# Restricted, concentrated linear estimators

## Preliminaries

The setup is as in “Fast & Wild” section 6.1 except that 1) comes before in , just as it comes first in ; and constraints and nulls are expressed as rather than .

The main idea is to partial out as many of the exogenous controls as possible while maintaining the “attack surface" that the null hypothesis refers to—which anyway will typically consist of just endogenous variables. Partialling is a linear operation and can be computationally optimized much as the WCR is in “Fast & Wild.” After, the steps of the IV/GMM regression that are nonlinear in endogenous variables will involve fewer parameters and less computation.

For example, if the null is that the coefficient on the sole endogenous regressor is 0, we can partial out all exogenous variables. If the null refers to the coefficient on an exogenous control, then we can partial out all exogenous variables but that one.

To allow for arbitrary linear null hypotheses, as well as arbitrary linear model constraints, one calculation will be repeated enough that we create a notation for it. It serves to factor a vector space into the components that are within or perpendicular to a subspace. If is a matrix, then we form an eigendecomposition of the projection onto its column space, :

|  |  |
| --- | --- |
| in which |  |

This eigendecomposition construction does not uniquely define , but any fixed choice will do. Since is symmetric, is orthonormal. Within it, and are orthonormal bases for the column space of and its orthogonal complement, respectively. So and . I will signify these constructs with and superscripts. I will also *subscript* with these marks, but then I will mean them to be part of a name rather than indicating an operator. “” means that the symbol represents to the “perp” of .

## 2SLS

The estimator is

|  |  |
| --- | --- |
|  |  |

and the null is . We factor the column space of into two components, one to be retained, the other to be partialled out. To this end, define

|  |  |
| --- | --- |
|  |  |

Then the column space of is factored into components spanned by

|  |  |
| --- | --- |
|  |  |

In fact, (3) is dangerous as written because it might allow endogenous variation into , the part we want to partial out, which the FWL theorem does not support. For example, there could be two instrumented variables, with the null referring only to one. So for the purpose of carrying out the constructions in (3), we augment with columns referring to each of the endogenous variables, in the form . To limit notational complexity, this augmentation is not indicated. It does no harm if the added columns are collinear with those defining the null in so long as the matrix inverse in (1) is a generalized inverse; the construction works out the same because it is based on the projection onto the column space of , which is well defined even when columns of are collinear.

Since the partialled-out components are thus constructed to contain no endogenous components, they can be computed more efficiently via

where consists of the rows of corresponding to within .

Applying FWL allows us to shrink not only the regressor set, which is to start with, but also the instrument set . To set up this operation, we situate the columns of within the span of and then construct their orthogonal complement within this space, for inclusion among the retained instruments. The instruments that are retained and subjected to partialling-out are then

|  |  |
| --- | --- |
|  |  |

includes any variation in that is independent of , as well, potentially, as some variation that is correlated with —which is why will be partialled out of it. The computational virtue of is that if is wide, then is much narrower than since the latter contains all of (in the sense of column space span).

To set the stage for partialling out, plug the definitions (4) into the estimator (2) and use and :

The last line has recast the null hypothesis expression as pertaining to a 2SLS regression of on .Since the coefficient on in this hypothesis expression is , and since is constructed to only contain exogenous variation, by the FWL theorem, we may partial out and drop it from the regressor set:

where the last line implies the definition of the “concentrated” 2SLS estimator, ,of the coefficients just on .

## Restricted 2SLS

The maintained model constraints are . As in Fast & Wild, appendix 1, the restricted solution space is parameterized with

with

This parameterization works because it assures satisfaction of the constraint () and because has the requisite rank, so that the mapping must cover the solution space.[[1]](#footnote-2)

Substitute the parameterization into the regression equation:

|  |  |
| --- | --- |
| whence |  |

To parallel (hehe) previous definitions, put

Then by (6), restricted 2SLS (R2SLS) of on subject to the constraint is achieved by unrestricted 2SLS of on :

|  |  |
| --- | --- |
|  |  |

To concentrate out exogenous variation not entering the null hypothesis, the null expression is developed much as before. It starts by using :

To tame the typographic complexity, redefine some symbols for this more general context:

|  |  |
| --- | --- |
|  |  |

which entails

(Compare with (3) and (4).) If there are no constraints, and the redefinition has no effect: it thus neatly generalizes from the unconstrained case.

To build intuition about these expressions, it helps to distinguish between *data spaces*, *parameter/restriction spaces*. The columns of data matrices such as belong in the first. belongs in the second, as do and **;** the latter imply restrictions through orthogonality requirements. In the expressions above, the columns of span the subspace of the parameter/restriction space associated with that is orthogonal to—i.e., free from—the constraint associated with . With respect to the coordinate system in that subspace, the product and its basis identify the further subspace that is compatible with the constraint and in the attack surface for the null associated with . Likewise, identifies the orthogonal component, the component of variation permitted by the constraint that is outside the attack surface, which can be partialled out. and are bases for those two components in -restricted parameter space. and span the corresponding components in data space, the first to be retained, the second to be partialled out.

|  |  |
| --- | --- |
| With the redefinitions, the development of the null expression continues: |  |

in which

In sum, can be computed with the recipe for if, before we use the recipe, we redefine and according to (8) and if, at the end, we add .

## OLS and -class

ROLS is a special case of R2SLS, which can be reached by deleting from (7), yielding

*Concentrated,* *restricted* OLS partials out **:**

|  |  |
| --- | --- |
|  |  |

We may therefore blend (9) and (10) to write the expressions for the *restricted* and the *concentrated, restricted -class* *estimators* as

|  |  |
| --- | --- |
|  |  |

## Residuals

Just as in the application of FWL to OLS, the residuals from the concentrated regression are the same as those from the full regression. And they are needed to estimate the variance of the estimator. They are

|  |  |
| --- | --- |
|  |  |

## LIML

For RLIML, introduced in “Fast & Wild,” appendix 2, when identification is exact, . Otherwise,

|  |  |
| --- | --- |
|  |  |

where (“Fast & Wild,” eq. 81).To perform CRLIML (concentrated, restricted LIML), FWL lets us partial out of the various data matrices. So we can replace above with

|  |  |
| --- | --- |
|  |  |

Notice that in a typical application, the null will refer to the endogenous variables . As a result, the retained linear combinations of regressors, , willall be endogenous, as of course will be **.** In this respect, is an apt name for , and we won’t use the symbol anymore.

Incorporating FWL replaces (13) with

To base the WRE bootstrap DGP on (C)RLIML, we need to compute the residuals from both stages of the two-stage model after the DGP regression. The second-stage/structural residuals are in (12). Per (76) in “Fast & Wild” and the discussion following, the first-stage/reduced-form residuals are

Again, FWL lets us narrow the matrices involved:

|  |  |
| --- | --- |
|  |  |

# Application to the wild bootstrap

We start the WRE by obtaining CRLIML estimates and on the original data, and corresponding residuals, using (11), (12), and (15), with the value of derived as in (13). (Unrestricted, concentrated LIML is used if the null is not imposed on the DGP.)

For efficiency, the matrix, whose columns determine which variation in is concentrated out and which retained, can be made the same as it will be in the second stage. In particular in the DGP stage would include the columns expressing the null. That way, computations such as (4) and (5) need not be repeated. Typically this would happen regardless, since usually the null refers only to endogenous variables, and all are retained in both stages when partialling out.

The bootstrap data for clustering group in replication are

Since plugging in above will return the original data, we can also write

|  |  |
| --- | --- |
|  |  |

We will rely so much on this sort of formulation that for concision we redefine:

Then (16) is equivalent to

where is what is called in “Fast & Wild.” (Notice the triple dot over in the second line.) Next, we apply the definitions in the earlier discussion to the bootstrap data:  
It is also useful to arrange the DGP residuals in a way that aligns with the variables retained after the partialling out, :

where consists of the rows of corresponding to . This allows us to define the entire bootstrap as

|  |  |
| --- | --- |
|  |  |

If the attack surface for the null is entirely endogenous, then every column of is too. But if some row of the null coefficient refers only to exogenous variables (i.e., is non-zero only in entries corresponding to them), then a corresponding column of will be as well, and it will be the same for all replications. Necessarily, it works out that the corresponding column of , which is the source of the bootstrap variation, will be .

The WRE applies the user’s estimator, such as 2SLS or LIML, to the bootstrap data. The exogenous factors embedded in the above calculations, ,, and , are the same in all replications.

## The bootstrap estimator

For concision once more, redefine the symbols for the original data (but not bootstrap data) to mean that has been partialled out of them:

|  |  |
| --- | --- |
|  |  |

Applying (11) along with this redefinition, the null expression for the bootstrap restricted -class estimator in each replication is

|  |  |
| --- | --- |
|  |  |

In that, label the “denominator,”

The on is needed only to allow the estimator to be overidentified LIML, in which case is computed for each replication.

Recalling that consists of and , we see that to compute (19), we need the scalar quantities,

|  |  |
| --- | --- |
|  |  |

where and index columns of .

As for OLS in “Fast and Wild,” we can optimize and vectorize the computations. A building block is

is worked out in the same way. For the first expression in (20), we haveThis vectorizes horizontally over replications as

|  |  |
| --- | --- |
|  |  |

where the stars on the left size abuse notation, and where sums a matrix columnwise. To analyze the computational costs of various ways of rewriting this expression, distill and the rearrange it as

where is the number of columns of , which typically will be the number of excluded instruments; is the number of bootstrap clusters, and is the number of replications. The three lines above imply different computational strategies. The first incorporates when the computation is still linear, then takes the Hadamard product of the two -width matrices. The second and third versions further delay incorporation of , at the expense of creating a quadratic term in , whose computation could be carried out in the two ways shown. I estimate the computational costs of each line, in multiply-adds, as

(An isolated addition or multiplication counts as half a multiply-add.) The dominant terms are blue.

It is hard to construct realistic scenarios in which the first line is worse than the second or third. will be small unless there are many instruments. will typically be small except in the subcluster bootstrap. Also, since the computation will need to be carried out for each combination, it is efficient to compute each and once, as in the first line, and then compute that line for all the combinations. So I use the first line.

The development of , the second expression in(20), is more complicated:

using . Continuing,

|  |  |
| --- | --- |
|  |  |

is sparse. It does not need to be constructed if is non-empty, for then the non-zero entries can be individually subtracted from the next term. And normally at least contains the constant term.

## Bootstrap LIML

When the replication regression—not just the DGP regression—is LIML, we need to compute for each iteration, by solving the eigenvalue problem

The major terms here are those in (20), whose computation was just reviewed.

## Bootstrap Wald denominators

The bootstrap Wald statistics have a sandwich formula. In Stata, the convention is to take in the sandwich “filling,” regardless of the value of used elsewhere. This does simplify the computation. The multiway-clustered bootstrap Wald statistics are then

For each error clustering, the denominator—the central term above—is also expressed as

where

|  |  |
| --- | --- |
|  |  |

Much as in section 2.1, we decompose the task of computing into computing the scalar quantities

|  |  |
| --- | --- |
|  |  |

where and denote single columns and the subscript denotes the rows corresponding to cluster . To accommodate multiway clustering, the quantities (24) are initially computed for the intersection of all clusterings, a grouping indicated by . Then they are summed over each of the coarser clusterings.

There seem to be two good ways to compute (24). When the clusters are many, it is faster to compute and over all replications, producing two intermediate matrices of width , and then carry out (24). In this case, there is not much gain from collapsing the data early by cluster before involving the wide matrix . But when the clusters are few, it is faster to expand (24) quadratically in in order to further delay its involvement. Vectorizing over replications, the two factors in (24) expand as

|  |  |
| --- | --- |
|  |  |

Repeating a pattern seen earlier, in the last line, is sparse. It does not need to be constructed if is non-empty.

After is computed for each and for each , then for each combination, (24) can be computed for all at once with

This creates two temporary matrices. If that taxes memory, it may be better to sequentially compute each -indexed row of the result.

When there are few -indexed clusters, it is faster to defer involvement of . We have the quadratic expansion

|  |  |
| --- | --- |
|  |  |

Since the four major terms are scalar, we can transpose the second one. Also, we can rearrange the quadratic term to defer involvement of . And we can vectorize horizontally over replications. These changes lead, with the usual abuse of notation for vectorizing over , to

|  |  |
| --- | --- |
|  |  |

The four terms in (27) are of orders 0, 1, 1, and 2 in .The 0th-order term can be vectorized vertically over -clusters:

Similarly, the two 1st-order terms can be neatly vectorized over -clusters. Borrowing from (25),

where the “crosstab” notation in the last line is defined in “Fast & Wild.” Since one can prepare for the crosstab by computing for each and for each .

The quadratic term is computed as implied by (27), separately for each in an explicit loop.

## Alternative for the quadratic term in the coarse-clustered Wald denominators

I explored an alternative, which I record here even though it did not increase efficiency much in Mata, even as it added complexity. Possibly this alternative would perform better in a programming environment that supports higher-dimensional arrays. For each the quadratic term in (27) can be developed with

The two versions differ in where they put . Placed optimally, it narrows the (pre-computed) matrices that are multiplied against . Which is faster depends on whether is typically wide or tall, i.e., whether its column count, which is typically the number of excluded instruments, exceeds the average observation count per group.

In the first version, is the crosstab of by and clustering *except* that it counts only elements of group , so it only has non-zero entries in column . In the second version, sums the columns of by -cluster.

## One-way fixed-effects

As in “Fast & Wild” appendix 1, let be the matrix left-multiplication by which demeans data matrices within fixed-effect groups. I will construct here in a way slightly different from the one in F&W. If is the matrix that sums data by FE group then is the data matrix for the fixed effects (labeled in F&W), and

Or if there are observation weights ,

is a diagonal matrix whose entries are the reciprocals of the number of observations in each FE group (or if , the total weight). As in F&W, accommodating fixed effects mostly just requires applying to all the original (non-bootstrap) data matrices. In fact, just as we are taking the partialling out of from the original data matrices as implicit, through the notation in (16), we will do the same for the fixed effects.

But also as in “Fast & Wild,” handling fixed effects without explicitly constructing their data matrices requires adding a few terms to the formulas developed before.

### The null expression

The null expression (19) generalizes to

|  |  |
| --- | --- |
|  |  |

If , the methods developed in section 2.1 hold without typographic change because (upon recalling that by the redefinition just made, ).

But if , one term emerges when the null expression is expanded that contains an instance of that is not adjacent to a data matrix, and which therefore cannot be effected merely by demeaning within FE groups. In expanding (28), contains , which contains . This, along with , are subparts of , so we will focus on the latter. Expanding it with (17),

remains to trouble us only in the quadratic term at the end. It requires us to replace in (22) with

The crosstab term at the end is the main novelty introduced by fixed effects into the null expression. Unlike in the case of OLS, we cannot assume the term is zero even when the fixed effect grouping coheres with the bootstrap grouping (see text after (62) in F&W), at least when the regression is overidentified, for then the residuals may not be orthogonal to the FE dummies.

### The Wald denominator: granular approach

As for the filling in the Wald denominator sandwich as developed in section 2.3, and become and . In the granular approach, we compute those terms separately as

Here there is no typographical change from the non-FE formulas, once application of to non-bootstrapped data matrices is understood. Again the factor involving is more complex:

|  |  |
| --- | --- |
|  |  |

The novel term is in the middle. It contains , whose transpose, , can be constructed in analogy with , by using matrix index operations to vertically explode from to .

### The Wald denominator: coarse approach

In the coarse approach, the central computational challenge expressed in (24) becomes

The novel terms in the product arise from multiplying the first factor (which is typographically unchanged) with the novel term in the second, which appears in (29). The novel terms (to be subtracted) are

In the last form, holds the sums of each column of by FE cluster *restricting* to the cluster. The th column of is the th row of . Or we can focus on the -vector , the FE-wise sum of the -group elements of .

The part of the last form that is linear in can be vectorized vertically over :

1. Equivalent is . [↑](#footnote-ref-2)