The Stata Journal (2021)
21, Number 2, pp. 411–429 DOI: 10.1177/1536867X211025801

Stacked linear regression analysis to facilitate testing of hypotheses across OLS regressions

Michael Oberfichtner
Institute for Employment Research (IAB)
Nürnberg, Germany
Michael.Oberfichtner2@iab.de

Harald Tauchmann
University of Erlangen-Nuremberg (FAU),
and RWI—Leibniz Institut für Wirtschaftsforschung,
and CINCH—Health Economics Research Center
Nürnberg, Germany
harald.tauchmann@fau.de

Abstract. In empirical work, researchers frequently test hypotheses of parallel form in several regressions, which raises concerns about multiple testing. One way to address the multiple-testing issue is to jointly test the hypotheses (for example, Pei, Pischke, and Schwandt [2019, *Journal of Business & Economic Statistics* 37: 205–216] and Lee and Lemieux [2010, *Journal of Economic Literature* 48: 281–355]). While the existing commands `suest` (Weesie, 1999, *Stata Technical Bulletin Reprints* 9: 231–248) and `mvreg` enable Stata users to follow this approach, both are limited in several dimensions. For instance, `mvreg` assumes homoskedasticity and uncorrelatedness across sampling units, and neither command is designed to be used with panel data. In this article, we introduce the new community-contributed command `stackreg`, which overcomes the aforementioned limitations and allows for some settings and features that go beyond the capabilities of the existing commands. To achieve this, `stackreg` runs an ordinary least-squares regression in which the regression equations are stacked as described, for instance, in Wooldridge (2010, *Econometric Analysis of Cross Section and Panel Data*, p. 166–173, MIT Press) and applies cluster–robust variance–covariance estimation.

Keywords: st0641, stackreg, xtstackreg, multiple testing, stacked regression, clustering, fixed effects

1 Introduction

In empirical work, researchers often test hypotheses of parallel form in several regressions. Examples are regression-based balancing tests for multiple independent variables (for example, following Lee and Lemieux [2010] and Pei, Pischke, and Schwandt [2019]) as well as studies that examine the relationships between multiple dependent variables and the same set of independent variables. Numerous researchers, including Lee and Lemieux and Pei, Pischke, and Schwandt, point out the need for joint testing across regression equations in such settings; otherwise, statistical inference may be invalid due to the multiple-comparisons problem. However, independently testing numerous parallel hypotheses without taking multiple-testing issues into account seems to still be common in applied research; see, for instance, the discussions in Anderson (2008) and List, Shaikh, and Xu (2019).

The Stata command `mvreg` and the more general `sureg` address such situations by joint estimation; however, they are subject to several limitations. For instance, they are based on a generalized least-squares approach that implies strong assumptions regarding the form of the error-correlation structure, they do not handle panel data in a satisfying way, and they have to rely on the bootstrap to obtain (cluster) robust standard errors. The originally community-contributed command `suest` (introduced by Weesie [1999]) overcomes some limitations of `mvreg`. Specifically, `suest` allows for (cluster) robust inference and, in addition, is highly flexible because it is able to combine different model types—say, ordinary least squares (OLS) and probit—and test across these models. This flexibility, however, comes at the cost of `suest` not covering all aspects of OLS regressions, the workhorse of much empirical research.

In this article, we introduce `stackreg`, a community-contributed command for joint testing of hypotheses across OLS regressions. `stackreg` offers three advantages over `suest`: first, it implements fixed-effects estimations; second, it allows for multiway clustering building upon the community-contributed command `cgmreg` (Gelbach and Miller 2009); and third, it enables cross-equation constraints. Furthermore, for convenience, `stackreg` allows users to specify factor variables as dependent variables.

In essence, `stackreg` stacks the data used in multiple OLS regressions and runs one regression on these stacked data. Stacking regressions is a conceptually simple way to do joint estimations that is laid out in, for instance, Wooldridge (2010, 166–173) and proposed for regression-based balancing tests for multiple independent variables in Lee and Lemieux (2010) and Pei, Pischke, and Schwandt (2019). However, stacking regressions is computationally more demanding than the residual-based approach used by `suest`.

Section 2 sketches the underlying econometric idea. Section 3 explains `stackreg`'s implementation in Stata and compares it with alternative Stata commands. Section 4 describes the syntax used for `stackreg`, and section 5 presents two applications of `stackreg`. Section 6 concludes.

2 Stacked regression analysis

Consider a set of G regressions, where the dependent variables $y_{1i}, \ldots, y_{Gi}$ are regressed on the same set of independent variables $\boldsymbol{x}_i$ for sampling units $i = 1, \ldots, N$:

$$\begin{aligned} y_{1i} &= \boldsymbol{x}_i \boldsymbol{\beta}_1 + \varepsilon_{1i} \\ &\vdots \\ y_{Gi} &= \boldsymbol{x}_i \boldsymbol{\beta}_G + \varepsilon_{Gi} \end{aligned} \tag{1}$$

This description accommodates a wide range of applications. Examples include balancing tests of covariates in experiments (with $\boldsymbol{x}_i$ comprising only a treatment indicator) and in regression discontinuity designs ($\boldsymbol{x}_i$ now including a function of the forcing variable) as well as sets of regressions that examine the relationship between several dependent variables and a fixed set of independent variables. In all such applications, statistical inference about $\boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_G$ should ideally account for the multitude of statistical tests. The loss of statistical power associated with a Bonferroni correction is, however, problematic, and it is thus not particularly attractive in many applications.

One alternative way to adjust statistical inference is to jointly estimate the regression equations in (1). Stacking the G regressions is a conceptually simple approach to estimate the regressions jointly (described in, for example, Wooldridge [2010, 166–173]). Using this approach, the statistical inference can account for possible cross-equation correlations of the errors ε_g without imposing additional structure. Defining $\boldsymbol{y}_i = (y_{1i}, \dots, y_{Gi})'$, $\boldsymbol{\varepsilon}_i = (\varepsilon_{1i}, \dots, \varepsilon_{Gi})'$, $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_G)'$, and

$$\boldsymbol{X}_i = \begin{bmatrix} \boldsymbol{x}_i & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{x}_i & & \boldsymbol{0} \\ \vdots & & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{x}_i \end{bmatrix}$$

the stacked regression reads

$$\boldsymbol{y}_i = \boldsymbol{X}_i \boldsymbol{\beta} + \boldsymbol{\varepsilon}_i \tag{2}$$

The stacked regression (2) mechanically gives regression coefficients that are identical to those from the separate regressions in (1). By clustering standard errors and adjusting the degrees of freedom (explained in more detail in section 3.4), the stacked regression additionally yields identical standard errors for the regression coefficients and still allows for joint testing of hypotheses across equations. The next section describes how `stackreg` implements this stacking approach and adjusts the degrees of freedom when computing standard errors.

3 Implementation in Stata

The core of the stacked regression procedure is temporarily reshaping the estimation sample from wide format to long format using `reshape long` or, optionally, `sreshape long` (Simons 2016).[1] This transforms the G original outcome variables $y_{1i}, \dots, y_{Gi}$ into a single left-hand-side variable. At the same time, a subobservation identifier—denoted by `j()` in the syntax of `reshape`—is generated that serves as a key to the original outcome variables. Subsequently, `stackreg` calls `regress` to run a regression of the stacked left-hand-side variable on a saturated set of interactions of $\boldsymbol{x}_i$ with the

1. `reshape` is known to be rather slow when used with large samples. The `sreshape` option makes `stackreg` instead call the much faster, community-contributed command `sreshape` (Simons 2016), which in turn speeds up `stackreg`.

dummy-expanded subobservation identifier. While this yields coefficient estimates that are identical to equation-by-equation estimation, it allows for estimating cross-equation coefficient covariances, which is required for testing hypotheses that involve coefficients from more than one equation. Analogously to an unbalanced panel setting, which is handled by `reshape`, this procedure accommodates settings in which the pattern of observed outcomes varies across the original sample units. This will be the case if, for instance, the different outcome variables exhibit different patterns of missing values. Even if clustering is not explicitly requested by the user, `stackreg` routinely applies cluster–robust variance–covariance estimation with clustering at the level of the original sampling units, because each contributes several (sub)observations to the stacked regression. The output from `stackreg`—displayed and stored in `e()`—is arranged to mimic the output from `mvreg`, which allows for inference after `stackreg` in the same way as after `mvreg`.

3.1 Panel data and fixed-effects estimation

`stackreg` is accompanied by the `xtstackreg` command, which implements fixed-effects panel estimation.[2] That is, rather than using the original variables in the stacked regression, the data are first temporarily within-transformed using `xtdata`. If the estimating samples are heterogeneous across the different outcomes, the within-transformation is applied equation by equation. This guarantees that `xtstackreg` yields exactly the same coefficient estimates one gets from equationwise applying `xtreg, fe` or `areg, absorb(`*panelvar*`)`. Fully equivalent to `stackreg, fe`, the key objective of `xtstackreg` is, hence, to facilitate postestimation inference that involves coefficients from more than one regression equation. As any Stata `xt` command, `xtstackreg` requires the data to be declared as panel data using `xtset`.

3.2 Higher-level and multiway clustering

Cluster–robust variance–covariance matrix estimation is a key feature of `stackreg`. The procedure can be adapted to grouped data by considering a level of clustering higher than the original sample unit. Extending this approach to multiway clustering is straightforward. If multiway clustering is requested—via a *varlist* entered in the `cluster()` option as an argument—then instead of `regress`, `stackreg` calls the community-contributed command `cgmreg` (Gelbach and Miller 2009), which implements multiway clustering as suggested in Cameron, Gelbach, and Miller (2011).

3.3 Constrained estimation

`stackreg` allows for imposing linear constraints on the estimated coefficients, including cross-equation constraints. This, in the usual fashion, requires specifying the option `constraints()`. Constrained estimation can be used, for instance, to specify sets of

2. `xtstackreg` is fully equivalent to `stackreg, fe`. In terms of the implementation as an ado-file, `xtstackreg.ado` is just a wrapper that calls `stackreg.ado` and activates the `stackreg` option `fe`.

explanatory variables that vary across the different equations. Technically, constrained estimation is implemented by calling `cnsreg` instead of `regress`.

3.4 Degrees-of-freedom adjustment

`stackreg` is designed to exactly reproduce the robust standard errors one gets from separately regressing $y_{1i}, \ldots, y_{Gi}$ on $\boldsymbol{x}_i$.[3] Because `stackreg` necessarily reproduces the coefficient estimates from separate regressions, the standard errors should also not deviate from what separate regressions yield. While this coincidence of the estimated standard errors is asymptotically guaranteed, it becomes an issue in finite samples, small ones in particular. In other words, the degrees-of-freedom correction `regress` initially applies when called by `stackreg` needs to be adjusted. In the most simple case of a cross-sectional estimating sample that is homogeneous across all regression equations, the correction factor that has to be applied to the initially estimated variance–covariance matrix is $(N-1)/\{N-(1/G)\}$. This adjustment also applies if panel data are used and one wants to reproduce the standard errors of `xtreg, fe robust`. More precisely, in this case the factor is $(TN-1)/\{TN-(1/G)\}$, with T denoting the number of panel waves.[4] If the standard errors of `areg, absorb(`*panelvar*`) robust` are to be reproduced, the adjustment factor is $\{(TN-K)/(TN-K-N+1)\}[(TN-1)/\{TN-(1/G)\}]$.

Things get more involved if the estimation samples are heterogeneous across the dependent variables[5] or if restrictions are imposed on the coefficients; see section 3.3. In these cases, different adjustment factors must be applied to the different equations. For this reason, the initially estimated variance–covariance matrix is not adjusted by a single scalar factor but is adjusted element by element. The element-specific adjustment factors are $\sqrt{c_g \times c_h}$, with c_g and c_h denoting the equation-specific adjustment factors and g and h indexing the equations $1, \ldots, G$. This approach to adjusting for degrees of freedom that are heterogeneous across equations parallels what `sureg` with option `dfk` does.[6]

3. Specifying the `stackreg` options `df()`, `cluster()`, and `fe` appropriately allows achieving a perfect match with the standard errors that various Stata commands yield—more specifically, `regress, robust`; `regress, cluster()`; `cgmreg, cluster()`; `xtreg, fe robust`; `xtreg, fe cluster()`; `areg, absorb(`*panelvar*`) robust`; and `areg, absorb(`*panelvar*`) cluster()`.
4. For simplicity, we consider a balanced panel, yet the result carries over to unbalanced panels.
5. By default, `stackreg` only considers observations for which information regarding all dependent variables is available, which renders the estimation samples homogeneous. Yet, with the `nocommon` option, `stackreg` also accommodates the heterogeneous case; see section 4.2 for more details.
6. Though `stackreg` exactly reproduces the standard errors the corresponding equation-specific regressions yield, this may not hold for the reported p-values and t statistics. This is because `stackreg` routinely applies clustering and, in accordance with other Stata commands that accommodate clustering, stores the number of clusters minus 1 in `e(df_r)`. If the corresponding equation-by-equation approach does not involve clustering, a different value is stored in `e(df_r)` and is thus used for calculating p-values and t statistics.

3.5 Comparison to existing Stata commands

`stackreg` is related to several existing Stata commands that also implement methods for statistical testing in a multiple-equation setting. Among these routines, the presumably most flexible is `suest` (Weesie 1999; StataCorp 2019a, 2578–2596). `stackreg` and `suest` share the idea of using cluster–robust variance–covariance estimation for valid inference in a multiple-equation setting. Unlike `stackreg`, however, `suest` is not confined to linear models and allows for testing joint hypotheses that involve various different linear and nonlinear models. Despite its confinement to the linear model, `stackreg` still accommodates features that go beyond `suest`. In detail, these are i) fixed-effects estimation (see section 3.1), ii) multiway clustering (see section 3.2),[7] iii) exact replication of the standard errors one gets from equation-by-equation estimation (see section 3.4),[8] and iv) imposing cross-equation restrictions (see section 3.3).[9] Finally, `stackreg` is more conveniently used, because it requires executing just one command and allows for factor variables in the list of dependent variables.

`mvreg` is another Stata routine to which `stackreg` is closely related.[10] Actually, in terms of the syntax and the output that appears on the screen, `stackreg` closely follows `mvreg`. From the perspective of econometric theory, the key difference between `mvreg` and `stackreg` is that `mvreg` implements a feasible generalized least-squares (FGLS)[11] procedure to estimate cross-equation coefficient covariances, while `stackreg` uses cluster–robust variance–covariance estimation for this purpose. FGLS hinges on correctly specifying the structure of the error variance–covariance matrix, while cluster–robust standard error estimation does not require this. For this reason, the approach `stackreg` takes is more robust and more flexible than FGLS implemented by `mvreg`. Because of its greater flexibility—unlike `mvreg`—`stackreg` can deal with grouped data (that is, clustered standard error estimation),[12] panel fixed-effects estimation, estimation samples that are not identical across outcomes, and constraint estimation.[13]

7. The community-contributed prefix command `vcemway` (Gu and Yoo 2019), which implements multiway clustering for many existing Stata commands, appears not to run with `suest`. It also appears not to accommodate multiequation estimation procedures, such as `mvreg` and `reg3`, and also does not run with `stackreg`.
8. The degrees-of-freedom correction applied by `suest` yields standard errors that differ from the standard errors that equation-by-equation estimation yields (StataCorp 2019a, 2581).
9. `stackreg` estimates all coefficients simultaneously, which allows imposing such restrictions. This is, by design, not possible with the equation-by-equation estimation approach on which `suest` is built. `suest`'s equation-by-equation estimation approach is, however, more efficient in terms of required computing time; `stackreg` is slower than `suest`—including the regressions that precede running `suest`—in particular, if the numbers of dependent variables is large.
10. See StataCorp (2019b, 556–557) for a discussion of how `mvreg` can be used for multiple comparisons.
11. Though `mvreg` yields coefficient estimates that do not deviate from what OLS—that is, `regress`—yields, it is still an FGLS estimator because the coefficient covariance matrix is estimated following an FGLS approach. This becomes obvious by thinking of `mvreg` as a special case of `sureg`, which implements FGLS estimation of a seemingly unrelated system of regression equations; see StataCorp (2019b, 558). In a setting in which no restrictions are imposed on $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_G$, the prime argument in favor of FGLS (that is, being more efficient) does not apply (Zellner 1962).
12. One-way higher-level clustering can only indirectly be implemented with `mvreg` through `bootstrap, cluster() idcluster(): mvreg`.
13. However, if `sureg`, which can be regarded as a generalized alternative to `mvreg`, is used, constraint can be imposed.

Finally, combining Stata's data management tool `stack` with `regress` can also be regarded as an alternative to `stackreg`. Weesie (1999), for instance, proposes implementing the stacked regression approach using `stack`. While this approach is relatively straightforward in basic applications, it becomes very cumbersome if the estimation procedure involves equation-specific data transformations, for example, the within-transformation to eliminate individual fixed effects. Moreover, using `stack` and afterwards `regress, cluster()` does not apply a degrees-of-freedom adjustment that takes into account the stacking of regressions.

4 The stackreg and xtstackreg commands

`stackreg` requires Stata 12 or higher. If multiway clustering is requested, `stackreg` requires the community-contributed command `cgmreg`.[14] The prefix commands `by` and `svy` are not allowed. The prefix commands `bootstrap` and `jackknife` are allowed. However, one may question whether bootstrapping `stackreg` makes much sense, because the prime benefit that the command provides is allowing for conventional—that is, no resampling or simulation based—inference in a multiple-testing framework. All weight types (`aweight`, `fweight`, `iweight`, and `pweight`) are allowed, with `pweight`s being the default weight type.

4.1 Syntax

```
stackreg depvars = indepvars [if] [in] [weight] [, fe noconstant
    constraints(numlist) nocommon cluster(clustvarlist) df(adjust|raw|areg)
    wald sreshape level(#) edittozero(#) omitted emptycells
    display_options]
```

```
xtstackreg depvars = indepvars [if] [in] [weight] [, noconstant
    constraints(numlist) nocommon cluster(clustvarlist) df(adjust|raw|areg)
    wald sreshape level(#) edittozero(#) omitted emptycells
    display_options]
```

The syntax for `xtstackreg` is exactly the same as the syntax for `stackreg`, except the `fe` option, which is automatically specified with `xtstackreg`. In other words, specifying `xtstackreg` is fully equivalent to specifying `stackreg` with option `fe`. We provide the separate `xt` command to make more salient that `stackreg` is designed to take the (possible) panel nature of the data into account.

14. Unfortunately, `cgmreg` is not available from the canonical sources (`ssc` and the *Stata Journal*) of community-contributed Stata commands. Hence, different versions of `cgmreg` may be found on the Internet that may substantially vary, for example, in terms of the available options. `stackreg` was tested with `cgmreg` version 3.0.0 (Gelbach and Miller 2009), which at the time of publication is available from the personal website of Colin Cameron at the University of California, Davis.

depvars specifies the list of outcome variables, and *indepvars* specifies the list of explanatory variables. Factor variables are allowed in both *indepvars* and *depvars*; see [U] **1.4.3 Factor variables**. Time-series operators such as `L.` and `F.` are also allowed.

4.2 Options

`fe` makes `stackreg` use within-transformed values of *indepvars* and *depvars* rather than their levels when estimating the stacked regression. That is, with the `fe` option (fixed effects), `stackreg` eliminates unobserved individual heterogeneity. `fe` requires that the data are declared as panel data by using `xtset`. `stackreg` with option `fe` is fully equivalent to `xtstackreg` (with and without option `fe`, that is, `fe` has no effect with `xtstackreg`). We provide the separate `xt` command to make more salient that `stackreg` can be used with panel data.

`noconstant` suppresses the constant terms in the stacked regression. `noconstant` drops the constant terms from all regression equations because `stackreg` considers the same set of explanatory variables for all equations.

`constraints(`*numlist*`)` requests that `stackreg` apply the linear constraints specified by *numlist*, which must comply with Stata's *numlist* syntax; see [U] **11.1.8 numlist**. The specified constraints must be defined in advance by using `constraint`; see [R] **constraint**. The syntax for referring to a coefficient when defining constraints is `[`*depvar*`]`*indepvar*. To identify coefficients, both the equation and the explanatory variable are thereby specified. Factor-variables syntax is allowed for specifying constraints, for example, `[health]1998.year = 0`. Cross-equation constraints can be defined as usual, for example, `[health]income = [happiness]income`. The option `constraints()` cannot be combined with multiway clustering. If `constraints()` is specified and `noconstant` is not specified, then `stackreg` estimates an overall constant and drops the equation-specific constant from the final equation.

`nocommon` makes `stackreg` select the estimation sample on an equation-by-equation basis. That is, observations for which information on some variables in *depvars* is missing are used, and the number of observations thus may vary across the different equations. The default (`common`) is to only consider observations for which information is available for all variables in *depvars*. Whether or not the estimation sample is heterogeneous across equations is stored in `e(common)`.

`cluster(`*clustvarlist*`)` specifies how `stackreg` clusters the standard errors (and covariances) at a higher level than the original unit of observation. By default, an identifier of the original observations serves as *clustvar*, because stacking the regression makes each original sampling unit contribute several observations to the stacked regression analysis. `stackreg` accommodates multiway higher-level clustering; that is, *clustvarlist* may consist of more than one variable. Multiway clustering requires the community-contributed command `cgmreg` (by Gelbach and Miller [2009]) to be installed. `stackreg` has been tested with `cgmreg` version 3.0.0. Other versions of `cgmreg` may behave differently and might make `stackreg` fail or produce incorrect results.

`df(adjust|raw|areg)` specifies the type of degrees-of-freedom adjustment `stackreg` applies. The default is `df(adjust)`. With `df(adjust)`, `stackreg` adjusts the degrees-of-freedom correction such that the reported standard errors coincide with those one gets from separately regressing the elements of *depvars* on *indepvars*, using `regress` with option `robust`. This, depending on how the `cluster()` option is specified, likewise applies to the standard errors one gets from `regress, cluster()` and `cgmreg, cluster()`, respectively. In the most simple case (no higher-level clustering, no panel data, homogeneous number of observations across *depvars*), the initially estimated variance–covariance matrix is adjusted by the factor $(N-1)/(N-1/G)$, with N denoting the genuine number of observations and G denoting the number of variables in *depvars*.[15]

For `xtstackreg`, the default—that is, `df(adjust)`—is to adjust the degrees-of-freedom correction such that the standard errors coincide with those from `xtreg, fe robust` and `xtreg, fe cluster()`, respectively. This implies that `xtstackreg`, by default, clusters the standard errors at the level of *panelvar*, which is the default with `xtreg, fe robust`. If `df(areg)` is specified, `xtstackreg` adjusts the degrees of freedom such that the standard errors match those from `areg, absorb(`*panelvar*`) robust` and `areg, absorb(`*panelvar*`) cluster()`, respectively. That is, with `df(areg)`, `stackreg` does not cluster the standard errors at the level of *panelvar* unless this is explicitly requested with `cluster(`*panelvar*`)`. `df(areg)` is ignored by `stackreg` if the `fe` option is not specified. `df(raw)` prevents `stackreg` from adjusting the degrees-of-freedom correction to the stacked regression setting. See section 3.4 for further details of the degrees-of-freedom adjustment that `stackreg` applies.

`wald` makes `test` and `testparm` apply a Wald rather than an F test after `stackreg`. This is achieved through preventing `stackreg` from saving the residual degrees of freedom in `e(df_r)`. With multiway clustering, as with heterogeneous estimation samples across the different regression equations, `e(df_r)` is never stored, because there is no (universal) answer to the question of what the number of clusters is. Thus, `test` and `testparm` apply a Wald test in these cases, even if the `wald` option is not specified.

`sreshape` requests that `stackreg` call the community-contributed command `sreshape` (Simons 2016) instead of `reshape`. Because `sreshape` is much faster than `reshape` (Simons 2016) in many settings, specifying `sreshape` may speed up `stackreg`.

`level(#)`; see [R] **Estimation options**. The reported confidence level can be changed by retyping `stackreg` without arguments and only specifying the `level(#)` option.

`edittozero(#)` specifies how close to 0 an element of the estimated variance–covariance needs to be to set its value to 0. The specified value is passed through to the Mata function `edittozero()`; see [M-5] **edittozero()**. The default is `edittozero(1)`.

15. With multiway clustering and the `nocommon` option, the match with the standard errors from `cgmreg` may not be perfect. See Cameron, Gelbach, and Miller (2011) for different approaches to the degrees-of-freedom correction in a multilevel-clustering setting; some are based on internal results not accessible via what `cgmreg` stores in `e()`.

The different estimation commands that are alternatively called by `stackreg` may differ with respect to how estimated coefficient variances that are close to 0 are dealt with. Specifying `edittozero()` aligns their behaviors.

`omitted` specifies that variables that were omitted because of collinearity be displayed and labeled as `(omitted)`. Unlike many Stata commands, the default is not to include in the results table any variables omitted because of collinearity. This is the default because `stackreg` regularly generates rather larger results tables due to *depvars* consisting of numerous variables. This applies in particular if factor variables are used. Hence, listing omitted variables may render the output hard to read.

`emptycells` specifies that empty cells for interactions of factor variables be displayed and labeled as `(empty)`. The default is not to include them in the results table, for the same reason as the default for the `omitted` option.

display_options: `noci`, `nopvalues`, `noomitted`, `vsquish`, `noemptycells`, `baselevels`, `allbaselevels`, `nofvlabel`, `fvwrap(`*#*`)`, `fvwrapon(`*style*`)`, `cformat(`*%fmt*`)`, `pformat(`*%fmt*`)`, `sformat(`*%fmt*`)`, and `nolstretch`; see [R] **Estimation options**.

4.3 Stored results

`stackreg` and `xtstackreg` store the following results in `e()`:

Scalars	
`e(N)`	number of observations (not expanded by stacking)
`e(k_eq)`	number of equations in `e(b)`
`e(N_g)`	number of groups (only stored with `xtstackreg` or the `fe` option)
`e(rank)`	rank of `e(V)`
`e(N_stack)`	number of observations in stacked regression
`e(N_clust)`	number of clusters
`e(df_r)`	residual degrees of freedom (only stored if `wald` is not specified and `e(common)`: `common` and `e(estimator)`: `regress`)
`e(df_m)`	model degrees of freedom (only stored if `e(common)`: `common`)
`e(N_`*l*`)`	number of observations *l*th equation (only stored if `e(common)`: `nocommon`)
`e(rank_`*l*`)`	rank of *l*th block (*l*th equation) of `e(V)` (only stored if `e(common)`: `nocommon` or `e(estimator)`: `cnsreg`)
`e(df_r_`*l*`)`	residual degrees of freedom *l*th equation (only stored if `e(common)`: `nocommon`)
`e(df_m_`*l*`)`	model degrees of freedom *l*th equation (only stored if `e(common)`: `nocommon` or `e(estimator)`: `cnsreg`)
`e(level)`	confidence level
Macros	
`e(cmd)`	`stackreg` or `xtstackreg`
`e(cmdline)`	command as typed
`e(depvar)`	names in *depvars*
`e(eqnames)`	names in *depvars*
`e(title)`	`Stacked Regression`
`e(estimator)`	`regress`, `cgmreg`, or `cnsreg`
`e(model)`	`ols` or `fe`
`e(common)`	`common` or `nocommon` (`nocommon` indicates that the estimation sample varies across equations)
`e(vcetype)`	`Clust. Robust`
`e(marginsok)`	predictions allowed by `margins`
`e(predict)`	program used to implement `predict`
`e(properties)`	`b V`

Matrices

`e(b)`	vector of estimated coefficients
`e(V)`	estimated coefficient variance–covariance matrix
`e(Cns)`	constraints matrix (only stored if `e(estimator): cnsreg`)

Function

`e(sample)`	marks estimation sample

4.4 stackreg postestimation

Using postestimation commands after `stackreg` is essential; `stackreg` is hardly an estimation command in its own right but is meant to facilitate postestimation inference. The postestimation commands available after `stackreg` are those available after `mvreg`[16] (see [MV] **mvreg postestimation**). After `stackreg`, these commands behave in the same way as they behave after `mvreg`. The most important postestimation command is `test` (`testparm`, respectively). For instance, `testparm *` can be used for testing the joint null hypothesis that no variable in *indepvars* has explanatory power for any of the variables in *depvars*.

5 Applications

In this section, we present two applications of `stackreg`. The first application illustrates the syntax of `stackreg` in a cross-sectional setting that includes weighting and demonstrates that `stackreg` reproduces the original estimation results for each equation. The second application shows how `stackreg` handles panel data and multiway clustering.

5.1 The persistent effects of Peru's mining Mita

To illustrate the application of `stackreg`, we in parts replicate and use data from Dell (2010). The data are available from the website of the *Econometric Society* and can be directly loaded—together with comprehensive documentation of Dell's empirical work—into a new subfolder (name `final`; size 1.13 GB) of Stata's current working directory.[17]

```
. quietly unzipfile "https://www.econometricsociety.org/sites/default/files/
> 8121_data_and_programs_0.zip"
```

Using a spatial regression discontinuity approach, Dell (2010) examines the long-run effects of a forced mining system in Peru and Bolivia, called *Mita*, that was in place in a clearly defined geographical area between 1573 and 1812. We focus on the regressions for which results are reported in table V, panel A, columns 6–8 (Dell 2010, 1886). Before carrying out the empirical analysis, some data preparation steps are necessary irrespective of `stackreg`. These steps and the replication of the original regressions are executed

16. More specifically these are `contrast`, `estat summarize`, `estat vce`, `estimates`, `forecast`, `hausman`, `lincom`, `margins`, `marginsplot`, `nlcom`, `predict`, `predictnl`, `pwcompare`, `test`, and `testnl`.
17. Only three data files of the huge package that is downloaded are actually required for running the current application: `gis_dist.dta`, `tribute1572.dta`, and `1572demographic.csv`.

by quietly running `dell_mita_prep.do`.[18] The equation-by-equation replication results are stored as `orig_Dell_sh_trib`, `orig_Dell_sh_boys`, and `orig_Dell_sh_women` by using `estimates store`.

```
. quietly do dell_mita_prep
```

As a regression-based balancing check, Dell (2010) examines whether the population composition differed between *Mita* regions and control regions before *Mita* came into force. Specifically, Dell regresses the population shares of men, boys, and females (`sh_trib`, `sh_boys`, `sh_women`) separately on numerous control variables and a *Mita* indicator labeled `pothuan_mita`, where observations are weighted by the square root of the district's total population. The estimation sample is restricted to districts that in terms of their capital are located no more than 50 kilometers distant from the boundary of the *Mita* area, with metropolitan Cusco being excluded from the sample. The output below displays the key results from these regressions, each equation being labeled with the name of its dependent variable. The coefficients of `pothuan_mita` are all close to 0 and statistically insignificant. However, separately testing the individual significance of these coefficients may not be an appropriate strategy for testing that *Mita* and non-*Mita* regions—conditional on controls—did not systematically differ prior to the implementation of *Mita*.

```
. estimates table orig_Dell_*, b(%9.7g) se(%9.7g) modelwidth(18)
> keep(pothuan_mita) stats(N)

-----------------------------------------------------------------------------
    Variable | orig_Dell_sh_trib   orig_Dell_sh_boys   orig_Dell_sh_women
-------------+---------------------------------------------------------------
pothuan_mita |          -.0063522            .0105358            -.0087577
             |           .0093512            .0121039             .0159997
-------------+---------------------------------------------------------------
           N |                 65                  65                   65
-----------------------------------------------------------------------------
                                                              legend: b/se
```

We next use `stackreg` to set the foundation for carrying out a presumably more appropriate regression-based joint balancing test. The output below illustrates that in terms of the coefficients and standard errors, `stackreg` yields exactly the same results as equation-by-equation estimation. The header of the output shows that 65 original observations contribute into the stacked regression and that they stem from 65 clusters; that is, the cluster variable `ubigeo` uniquely identifies the observations.

```
. stackreg sh_trib sh_boys sh_women = pothuan_mita x y x2 y2 xy x3 y3 x2y
> xy2 elv_sh slope bfe4* if (cusco!=1 & d_bnd<50) [aw=total_pop]
Stacked linear regression                         Number of obs    =       65
```

18. `dell_mita_prep.do` is available from the supplementary materials to this article. Major parts of the code in `dell_mita_prep.do` are borrowed from the supplementary materials to Dell (2010). `dell_mita_prep.do` saves the working data as `spec_check1572.dta` to the working directory.

```
------------------------------------------------------------------------------
             |               Clust. Robust
             |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
sh_trib      |
pothuan_mita |  -.0063522   .0093512    -0.68   0.499    -.0250334     .012329
           x |  -.0026342   .0116623    -0.23   0.822    -.0259323    .0206639
           y |   .0176393    .019031     0.93   0.357    -.0203796    .0556582
          x2 |  -.0124928   .0090399    -1.38   0.172    -.0305521    .0055665
          y2 |   .0153724   .0289802     0.53   0.598    -.0425222     .073267
          xy |  -.0358909   .0299991    -1.20   0.236     -.095821    .0240391
          x3 |   .0170006   .0100751     1.69   0.096    -.0031268    .0371279
          y3 |   .0026171   .0288595     0.09   0.928    -.0550363    .0602705
         x2y |   -.021492   .0161639    -1.33   0.188     -.053783    .0107991
         xy2 |  -.0610061   .0378611    -1.61   0.112    -.1366423    .0146301
      elv_sh |  -.0270248   .0131746    -2.05   0.044     -.053344   -.0007055
       slope |  -.0016688   .0010648    -1.57   0.122     -.003796    .0004585
      bfe4_1 |   .0432862   .0105804     4.09   0.000     .0221494     .064423
      bfe4_2 |   .0134728    .024949     0.54   0.591    -.0363685    .0633142
      bfe4_3 |  -.0283804   .0167855    -1.69   0.096    -.0619133    .0051525
       _cons |   .3255745   .0613156     5.31   0.000     .2030826    .4480665
-------------+----------------------------------------------------------------
sh_boys      |
pothuan_mita |   .0105358   .0121039     0.87   0.387    -.0136445    .0347161
           x |  -.0092446   .0267009    -0.35   0.730    -.0625859    .0440967
           y |  -.1103918   .0365711    -3.02   0.004    -.1834509   -.0373326
          x2 |   .0134455   .0133019     1.01   0.316    -.0131281    .0400191
          y2 |    .044934   .0427878     1.05   0.298    -.0405445    .1304125
          xy |  -.0636739   .0498789    -1.28   0.206    -.1633183    .0359706
          x3 |   -.005925   .0171742    -0.34   0.731    -.0402344    .0283845
          y3 |   .0810525   .0434243     1.87   0.067    -.0056976    .1678026
         x2y |   .0441965   .0212157     2.08   0.041     .0018133    .0865796
         xy2 |  -.0230336   .0501501    -0.46   0.648      -.12322    .0771527
      elv_sh |   .0052092   .0232938     0.22   0.824    -.0413254    .0517439
       slope |  -.0012942   .0014343    -0.90   0.370    -.0041596    .0015711
      bfe4_1 |   .0737769   .0262476     2.81   0.007     .0213413    .1262125
      bfe4_2 |  -.0357702   .0479029    -0.75   0.458    -.1314673    .0599269
      bfe4_3 |  -.0268769   .0177872    -1.51   0.136    -.0624109    .0086571
       _cons |    .147011   .1077998     1.36   0.177    -.0683439     .362366
-------------+----------------------------------------------------------------
sh_women     |
pothuan_mita |  -.0087577   .0159997    -0.55   0.586    -.0407208    .0232053
           x |   .0043607   .0298454     0.15   0.884    -.0552624    .0639838
           y |   .0725386   .0421132     1.72   0.090     -.011592    .1566693
          x2 |  -.0105578   .0104443    -1.01   0.316    -.0314226    .0103071
          y2 |  -.0199129   .0431044    -0.46   0.646    -.1060238     .066198
          xy |   .0949607   .0506969     1.87   0.066     -.006318    .1962394
          x3 |  -.0085574   .0197935    -0.43   0.667    -.0480996    .0309847
          y3 |  -.0305922   .0414253    -0.74   0.463    -.1133489    .0521644
         x2y |  -.0406215   .0178419    -2.28   0.026    -.0762648   -.0049783
         xy2 |   .0493898   .0533738     0.93   0.358    -.0572366    .1560162
      elv_sh |   .0369149   .0287867     1.28   0.204    -.0205931     .094423
       slope |   .0053587   .0015536     3.45   0.001      .002255    .0084623
      bfe4_1 |  -.0965441    .027226    -3.55   0.001    -.1509344   -.0421539
      bfe4_2 |   .0332263   .0482631     0.69   0.494    -.0631903    .1296429
      bfe4_3 |   .0565703   .0161224     3.51   0.001     .0243621    .0887786
       _cons |   .3841743   .1320239     2.91   0.005     .1204261    .6479224
------------------------------------------------------------------------------
```

Finally, after running stackreg, we test the joint hypothesis that none of the population shares differs between the two groups. This joint test is easily done using Stata's test command.

```
. test pothuan_mita
 ( 1)  [sh_trib]pothuan_mita = 0
 ( 2)  [sh_boys]pothuan_mita = 0
 ( 3)  [sh_women]pothuan_mita = 0
       F(  3,    64) =    0.58
            Prob > F =    0.6291
```

The output from test does not provide any evidence for systematic pretreatment disparities between *Mita* and non-*Mita* regions.

5.2 One Mandarin benefits the whole clan

We next illustrate additional features of stackreg—in particular, xtstackreg—by replicating in parts and using data from Do, Nguyen, and Tran (2017b). These data are published as Do, Nguyen, and Tran (2017a). Downloading the data requires an account at the Inter-university Consortium for Political and Social Research (ICPSR).

Do, Nguyen, and Tran (2017b) examine how the promotion of Vietnamese officials affects their hometowns. In their table 3 (Do, Nguyen, and Tran 2017b, 18), which we will focus on, the authors present results for six different dependent variables, three of which measure infrastructure investments (Infras3yr_Productive, Infras3yr_Information, and Infras3yr_EduHealth) and the other three of which measure regional outcomes (F2logComAvgInc, F2logComAvgExp, and F2logComPop). They obtain these results from six separate regressions that include the same set of independent variables. Unlike the first example application, for which a multiple-testing issue arises in connection with balancing checks, Do, Nguyen, and Tran (2017b) resembles a classical multiple-testing problem, where a bunch of outcome variables are considered to test the comprehensive hypothesis that state and party officials favor their hometowns.

Together with data-preparation steps, we run the original regressions quietly in do_mandarin_prep.do[19] and store the results as orig_Do_*.[20] To make the code easier to read, we place all control variables in the global macro controls. These regressions allow for commune fixed effects by using areg, absorb(ComID). Standard errors are clustered at the commune level, and the number of observations differs between equations. To accommodate including commune fixed effects, we declare the data to be panel data. Finally, we display the results of the one-to-one replication of Do, Nguyen, and Tran (2017b) as reference, using estimates table and focusing on the coefficients of key explanatory variable PowerCapital.

19. do_mandarin_prep.do is available from the supplementary materials to this article. Major parts of the code in do_mandarin_prep.do are borrowed from Do, Nguyen, and Tran (2017a).

20. More specifically, orig_Do_iprod refers to Infras3yr_Productive, orig_Do_iinfo refers to Infras3yr_Information, orig_Do_ieduH refers to Infras3yr_EduHealth, orig_Do_inc refers to F2logComAvgInc, orig_Do_exp refers to F2logComAvgExp, and orig_Do_pop refers to F2logComPop.

```
. global controls "c.logComAvgInc c.logComPop i.ComZone i.Year"

. quietly do do_mandarin_prep

. xtset ComID
       panel variable:  ComID (unbalanced)

. estimates table orig_Do_*, b(%9.7g) se(%9.7g) modelwidth(18)
> keep(PowerCapital) stats(N)

------------------------------------------------------------------------
    Variable |    orig_Do_iprod        orig_Do_iinfo        orig_Do_ieduH
-------------+----------------------------------------------------------
PowerCapital |          .1317135             .0781087             .0168334
             |          .0544799             .0470598             .0234967
-------------+----------------------------------------------------------
           N |              1237                 1237                 1237
------------------------------------------------------------------------
                                                           legend: b/se

------------------------------------------------------------------------
    Variable |     orig_Do_inc          orig_Do_exp          orig_Do_pop
-------------+----------------------------------------------------------
PowerCapital |         -.0110741            -.0109512             .0103588
             |          .0343692             .0273585             .0121625
-------------+----------------------------------------------------------
           N |              1023                 1023                 1012
------------------------------------------------------------------------
                                                           legend: b/se
```

Then we use `xtstackreg`[21] for the replication. Because the original set of regressions used different observations, we now specify the `nocommon` option. The output's header informs us that the stacked regression uses information from a total of 1,239 original observations. We furthermore specify the `cluster(ComID)` and `df(areg)` options because Do, Nguyen, and Tran (2017b) combine `areg` with standard errors clustered at the commune level. Using these options, `xtstackreg` yields the same point estimates and standard errors as the original code.

```
. xtstackreg Infras3yr_Productive Infras3yr_Information Infras3yr_EduHealth
> F2logComAvgInc F2logComAvgExp F2logComPop = PowerCapital ${controls},
> df(areg) nocommon cluster(ComID)

Stacked within-transformed linear regression       Number of obs    =    1,239
                                                   Number of groups =      334

                               (Std. Err. adjusted for 334 clusters in ComID)
-------------------------------------------------------------------------------
             |               Clust. Robust
             |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+-----------------------------------------------------------------
Infras3yr~ve |
PowerCapital |   .1317135   .0544799     2.42   0.016      .024935    .2384921
                                                  (output for controls omitted)
-------------+-----------------------------------------------------------------
Infras3yr_~n |
PowerCapital |   .0781087   .0470598     1.66   0.097    -.0141269    .1703443
                                                  (output for controls omitted)
```

21. Equivalently, we could have used `stackreg, fe`.

```
-------------+----------------------------------------------------------------
Infras3yr_~h |
PowerCapital |   .0168334   .0234967     0.72   0.474    -.0292193    .0628862
                                                 (output for controls omitted)
-------------+----------------------------------------------------------------
F2logComAv~c |
PowerCapital |  -.0110741   .0343692    -0.32   0.747    -.0784365    .0562884
                                                 (output for controls omitted)
-------------+----------------------------------------------------------------
F2logComAv~p |
PowerCapital |  -.0109512   .0273585    -0.40   0.689     -.064573    .0426705
                                                 (output for controls omitted)
-------------+----------------------------------------------------------------
F2logComPop  |
PowerCapital |   .0103588   .0121625     0.85   0.394    -.0134792    .0341968
                                                 (output for controls omitted)
------------------------------------------------------------------------------
```

We can now perform a joint test of whether power capital is related to any of the outcome variables by using `test`.

```
. test PowerCapital
 ( 1)  [Infras3yr_Productive]PowerCapital = 0
 ( 2)  [Infras3yr_Information]PowerCapital = 0
 ( 3)  [Infras3yr_EduHealth]PowerCapital = 0
 ( 4)  [F2logComAvgInc]PowerCapital = 0
 ( 5)  [F2logComAvgExp]PowerCapital = 0
 ( 6)  [F2logComPop]PowerCapital = 0
           chi2(  6) =   10.41
         Prob > chi2 =    0.1083
```

Though tests on individual significance suggest that officials' hometowns benefited in terms of higher investment in productive (`Infras3yr_Productive`) and information (`Infras3yr_Information`) infrastructure, on the basis of the joint test at the 10% level, we marginally cannot reject the null that it is immaterial for town-level outcomes, if people from that town reach high positions.

In this setting, multiway clustering may be an attractive alternative, say, because error terms are not only related within communes across years but also in each year across communes.[22] To use multiway clustering, we list the *varlist* of clustering variables as arguments in `cluster()`.

22. Given that the data cover only four years, there is reasonable doubt that this approach is sensible. However, it still illustrates how to handle multiway clustering with `stackreg` and `xtstackreg`.

```
. xtstackreg Infras3yr_Productive Infras3yr_Information Infras3yr_EduHealth F2
> logComAvgInc F2logComAvgExp F2logComPop = PowerCapital ${controls}, df(areg)
>  nocommon cluster(ComID Year)
Stacked within-transformed linear regression      Number of obs    =    1,239
                                                  Number of groups =      334
                            (Std. Err. adjusted for clustering on ComID Year)
─────────────┬────────────────────────────────────────────────────────────────
             │              Clust. Robust
             │      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
─────────────┼────────────────────────────────────────────────────────────────
Infras3yr~ve │
PowerCapital │   .1317135   .0470925     2.80   0.005     .0394139    .2240131
                                                     (output for controls omitted)
─────────────┼────────────────────────────────────────────────────────────────
Infras3yr_~n │
PowerCapital │   .0781087   .0307222     2.54   0.011     .0178942    .1383231
                                                     (output for controls omitted)
─────────────┼────────────────────────────────────────────────────────────────
Infras3yr_~h │
PowerCapital │   .0168334   .0306676     0.55   0.583     -.043274    .0769408
                                                     (output for controls omitted)
─────────────┼────────────────────────────────────────────────────────────────
F2logComAv~c │
PowerCapital │  -.0110741   .0336166    -0.33   0.742    -.0769614    .0548132
                                                     (output for controls omitted)
─────────────┼────────────────────────────────────────────────────────────────
F2logComAv~p │
PowerCapital │  -.0109512   .0281032    -0.39   0.697    -.0660326    .0441301
                                                     (output for controls omitted)
─────────────┼────────────────────────────────────────────────────────────────
F2logComPop  │
PowerCapital │   .0103588   .0148309     0.70   0.485    -.0187091    .0394268
                                                     (output for controls omitted)
─────────────┴────────────────────────────────────────────────────────────────
```

After rerunning `xtstackreg` with multiway clustering, we carry out another joint test regarding the effects of `PowerCapital` on the six considered outcomes. The new test result is more informative: the null of no effects is clearly rejected.

```
. test PowerCapital
 ( 1)  [Infras3yr_Productive]PowerCapital = 0
 ( 2)  [Infras3yr_Information]PowerCapital = 0
 ( 3)  [Infras3yr_EduHealth]PowerCapital = 0
 ( 4)  [F2logComAvgInc]PowerCapital = 0
 ( 5)  [F2logComAvgExp]PowerCapital = 0
 ( 6)  [F2logComPop]PowerCapital = 0
           chi2(  6) =   25.87
         Prob > chi2 =    0.0002
```

6 Conclusions

In this article, we introduced the `stackreg` command, which offers a convenient way to test hypotheses across multiple OLS regressions. `stackreg` goes beyond similar Stata commands in three aspects: first, it implements fixed-effects estimations; second, it allows for multiway clustering; and third, it enables cross-equation constraints.

7 Acknowledgments

We would like to thank Julia Lang, Johannes Ludsteck, Sabrina Schubert, and an anonymous reviewer for many valuable comments and suggestions. Excellent research assistance from Irina Simankova is gratefully acknowledged.

8 Programs and supplemental materials

To install a snapshot of the corresponding software files as they existed at the time of publication of this article, type

```
. net sj 21-2
. net install st0641      (to install program files, if available)
. net get st0641          (to install ancillary files, if available)
```

9 References

Anderson, M. L. 2008. Multiple inference and gender differences in the effects of early intervention: A reevaluation of the Abecedarian, Perry Preschool, and Early Training Projects. *Journal of the American Statistical Association* 103: 1481–1495. https://doi.org/10.1198/016214508000000841.

Cameron, A. C., J. B. Gelbach, and D. L. Miller. 2011. Robust inference with multiway clustering. *Journal of Business & Economic Statistics* 29: 238–249. https://doi.org/10.1198/jbes.2010.07136.

Dell, M. 2010. The persistent effects of Peru's mining *mita*. *Econometrica* 78: 1863–1903. https://doi.org/10.3982/ECTA8121.

Do, Q.-A., K.-T. Nguyen, and A. N. Tran. 2017a. Replication data for: One Mandarin benefits the whole clan: Hometown favoritism in an authoritarian regime. Ann Arbor, MI. Inter-university Consortium for Political and Social Research. https://doi.org/10.3886/E113594V1.

———. 2017b. One Mandarin benefits the whole clan: Hometown favoritism in an authoritarian regime. *American Economic Journal: Applied Economics* 9: 1–29. https://doi.org/10.1257/app.20130472.

Gelbach, J. B., and D. L. Miller. 2009. The community-contributed command cgmreg version 3.0.0. http://cameron.econ.ucdavis.edu/research/cgmreg.ado.

Gu, A., and H. I. Yoo. 2019. vcemway: A one-stop solution for robust inference with multiway clustering. *Stata Journal* 19: 900–912. https://doi.org/10.1177/1536867X19893637.

Lee, D. S., and T. Lemieux. 2010. Regression discontinuity designs in economics. *Journal of Economic Literature* 48: 281–355. https://doi.org/10.1257/jel.48.2.281.

List, J. A., A. M. Shaikh, and Y. Xu. 2019. Multiple hypothesis testing in experimental economics. *Experimental Economics* 22: 773–793. https://doi.org/10.1007/s10683-018-09597-5.

Pei, Z., J.-S. Pischke, and H. Schwandt. 2019. Poorly measured confounders are more useful on the left than on the right. *Journal of Business & Economic Statistics* 37: 205–216. https://doi.org/10.1080/07350015.2018.1462710.

Simons, K. L. 2016. A sparser, speedier reshape. *Stata Journal* 16: 632–649. https://doi.org/10.1177/1536867X1601600305.

StataCorp. 2019a. *Stata 16 Base Reference Manual*. College Station, TX.

———. 2019b. *Stata 16 Multivariate Statistics Reference Manual*. College Station, TX.

Weesie, J. 1999. sg121: Seemingly unrelated estimation and the cluster-adjusted sandwich estimator. *Stata Technical Bulletin* 52: 34–47. Reprinted in *Stata Technical Bulletin Reprints*. Vol. 9, pp. 231–248. College Station, TX: Stata Press.

Wooldridge, J. M. 2010. *Econometric Analysis of Cross Section and Panel Data*. 2nd ed. Cambridge, MA: MIT Press.

Zellner, A. 1962. An efficient method of estimating seemingly unrelated regressions and tests for aggregation bias. *Journal of the American Statistical Association* 57: 348–368. https://doi.org/10.2307/2281644.

About the authors

Michael Oberfichtner is a senior researcher at the Institute for Employment Research (IAB), Germany.

Harald Tauchmann is a professor of health economics at the Friedrich-Alexander-Universität Erlangen-Nürnberg, a guest researcher at RWI—Leibniz Institut für Wirtschaftsforschung in Essen, Germany, and a research fellow at CINCH—Health Economics Research Center in Essen, Germany. His research interests include health economics, applied econometrics, and statistical methods.