ is the critical value and σ^* is the associated

$$\sigma^* = |\mathcal{G}_g|^{-1} \Big(\sum_{j \in \mathcal{G}_g} (\sigma_j^f + \sigma_j^r)^2 \Big)^{1/2}$$

   - 4.3

$$ATT_g - c_{\alpha/2} * \sigma^*, ATT_g + c_{\alpha/2} * \sigma^*$$

Simulation results for when the $\sigma_\epsilon(D_i = 1) = \sigma_\epsilon(D_i = 0)$

Imposing a data generating process where outcomes are generated based on

$$Y_{i,t} = \alpha_i + \tau D_{i,t} + \epsilon_{i,t}$$

where $D_{i,t}$ is the treatment status and τ is the treatment effect, and α_i is a unit-level mean. The perturbation $\epsilon_{i,t}$ are distributed with mean zero and standard deviation σ_ϵ.

We assume the perturbations the $\epsilon_{i,t}$ do not differ in mean or variance based on treatment status. Under this assumption, conformal intervals achieve finite sample coverage of $1-\alpha$. Table 1 of the main text, reports the coverage rates from the simulation exercise where $\sigma_\epsilon = 1$. The table is reproduced here for ease of reference.

Group Size	**did**	**did (unif)**	**didunit (indep)**	**didunit (Minkow)**
1	5.9	5.9	95.5	95.5
2	59.2	59.5	94.1	99.6
5	86.6	86.7	95.2	100
10	92.7	92.7	94.9	100
20	92.6	92.6	94.2	100
30	93.6	93.6	94.2	100
50	94.5	94.7	94.4	100

Table 4: Coverage of 95pc CIs (%) by size of group across software

The average lengths of these intervals are in the table.

Group Size	**did**	**did (unif)**	**didunit (indep)**	**didunit (Minkow)**
1	0.28	0.29	7.81	7.70
2	3.15	5.63	5.53	9.51
5	3.31	3.33	3.49	12.39
10	2.46	2.47	2.47	15.04
20	1.78	1.79	1.75	17.97
30	1.46	1.47	1.43	18.03
50	1.15	1.16	1.11	21.98

Table 5: Length of 95pc CIs by size of group across software

Simulation results for when the $\sigma_\epsilon(D_i = 1) > \sigma_\epsilon(D_i = 0)$

If the pertubations differ in variance such that $\sigma_\epsilon(D_i = 1) = 1$ but $\sigma_\epsilon(D_i = 0) = 0.5$, coverage will be poor for conformal intervals in `didunit`. The coverage tables and lengths of confidence intervals under a simulation are in the following tables.

Group Size	did	did (unif)	didunit (indep)	didunit (Minkow)
1	1.3	1.3	66.3	66.1
2	54.1	58.5	66.5	87.6
5	86.5	86.7	66.4	99.8
10	91.1	91.1	66	100
20	94	94.6	67.9	100
30	95.1	95.1	69.5	100
50	95.9	96.5	67.6	100

Table 6: Coverage of 95pc CIs (%) by size of group across software

The average lengths of these intervals are in the next table.

Group Size	did	did (unif)	didunit (indep)	didunit (Minkow)
1	0.09	0.09	2.77	2.77
2	2.41	6.19	1.96	3.16
5	2.39	2.42	1.24	3.62
10	1.73	1.73	0.88	3.92
20	1.23	1.24	0.62	4.26
30	1.01	1.01	0.51	4.48
50	0.78	0.79	0.39	4.47

Table 7: Length of 95pc CIs by size of group across software

Simulation results for when the $\sigma_\epsilon(D_i = 1) < \sigma_\epsilon(D_i = 0)$

If the pertubations differ in variance such that $\sigma_\epsilon(D_i = 1) = 0.5$ but $\sigma_\epsilon(D_i = 0) = 1$, conformal intervals in `didunit` will still achieve coverage but will be inefficient. The coverage tables and lengths of confidence intervals under a simulation are,

Group Size	did	did (unif)	didunit (indep)	didunit (Minkow)
1	8	8	100	100
2	57.8	59.5	100	100
5	87.2	87.6	100	100
10	91.5	91.3	100	100
20	94.5	94.6	100	100
30	95.1	95.1	100	100
50	96	96.2	100	100

Table 8: Coverage of 95pc CIs (%) by size of group across software

The average lengths of these intervals are,

Group Size	did	did (unif)	didunit (indep)	didunit (Minkow)
1	0.17	0.18	5.54	5.54
2	1.23	1.60	3.92	6.33
5	1.20	1.20	2.48	7.23
10	0.88	0.88	1.75	7.84
20	0.64	0.64	1.24	8.53
30	0.53	0.53	1.01	8.95
50	0.43	0.43	0.78	8.95

Table 9: Length of 95pc CIs by size of group across software