The Stata Journal (2020)
20, Number 4, pp. 812–843 DOI: 10.1177/1536867X20976314

The Romano–Wolf multiple-hypothesis correction in Stata

Damian Clarke
Department of Economics
Universidad de Chile
Santiago, Chile
dclarke@fen.uchile.cl

Joseph P. Romano
Departments of Statistics and Economics
Stanford University
Stanford, CA
romano@stanford.edu

Michael Wolf
Department of Economics
University of Zurich
Zurich, Switzerland
michael.wolf@econ.uzh.ch

Abstract. When considering multiple-hypothesis tests simultaneously, standard statistical techniques will lead to overrejection of null hypotheses unless the multiplicity of the testing framework is explicitly considered. In this article, we discuss the Romano–Wolf multiple-hypothesis correction and document its implementation in Stata. The Romano–Wolf correction (asymptotically) controls the familywise error rate, that is, the probability of rejecting at least one true null hypothesis among a family of hypotheses under test. This correction is considerably more powerful than earlier multiple-testing procedures, such as the Bonferroni and Holm corrections, given that it takes into account the dependence structure of the test statistics by resampling from the original data. We describe a command, `rwolf`, that implements this correction and provide several examples based on a wide range of models. We document and discuss the performance gains from using `rwolf` over other multiple-testing procedures that control the familywise error rate.

Keywords: st0618, rwolf, bootstrap, familywise error rate, multiple-hypothesis testing, permutation methods, rwolf, stepdown procedure

1 Introduction

Advances in computational power and statistical programming languages such as Stata mean that, typically, the testing of multiple hypotheses in an empirical analysis is easy and quick to carry out. This often leads to a situation in which existing data sources or experiments are used to examine a number of hypotheses. Although the computational costs of such a situation are generally very low, there is a well-known statistical cost in cases of multiple-hypothesis testing. Namely, as the number of hypotheses considered in a given analysis grows, so too does the probability of falsely rejecting true null hypotheses. Starting from Bonferroni (1935), there has been considerable development of techniques that account for multiplicity in hypothesis testing.

In this article, we discuss the implementation of one such procedure, the Romano–Wolf multiple-hypothesis correction, described in Romano and Wolf (2005a,b, 2016). This procedure uses resampling methods, such as the bootstrap, to control the familywise error rate (FWER), that is, the probability of rejecting at least one true null hypothesis among the family of hypotheses under test.[1] The procedure is noteworthy given that, in addition to controlling the FWER, it offers considerably more power compared with earlier multiple-hypothesis correction procedures, such as Holm (1979) and Bonferroni (1935), where by "power" we mean the ability to correctly reject false null hypotheses. What is more, the Romano–Wolf procedure is able to eliminate a key assumption of previous resampling-based procedures such as the procedure described in Westfall and Young (1993), namely, the so-called subset pivotality assumption. We provide a discussion of the general nature of the multiple-hypothesis problem as well as a discussion of several multiple-testing procedures in section 2 of this article; a more detailed overview can be found in Romano, Shaikh, and Wolf (2010).

In this article, we focus on the control of the FWER, but this is certainly not the only error rate that can be considered in multiple-hypothesis testing. For example, a series of alternative procedures focus on controlling the false discovery rate, defined as the expected proportion of true null hypotheses rejected among all hypotheses rejected. Details of many such procedures, such as Benjamini and Hochberg's (1995) procedure, as well as earlier, less powerful techniques to control the FWER and their implementation in Stata can be found in Newson and the ALSPAC Study Team (2003) and Newson (2010). In general, false discovery-rate techniques are less conservative than FWER techniques, and are particularly common in genetic or biochemical applications where often a huge number of null hypotheses are considered (in the thousands or even tens of thousands). An illustrative applied discussion is provided in Anderson (2008).

The Romano–Wolf procedure is increasingly used in a range of fields, given the recognition of its relative power, computational efficiency, and generalizability. This multiple-hypothesis correction can be found in settings as diverse as finance (Liu, Patton, and Sheppard 2015), early childhood interventions (Gertler et al. 2014), and the evaluation of conditional cash transfers (Attanasio et al. 2014), to name but a few. Work by List, Shaikh, and Xu (2019) intended for an applied audience details how to implement the general Romano–Wolf methodology for specific applications in experimental economics. In line with the frequency of use of this procedure, this article provides a program, `rwolf`, to allow for its implementation simply in Stata in a broad range of circumstances.[2] Along with the theory underlying this program, we provide here some demonstrations chosen to illustrate the broad range of situations to which the Romano–Wolf multiple-hypothesis correction, and the `rwolf` command, is suited.

1. To be precise, the procedure only controls the FWER asymptotically, that is, as the sample size goes to infinity while the family of hypotheses under test remains fixed. But this is also the case for other multiple-testing procedures, such as the Bonferroni and Holm procedures, unless very strict distributional assumptions hold. Hence, for convenience of terminology, in this article, we will equate "control of the FWER" with "asymptotic control of the FWER".
2. An earlier version of this program is available by Clarke (2016).

In what remains of this article, we first provide a brief description of multiple-hypothesis testing and the basic notation used here before defining corrections for multiple hypotheses, and the Romano–Wolf procedure in particular, in section 2. We then define the syntax of the `rwolf` command in section 3 and provide several examples based on both simulated and real datasets in section 4. We conclude in section 5.

2 Procedures

2.1 Multiple-hypothesis testing procedures and definitions

Consider inference for a generic parameter θ. In what follows, we will refer to data used to estimate this parameter as X and the value estimated using these data as $\widehat{\theta}$. When a single null hypothesis H about θ is tested against a single alternative hypothesis H', a decision rule can be defined based on the rate of a type I error, defined as

$$\Pr_{\theta^0}\{\text{reject } H\} \tag{1}$$

The test could be a two-sided test, such as $H\colon \theta = \theta^0$ versus $H'\colon \theta \neq \theta^0$, or it could be a one-sided test, such as $H : \theta \leq \theta^0$ versus $H' : \theta > \theta^0$. The quantity θ^0 refers to the null value of interest defined by the researcher and is often $\theta^0 = 0$.[3] The type I error is defined when the null hypothesis is true, and as such, $\Pr_{\theta^0}$ refers to the probability of (falsely) rejecting the null when θ^0 is the actual value of the parameter. If the probability in (1) is bounded above by a value α, then α is the significance level at the test. Often, a particular value of α is chosen in advance, such as $\alpha = 0.05$, and the testing procedure is designed to meet this criterion. Alternatively, to avoid somewhat arbitrary criteria, one can report a p-value, here p, that fulfills (at least asymptotically)

$$\Pr_{\theta^0}\{p \leq \alpha\} \leq \alpha \tag{2}$$

for every $0 \leq \alpha \leq 1$ and is defined as the smallest value of α at which H would be rejected. Smaller values of p provide more evidence in favor of the alternative hypothesis H'.

The validity of this error rate of α assumes that a single null hypothesis is tested. We now extend the setting to a family of S null hypotheses $\{H_s\}_{s=1}^{S}$, which are related to parameters $\{\theta_s\}_{s=1}^{S}$, versus respective alternative hypotheses $\{H'_s\}_{s=1}^{S}$. If the error rate in (1) is only controlled at α one null hypothesis at a time, it is clear that the probability of committing at least one type I error across the S null hypotheses will generally exceed α.[4] Following Lehmann and Romano (2005b), we let $I \subseteq \{1, \ldots, S\}$ denote the set of true null hypotheses, and the FWER is then defined as

$$\text{FWER} := \Pr\{\text{reject at least one } H_s \text{ with } s \in I\}$$

3. For example, many of Stata's estimation, or `e()`, class of commands, such as `regress` and `ivregress`, by default report two-sided hypothesis tests where $\theta^0 = 0$.
4. This is often illustrated with a simple example assuming 1) independence of individual p-values, p_s, 2) equality instead of weak inequality in (2), and 3) all S null hypotheses being true, in which case the probability of falsely rejecting at least one true null hypothesis is equal to $1 - (1 - \alpha)^S$.

To account for multiple-hypothesis testing, we seek to control the FWER at level α. By definition, if all the null hypotheses are false, the FWER is equal to 0.

A traditional solution has been to implement the Bonferroni procedure, which consists of rejecting any H_s for which the corresponding p-value, p_s, satisfies $p_s \leq \alpha/S$. This procedure provides strong control[5] of the FWER (see, for example, Lehmann and Romano [2005a, 350]); however, it often has low power to detect false null hypotheses, particularly when the number of hypotheses is large. A procedure that has greater power, while still maintaining control of the FWER was described in Holm (1979). This is a stepdown procedure that begins by ordering the individual p-values such that $p_{(1)} \leq p_{(2)} \leq \cdots \leq p_{(S)}$, corresponding to hypotheses $H_{(1)}, H_{(2)}, \ldots, H_{(S)}$; in this way, the hypotheses are ordered according to their significance. $H_{(1)}$ is rejected if and only if (iff) $p_{(1)} \leq \alpha/S$, just as would be the case for Bonferroni. But if it is rejected, then $H_{(2)}$ is rejected if $p_{(2)} \leq \alpha/(S-1)$, which is a more lenient criterion. The procedure proceeds in this manner until at some step s, $p_{(s)} > \alpha/(S-s+1)$, and then it stops. (If the stopping criterion is never met, the procedure rejects all S hypotheses.) The Holm procedure rejects all hypotheses rejected by the Bonferroni procedure but potentially also rejects other hypotheses.

Both the Bonferroni and the Holm procedures are easy to implement and only require the list of individual p-values $\{p_s\}_{s=1}^{S}$. But this convenience comes at a (potentially) severe loss in power. The two procedures achieve control of the FWER by assuming a worst-case dependence structure among the p-values, which is close to the individual p-values being independent of each other. If the FWER is controlled in the worst case, it is always controlled. But if there is noticeable dependence among the p-values, it would be possible to maintain control of the FWER while increasing power at the same time.

We now illustrate this point with a simple example. Let's say there are $S = 100$ hypotheses under test, and one wants to control the FWER at level $\alpha = 0.05$. The p-values p_s are based on test statistics T_s that have a joint normal distribution with common variance 1 and common pairwise correlation ρ. The mean of all test statistics T_s for which H_s is true is equal to 0. The tests are one-sided in the sense that

$$p_s := \Pr\{Z \geq t_s\}$$

where $Z \sim \mathcal{N}(0,1)$ and t_s is the observed realization of T_s. We consider the following single-step procedure: Reject H_s iff $p_s \leq c_\rho$, where c_ρ is a common cutoff allowed to depend on ρ. The Bonferroni procedure uses $c_\rho := 0.05/100 = 0.0005$, irrespective of ρ. But as a function of ρ, the following, less conservative, cutoffs suffice to guarantee control of the FWER:

ρ	0	0.25	0.5	0.75	0.9	0.95	1
c_ρ	0.0005	0.0006	0.0012	0.0032	0.0089	0.0149	0.05

5. Strong control of the FWER means that the FWER will be no greater than α regardless of which of the null hypotheses are true. This contrasts with weak control of the FWER, which refers to when the FWER does not exceed α only in the case that all null hypotheses are true. Unless otherwise noted, in the remainder of the article, control of the FWER is equated with strong control.

(Apart from $\rho = 0$ and $\rho = 1$, the cutoffs c_ρ need to be determined by numerical simulations and are, therefore, subject to some small simulation error.) In particular, when $\rho = 1$, all p-values are perfectly correlated, meaning that there really is only a single shared hypothesis under test, in which case it is allowed to use the naïve cutoff $\alpha = 0.05$ itself. In general, the larger is ρ, the larger is the cutoff c_ρ, and thus, the easier it becomes to reject false hypotheses, which increases power. Similar considerations carry over to the more general case when pairwise correlations are not constant and to stepwise procedures.

Hence, as described above, if there exists noticeable dependence among the p-values, it would be possible to maintain control of the FWER while increasing power at the same time. To this end, resampling-based multiple-testing procedures have been proposed in the literature. By resampling from the observed data, one can mimic the true dependence structure of the p-values (or, equivalently, among the test statistics) and thus account for it (in an implicit fashion). An early such proposal goes back to Westfall and Young (1993), who use the bootstrap to account for the dependence structure of the p-values. This procedure (described in Westfall and Young [1993, algorithm 2.8] and summarized in appendix A of this article), assumes a certain subset pivotality condition, which is used in establishing FWER control. Roughly speaking, this condition means the following: Take any nonempty, strict subset of hypotheses (out of all hypotheses under test) and look at the random vector of corresponding test statistics. Then the joint distribution of this random vector is unique in the sense that it does not depend on whether the remaining hypotheses (not under test) are true or false. For a precise definition, see Romano and Wolf (2005b, example 2).

Subset pivotality can be violated in certain applications and is thus undesirable, because in such instances the Westfall–Young procedure only guarantees weak control of the FWER. An example where this condition is violated is given by jointly testing the entries of a correlation matrix. Take the case of independent and identically distributed multivariate observations of dimension three. The multiple-testing problem concerns the three (distinct) pairwise correlations

$$H_{ij} : \rho_{ij} = 0 \quad \text{for } 1 \leq i < j \leq 3$$

where ρ_{ij} denotes the correlation between components i and j of any of the observations. The individual test statistics are the sample correlations, denoted by $\widehat{\rho}_{ij}$, and regularity conditions are in place such that the (normalized) vector $(\widehat{\rho}_{12}, \widehat{\rho}_{13}, \widehat{\rho}_{23})'$ has an asymptotic normal distribution. Now assume that one is only interested in testing the subset $\{H_{12}, H_{13}\}$. Then the asymptotic distribution of the vector $(\widehat{\rho}_{12}, \widehat{\rho}_{13})'$ depends on ρ_{23}, so it depends on whether the remaining hypothesis (not under test) H_{23} is true or false; see Romano and Wolf (2005b, example 7) for further details.

Having said this, the procedure of Westfall and Young (1993) is available for Stata, as provided in Reif (2017), with additional discussion in Jones, Molitor, and Reif (2019).

More recently, the Romano–Wolf multiple-hypothesis correction has been proposed, as described in Romano and Wolf (2005a,b) as well as in the subsection below. This procedure also uses resampling and stepdown procedures to gain additional power by

accounting for the underlying dependence structure of the test statistics. However, and crucially, this procedure does not require the subset pivotality condition and is thus more broadly applicable than the Westfall–Young procedure.

2.2 The Romano–Wolf procedure(s)

Romano and Wolf (2005b) propose a procedure that controls the FWER at a given level α; hence, this procedure leads to a decision to reject or not reject for each null hypothesis H_s considered, for $s = 1, \ldots, S$. In follow-up work, Romano and Wolf (2016) describe how to compute p-values adjusted for multiple testing, which is a more flexible approach. Having adjusted p-values, one immediately knows which hypotheses to reject for any level α, as opposed to just for a specific level α. In other words, if one were to consider a different level α compared with a prior analysis, one would have to rerun the procedure of Romano and Wolf (2005b) with the new value of α; on the other hand, having adjusted p-values at hand, no new analysis is needed. The `rwolf` command returns the p-value adjustment documented in Romano and Wolf (2016), following Romano and Wolf's (2005b) Studentized StepM Procedure (algorithms 4.1–4.2). Here we describe the algorithms implemented, before turning to the precise syntax in the following section.

Prior to describing the p-value adjustment algorithm implemented in `rwolf`, we first describe the Romano and Wolf (2005b) procedure. The algorithm below is based on a bootstrap procedure; by default, `rwolf` is based on the standard Efron (1979) bootstrap (see Romano and Wolf [2005b, appendix B] for discussion). In section 4, we discuss extensions to alternative bootstrap methods, such as the block bootstrap.[6]

6. The Romano–Wolf procedure can also be implemented using permutation methods rather than bootstrap methods (Romano and Wolf 2005a); however, permutation tests of regression coefficients can result in rates of type I error that exceed the nominal size, so these methods are likely not ideal for such applications (DiCiccio and Romano 2017). Nevertheless, the `rwolf` command can be used with permutation testing following a generalization of the procedures described in section 4.2. That is, permutation tests can be used whenever the model exhibits a certain structure so that the so-called randomization hypothesis holds, as described in section 15.2 of Lehmann and Romano (2005b).

As above, suppose we wish to test S hypotheses, using data X. Each hypothesis H_s is associated with a parameter of interest θ_s, an estimator of this parameter $\widehat{\theta}_s$, and a corresponding standard error $\widehat{\sigma}_s$.[7] For notational convenience, we generally assume that $\theta_s^0 = 0$, for $s = 1, \ldots, S$. We assume further that the alternative hypotheses are either all one-sided of the type $H_s' : \theta_s > 0$ or all two-sided of the type $H_s' : \theta_s \neq 0$.[8] A Studentized test statistic for H_s is given by[9]

$$t_s := \frac{\widehat{\theta}_s}{\widehat{\sigma}_s} \tag{3}$$

Next consider M resampled data matrices of X, denoted by $X_1^*, \ldots, X_M^*$. They give rise to estimators, denoted by $\widehat{\theta}_s^{*,m}$, and corresponding standard errors, denoted by $\widehat{\sigma}_s^{*,m}$, for $m = 1, \ldots, M$. For each resample m and for each hypothesis H_s, a Studentized null statistic is given by

$$t_s^{*,m} := \frac{\widehat{\theta}_s^{*,m} - \widehat{\theta}_s}{\widehat{\sigma}_s^{*,m}} \tag{4}$$

These statistics $t_s^{*,m}$ are centered around 0 given that the numerator consists of a resample estimate minus the original estimate (rather than the null parameter), and as such, the distributions of $t_s^{*,m}$ will form the null distributions for the procedure.[10]

In case the alternative hypotheses are two-sided of the type $H_s' : \theta_s \neq 0$, it is important to work with the absolute values of the test statistics. To keep the notation uniform in the algorithm outlined below, in slight abuse of notation, we will use the following convention in the two-sided case (but not in the one-sided case):

$$t_s := |t_s| \qquad \text{and} \qquad t_s^{*,m} := |t_s^{*,m}|$$

Finally, as above, we will relabel hypotheses in order of significance—but now based on their test statistics t_s instead of their p-values p_s, as was done before by the Holm procedure—such that $H_{(1)}$ refers to the hypothesis with the largest corresponding test statistic [labeled $t_{(1)}$], and $H_{(S)}$ refers to that with the smallest [labeled $t_{(S)}$]. In what follows, we denote by $\max_{t,j}^{*,m}$ the largest value of the vector $\{t_{(j)}^{*,m}, \ldots, t_{(S)}^{*,m}\}$:

$$\max_{t,j}^{*,m} := \max\{t_{(j)}^{*,m}, \ldots, t_{(S)}^{*,m}\} \qquad \text{for } j = 1, \ldots, S \text{ and } m = 1, \ldots, M \tag{5}$$

For a given j, we denote by $\widehat{c}(1-\alpha, j)$ the empirical $1-\alpha$ quantile of the statistics $\{\max_{t,j}^{*,m}\}_{m=1}^{M}$. For example, in a case where one is testing $S = 4$ hypotheses, $\max_{t,1}^{*,m}$

7. In our terminology, a standard error of an estimator is an estimated standard deviation of the estimator.
8. One-sided hypotheses of the type $H_s' : \theta_s < 0$ can be accommodated as well by properly preprocessing the data. Both one-sided options are implemented in the `rwolf` command.
9. This is assuming that each hypothesis test is of the form $H_s : \theta_s = 0$. If instead the test is for a nonzero value, $H_s : \theta_s = \theta_s^0$, (3) should be changed to

$$t_s := \frac{\widehat{\theta}_s - \theta_s^0}{\widehat{\sigma}_s}$$

10. Equation (4) is also valid if the test is for a nonzero value, $H_s : \theta = \theta_s^0$, and needs not to be changed in such a case.

denotes the maximum value of $\{t^*_{(1)}, t^*_{(2)}, t^*_{(3)}, t^*_{(4)}\}$, $\max^{*,m}_{t,2}$ denotes the maximum value of the subvector $\{t^*_{(2)}, t^*_{(3)}, t^*_{(4)}\}$ (that is, the vector of test statistics corresponding to the three least significant hypotheses only), and so on. At last, we simply have $\max t^{*,m}_{(4)} = t^{*,m}_{(4)}$. An important consequence is that $\widehat{c}(1-\alpha, j)$ is weakly decreasing in j; that is, $\widehat{c}(1-\alpha, j) \geq \widehat{c}(1-\alpha, j+1)$, for $j = 1, \ldots, S-1$.

Based on the above, the principal stepdown multiple-testing procedure at level α (based on algorithm 3.1 from Romano and Wolf [2016]) can be summarized as follows.

1. For $s = 1, \ldots, S$, reject $H_{(s)}$ iff $t_{(s)} > \widehat{c}(1-\alpha, 1)$.
2. Denote by R_1 the number of hypotheses rejected. If $R_1 = 0$, stop; otherwise, let $j = 2$.
3. For $s = R_{j-1}+1, \ldots, S$, reject $H_{(s)}$ iff $t_{(s)} > \widehat{c}(1-\alpha, R_{j-1}+1)$.
4. a. If no further hypotheses are rejected, stop.
   b. Otherwise, denote by R_j the number of hypotheses rejected so far, and afterward let $j = j+1$. Then return to step 3.

As was the case with the procedure of Holm (1979), this correction is a stepdown procedure: $\widehat{c}(1-\alpha, j)$ is weakly decreasing in j and, as such, the criterion for rejection is more demanding for more significant hypotheses and becomes less demanding for less significant hypotheses. Given that the null distributions based on $\max^{*,m}_{t,j}$ are based on resamples of the original data, they implicitly account for the underlying dependence structure of the test statistics, leading to potentially considerable gains in power over the Holm procedure, which assumes a worst-case dependence structure.

The above algorithm leads to a decision whether to reject or not reject for each null hypothesis H_s at a given significance level α. However, additionally and perhaps more conveniently, a multiple-testing-adjusted p-value can be directly computed for each H_s, as described in the algorithm below. This algorithm is a generalization of a resample-based p-value for a single null hypothesis, which can be defined as (Romano and Wolf 2016; Davison and Hinkley 1997, 158).

$$p := \frac{\#\{t^{*,m} \geq t\} + 1}{M + 1} \tag{6}$$

Note that other definitions exist.[11] To generalize this to a situation where multiple hypotheses are considered, `rwolf` implements the following algorithm to compute the p-values using the distribution of $\max^{*,m}_{t,j}$, following Romano and Wolf (2016, algorithm 4.2).

11. Another (somewhat less conservative) common definition is

$$p := \frac{\#\{t^{*,m} \geq t\}}{M} \tag{7}$$

Either option can be implemented in the `rwolf` command, as described in section 3.

1. Define
$$p^{\text{adj}}_{(1)} := \frac{\#\{\max^{*,m}_{t,1} \geq t_{(1)}\} + 1}{M + 1}$$
2. For $s = 2, \ldots, S$,
   a. first let
   $$p^{\text{initial}}_{(s)} := \frac{\#\{\max^{*,m}_{t,s} \geq t_{(s)}\} + 1}{M + 1}$$
   b. and then enforce monotonicity by defining
   $$p^{\text{adj}}_{(s)} := \max\{p^{\text{initial}}_{(s)}, p^{\text{adj}}_{(s-1)}\}$$

3 The rwolf command

The Romano–Wolf multiple-hypothesis correction is implemented using the following syntax, returning the adjusted p-values described above, as well as the unadjusted p-values according to (6) [or (7), if `noplusone` is specified] for comparison.

```
rwolf depvars [if] [in] [weight] [, indepvar(varlist) method(string)
    controls(varlist) nulls(numlist) seed(#) reps(#) verbose strata(varlist)
    cluster(varlist) onesided(string) bl(string) iv(varlist) otherendog(varlist)
    indepexog noplusone nodots holm graph varlabels other_options
    nobootstraps pointestimates(numlist) stderrs(numlist) stdests(varlist)
    nullimposed]
```

where *depvars* refers to the multiple outcome variables that are being considered. There are two ways for this command to be used. First, `indepvar()` and `method()` must be specified if the complete Romano–Wolf procedure should be implemented including the estimation of bootstrap replications and generation of adjusted p-values. Alternatively, if the user is providing `rwolf` with precomputed bootstrap or permuted replications of the estimated statistic and standard errors for each of their multiple-hypothesis tests of interest, and only wishes for `rwolf` to calculate the adjusted p-values, then `nobootstraps`, `pointestimates(`*numlist*`)`, `stderrs(`*numlist*`)`, and `stdests(`*varlist*`)` should be indicated. `pweight`s, `aweight`s, `fweight`s, and `iweight`s are allowed; see [U] **11.1.6 weight**.

The first of these cases is provided for situations in which a single type of regression model is implemented with a range of outcome variables. In this case, `rwolf` can take care of the full procedure, including estimating the baseline models, and few details are required. The latter case is provided for more complex situations, such as cases where different models are used to test each hypothesis, where both dependent and independent variables change across models, or where more complicated resampling schemes are desired. Examples of each of these is provided in section 4.

3.1 Options

Standard options

`indepvar(`*varlist*`)` indicates the independent (treatment) variable that is included in multiple-hypotheses tests. This will typically be a single independent variable; however, it is possible to indicate various independent (treatment) variables that are included in the same model. The Romano–Wolf procedure will be implemented, efficiently returning p-values for each dependent variable of interest, corresponding to each of the specified independent variables. This option must be specified unless the `nobootstraps` option is indicated (see below).

`method(`*string*`)` indicates to Stata how each of the multiple-hypothesis tests (that is, the baseline models) are performed. Any standard regression–based estimation commands permitted by Stata may be specified, including `logit`, `probit`, `ivregress`, `regress`, and `areg`; see `help estimation commands` for a full list of estimation commands in Stata. The default is `method(regress)`. If instrumental-variables (IV) estimation is desired, `method()` must be specified with `ivregress` only, and the `iv()` option below must be specified.

`controls(`*varlist*`)` indicates additional controls that should be included in regressions of each *depvar* on the *indepvar* of interest. Any variable format accepted by *varlist* is permitted, including time series operators and factor variables.

`nulls(`*numlist*`)` indicates the parameter values of interest (θ_s^0 in section 2.2) used in each test. A single scalar value should be indicated for each of the multiple hypotheses tested, and these values should be listed in the same order as the variables listed as *depvars*. If multiple `indepvar()` are specified, null parameters should be specified grouped first by `indepvar()` and then by *depvars*. For example, if two independent variables are considered with four dependent variables, first the four null parameters associated with the first independent variable should be listed, followed by the four null parameters associated with the second independent variable. By default, each null hypothesis is assumed to be that the parameter is equal to 0.

`seed(#)` sets the seed to indicate the initial value for the pseudo-random-number generator. # may be any integer between 0 and $2^{31} - 1$.

`reps(#)` specifies to perform # bootstrap replications. The default is `reps(100)`. Where possible, a larger number (by a magnitude) of replications should be used for more precise p-values. In IV models, a considerably larger number of replications is highly recommended.

`verbose` requests additional output, including display of the initial (uncorrected) models fit. This will also generate a summary output message indicating the number of hypotheses rejected in uncorrected models and when implementing the Romano–Wolf procedure, as well as any dependent variables for which the null is rejected in the Romano–Wolf procedure.

`strata(`*varlist*`)` specifies the variables identifying strata. If `strata()` is specified, bootstrap samples are selected within each stratum when forming the resampled null distributions.

`cluster(`*varlist*`)` specifies the variables identifying resampling clusters. If `cluster()` is specified, the sample drawn when forming the resampled null distributions is a bootstrap sample of clusters. This option should always be specified when underlying models require cluster–robust inference. `cluster()` does not cluster standard errors in each regression; if desired, this can be specified using `vce(cluster` *clustvar*`)`.

`onesided(`*string*`)` specifies to calculate p-values based on one-sided tests. The default is p-values based on two-sided tests, corresponding to the null that each parameter is equal to 0 (or to the values indicated in `nulls()`). *string* must be either `positive`, in which case the null is that each parameter $\beta \geq 0$, or `negative`, in which case the null is that each parameter $\beta \leq 0$.

`bl(`*string*`)` allows for the inclusion of baseline measures of the dependent variable as controls in each model. If desired, these variables should be created with some suffix, and the suffix should be included in the `bl()` option. For example, if outcome variables are called `y1`, `y2`, and `y3`, then variables `y1_bl`, `y2_bl`, and `y3_bl` could be created with baseline values, and `bl(_bl)` would be specified.

`iv(`*varlist*`)` is only necessary when `method(ivregress)` is specified. The IVs for the treatment variable of interest should be specified in `iv()`. At least as many instruments as endogenous variables must be included.

`otherendog(`*varlist*`)` specifies additional endogenous variables if more than one endogenous variable is required in `ivregress` models. By default, when the option `method(ivregress)` is specified, it is assumed that the variable specified in `indepvar(`*varlist*`)` is an endogenous variable that must be instrumented. If this is the case, the variable should not be entered again in `otherendog()`.

`indepexog` should be specified if `method(ivregress)` is specified but `indepvar()` is an exogenous variable. In this case, all endogenous variables must be specified in `otherendog()`, and all instruments must be specified in `iv()`.

`noplusone` calculates the resampled and Romano–Wolf adjusted p-values without adding 1 to the numerator and denominator [that is, according to (7) rather than (6)].

`nodots` suppresses replication dots in bootstrap resamples.

`holm` specifies to provide p-values corresponding to the Holm (1979) correction along with the standard output. These will be included in the command output and stored in the final column of the matrix `e(RW)` (described in section 3.2 below).

`graph` requests that a graph be produced showing the Romano–Wolf null distribution corresponding to each variable examined. This graph shows the distribution of $\max_{t,j}^{*,m}$ from (5) for each j in $1, \ldots, S$. Examples of such a graph are provided in section 4.

`varlabels` specifies to name panels on the graph of null distributions using their variable labels rather than their variable names.

other_options are any additional options that correspond to the baseline regression model. All options permitted by the indicated method are allowed.

Options specific to cases where resampled estimates are user provided

`nobootstraps` indicates that no bootstrap replications be generated by the `rwolf` command. In this case, each variable indicated in *depvars* must consist of M bootstrap realizations of the statistic of interest corresponding to each of the multiple baseline models. Additionally, for each variable indicated in *depvars*, the corresponding standard errors for each of the M bootstrap replicates should be stored as another variable, and these variables should be indicated as `stdests(`*varlist*`)`. Finally, the original estimates corresponding to each model in the full sample should be provided in `pointestimates(`*numlist*`)`, and the original standard errors should be provided in `stderrs(`*numlist*`)`. `nobootstraps` may not be specified if `indepvar()` and `method()` are specified; for all standard implementations based on regression models, `indepvar()` and `method()` should be preferred.

`pointestimates(`*numlist*`)` provides the estimated statistics of interest in the full sample corresponding to each of the *depvars* indicated in the command. These estimates must be provided in the same order as the *depvars* are specified. `pointestimates()` may not be specified if `indepvar()` and `method()` are specified; for all standard implementations based on regression models, `indepvar()` and `method()` should be preferred.

`stderrs(`*numlist*`)` provides the estimated standard errors for each estimated statistic in the full sample. These estimates must be provided in the same order as the *depvars* are specified. `stderrs()` may not be specified if `indepvar()` and `method()` are specified; for all standard implementations based on regression models, `indepvar()` and `method()` should be preferred.

`stdests(`*varlist*`)` contains variables consisting of estimated standard errors from each of the M resampled replications. These standard errors should correspond to the resampled estimates listed as each *depvar* and must be provided in the same order as the *depvars* are specified. `stdests()` may not be specified if `indepvar()` and `method()` are specified; for all standard implementations based on regression models, `indepvar()` and `method()` should be preferred.

`nullimposed` indicates that resamples are centered around the null rather than the original estimate. `nullimposed` is generally used only when permutations rather than bootstrap resamples are performed.

3.2 Stored results

`rwolf` is an e-class program and returns many elements in the `e()` list. It returns scalars corresponding to each calculated Romano–Wolf p-value, which are available as `e(rw_`*depvar*`)`, where *depvar* refers to the name of each dependent variable. If multiple independent variables are considered, a scalar for each p-value corresponding to each independent–dependent variable pair will be returned as `e(rw_`*depvar_indepvar*`)`. A matrix is also returned as `e(RW)` providing the full set of Romano–Wolf-corrected p-values. If the `holm` option is indicated, p-values according to Holm (1979) will be returned in column 4 of this matrix. Again if multiple independent variables are considered, a matrix named `e(RW_`*indepvar*`)` will be returned corresponding to each *depvar*.

4 Some examples

4.1 Regression-based examples and performance

We begin by presenting a particularly simple case. Consider the following linear model, in which 10 outcome variables y^s are regressed on a single independent variable, `treat`. Superscript-s terms refer to each of the multiple outcomes, of which there are a total of S, or their determinants. The data-generating process (DGP) is simulated as

$$y_i^s = \beta_0^s + \beta_1^s \texttt{treat}_i + \varepsilon_i^s \quad \text{for } s = 1, \ldots, 10 \tag{8}$$

for observation i. For each observation i, the 10 stochastic error terms ε_i^s are drawn from a multivariate normal (MVN) distribution, following

$$\boldsymbol{\varepsilon}_i \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho & \ldots & \rho \\ \rho & 1 & \ldots & \rho \\ \vdots & \vdots & \ddots & \vdots \\ \rho & \rho & \ldots & 1 \end{bmatrix} \right)$$

with $\rho \geq 0$, and the independent variable of interest, $\texttt{treat}_i$, is generated as $\texttt{treat}_i := \mathbb{1}_{\{U_i > 0.5\}}$, where $\mathbb{1}$ denotes the indicator function and $U_i \sim U(0,1)$. This is a highly stylized setting, but it allows us to vary the correlation between the multiple outcome variables of interest $(y^1, y^2, \ldots, y^{10})$ via the parameter ρ, as well as the impact of treatment via the parameter β_1^s. In particular, we can examine both the empirical FWER and the empirical proportion of false null hypotheses that get rejected (that is, power). We can examine this performance not only for the Romano–Wolf procedure but also for Holm's procedure and for naïve procedures that do not account for multiple testing.

We consider a series of simulations based on $N = 100$ observations. Each null hypothesis is that $\beta_1^s = 0$, versus the two-sided alternative hypothesis $\beta_1^s \neq 0$. Across simulations, we vary the number of models in $k = 1, 2, \ldots, 10$ for which $\beta_1^s = 0$, as well as ρ, the correlation between outcomes induced by the stochastic error terms. Below, we document one such simulation and resulting multiple-hypothesis correction.

In the simulation below, each of the 10 β_1^s terms is equal to 0, and the correlation between draws in the MVN distribution is set at $\rho = 0.25$. We follow the DGP described above, where first we simulate $\texttt{treat}_i$ (as `treat`), and then we generate ε_i as a draw from an MVN. This draw is generated using a transformation of independent normal draws using a Cholesky decomposition of the desired covariance matrix and Stata's `matrix score` command to multiply row vectors.[12]

```
. set seed 130319
. set obs 100
number of observations (_N) was 0, now 100
. generate treat  = runiform()>0.5
. foreach num of numlist 1(1)10 {
  2.     generate c`num´     = rnormal()
  3. }
. local r=0.25
. #delimit ;
delimiter now ;
. matrix corr = (1,`r´,`r´,`r´,`r´,`r´,`r´,`r´,`r´,`r´ \
>                `r´,1,`r´,`r´,`r´,`r´,`r´,`r´,`r´,`r´ \
>                `r´,`r´,1,`r´,`r´,`r´,`r´,`r´,`r´,`r´ \
>                `r´,`r´,`r´,1,`r´,`r´,`r´,`r´,`r´,`r´ \
>                `r´,`r´,`r´,`r´,1,`r´,`r´,`r´,`r´,`r´ \
>                `r´,`r´,`r´,`r´,`r´,1,`r´,`r´,`r´,`r´ \
>                `r´,`r´,`r´,`r´,`r´,`r´,1,`r´,`r´,`r´ \
>                `r´,`r´,`r´,`r´,`r´,`r´,`r´,1,`r´,`r´ \
>                `r´,`r´,`r´,`r´,`r´,`r´,`r´,`r´,1,`r´ \
>                `r´,`r´,`r´,`r´,`r´,`r´,`r´,`r´,`r´,1);
. #delimit cr
delimiter now cr
. matrix corsim = cholesky(corr)
. foreach num of numlist 1(1)10 {
  2.     matrix eps`num´ = corsim[`num´,1..10]
  3.     matrix score epsilon`num´ = eps`num´
  4.     generate y`num´ = 1 + 0*treat + epsilon`num´
  5. }
```

12. Further discussion of this is provided in Gould (1995). The use of the Cholesky decomposition is a convenient way to generate a multivariate normal from a vector of uncorrelated random variables of mean 0 and variance 1, given that an uncorrelated draw of a vector $\mathbf{c} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ can be transformed following $\boldsymbol{\varepsilon} := \boldsymbol{\mu} + \mathbf{Lc}$, resulting in a vector $\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{LL}')$. Thus if $\boldsymbol{\Sigma} = \mathbf{LL}'$ is the desired covariance matrix, the Cholesky decomposition can be taken to give $\mathbf{L} = \text{Cholesky}(\boldsymbol{\Sigma})$, and the resulting $\mathbf{L}$ can be used to generate the correlated random draws with desired mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

```
. rwolf y1 y2 y3 y4 y5 y6 y7 y8 y9 y10, indepvar(treat) reps(1000) nodots holm
Bootstrap replications (1000). This may take some time.

Romano-Wolf step-down adjusted p-values

Independent variable:  treat
Outcome variables:   y1 y2 y3 y4 y5 y6 y7 y8 y9 y10
Number of resamples: 1000
```

Outcome Variable	Model p-value	Resample p-value	Romano-Wolf p-value	Holm p-value
y1	0.1142	0.1009	0.5534	0.8072
y2	0.8906	0.8931	0.9940	0.8931
y3	0.2750	0.2797	0.7872	1.0000
y4	0.0292	0.0280	0.1938	0.2517
y5	0.1914	0.1818	0.6883	1.0000
y6	0.8683	0.8741	0.9940	1.0000
y7	0.0137	0.0100	0.1009	0.0999
y8	0.8337	0.8372	0.9940	1.0000
y9	0.3849	0.3966	0.8382	1.0000
y10	0.1199	0.1149	0.5534	0.8042

Once we have simulated the $N \times S$ matrix $\mathbf{Y}$ in the above code, we apply the Romano–Wolf stepdown correction. Here we are considering the $S = 10$ outcome variables `y1`–`y10` and the single independent variable `treat`. We request that the command perform 1,000 bootstrap repetitions, which, given the lack of other options, will be performed by resampling observational units with replacement. By specifying the `nodots` and `holm` options, we request that no dots be displayed on the Stata output to indicate the degree to which the resample procedure has advanced, and we request for `rwolf` to return p-values corrected using Holm's (1979) procedure (which will be listed in the final column of tabular output and saved in the returned `e()` list).

The command returns a list of p-values associated with each of the multiple outcomes. The column `Model p-value` lists the analytical p-values coming directly from the estimated regression model based, in each case, on the t statistic and the (inverse) cumulative distribution function of a t distribution with appropriate degrees of freedom. The column `Resample p-value` lists the resampling-based p-values as per (6), which also do not correct for multiple testing. In the case of both of these uncorrected procedures, despite the fact that all true β_1 values were 0, many of the hypotheses that $\beta_1 = 0$ are rejected at $\alpha = 0.05$. In particular, the variables `y4` and `y7` have p-values below 0.05 for both uncorrected procedures. The third column displays the Romano–Wolf adjusted p-values, where we note that now no null hypothesis is rejected (even at $\alpha = 0.10$).

The simulated example above provides one example of a multiple-hypothesis correction based upon a known DGP. To examine the performance of the `rwolf` command (and the Romano–Wolf correction in this context) more generally, we can examine the error rates and proportion of hypotheses correctly rejected when we vary the number of values of $\beta_1^s = 0$ and the correlation between outcomes, ρ. We consider such an example in tables 1 and 2 (which are in the spirit of tables 1–3 in Romano and Wolf [2005a]).

The first table considers the proportion of times at least one null hypothesis is rejected when the null hypothesis is actually true (the FWER), and the second table considers the proportion of hypotheses that are correctly rejected when the null hypothesis is false (the power of the tests).[13]

In these tables, we consider 1) a series of models where all of the β_1^s terms are equal to 0 (presented in panel A of table 1), 2) a series where half of the β_1^s terms are equal to 0 and the other half are equal to 0.5 (presented in panel B of table 1 when considering FWERs and panel A of table 2 when considering power),[14] and 3) a series where all of the β_1^s terms are equal to 0.5 (presented in panel B of table 2). Case 1 is not considered in table 2 given that all null hypotheses are true and so cannot be "correctly rejected". Similarly, case 3 is not considered in table 1 given that all null hypotheses are false and so the FWER is trivially equal to 0 given that they cannot be "incorrectly rejected". Across columns, we vary the degree of correlation between outcomes, from $\rho = 0$ in the first two columns, to $\rho = 0.75$ in the final two columns. The nominal levels for the FWER are set at $\alpha = 0.05$ and $\alpha = 0.10$, respectively.

Table 1. Simulated error rates

	$\rho = 0$		$\rho = 0.25$		$\rho = 0.50$		$\rho = 0.75$	
	5%	10%	5%	10%	5%	10%	5%	10%
Panel A: All β_1=0								
FWER uncorrected	0.396	0.642	0.365	0.602	0.281	0.492	0.197	0.341
FWER Holm	0.035	0.094	0.036	0.084	0.029	0.068	0.021	0.046
FWER Romano–Wolf	0.048	0.100	0.049	0.097	0.046	0.097	0.047	0.096
Panel B: Half β_1=0.5								
FWER uncorrected	0.222	0.408	0.212	0.390	0.180	0.335	0.147	0.258
FWER Holm	0.024	0.065	0.028	0.061	0.025	0.052	0.025	0.049
FWER Romano–Wolf	0.029	0.067	0.033	0.067	0.034	0.075	0.040	0.083

NOTES: Error rates are documented familywise over all outcomes. Uncorrected error rates are displayed at the top of each panel, where the proportion refers to the proportion of simulations where at least one true null hypothesis was falsely rejected. Below, similar rates are displayed when p-values are corrected using Holm's and Romano–Wolf's procedures. The correlation between outcomes is varied across columns, and error rates at $\alpha = 0.05$ and $\alpha = 0.10$ are displayed. The number of repetitions is 1,000 per scenario, and the number of bootstrap resamples in each case is equal to $M = 5000$.

Here we briefly discuss the performance of the `rwolf` command and note several important features of the Romano–Wolf multiple-hypothesis correction. In particular, we always present the performance criteria for three testing procedures: those corresponding to the naïve case, where no correction for multiple-hypothesis testing is made; those corresponding to Holm's procedure; and those corresponding to the Romano–Wolf procedure.

13. The replication code of these results, and all results displayed in this article, is available at http://www.damianclarke.net/replication/multHypRWolf.zip.
14. Specifically, $\beta_1^1 = \beta_1^2 = \cdots = \beta_1^5 = 0$, and $\beta_1^6 = \beta_1^7 = \cdots = \beta_1^{10} = 0.5$.

In panel A of table 1 (where all values for $\beta_1^s = 0$), it is clear in the uncorrected case that the empirical FWERs greatly exceed nominal levels of $\alpha = 0.05$ and 0.10. When all outcomes are uncorrelated, these values are 0.396 and 0.642, respectively, suggesting a large proportion of families in which a null hypothesis is incorrectly rejected, in line with that predicted in theory.[15] As the correlation between outcomes grows, these naïve values fall closer to the nominal levels but still considerably exceed desired error rates.

With the Holm correction, we observe that the FWER is controlled at both the 5% and 10% levels. This is observed regardless of the degree of correlation considered, between $\rho = 0$ and $\rho = 0.75$. Similar control is observed with the Romano–Wolf procedure. Indeed, in each case the empirical FWER is very close to the desired nominal rate of 0.05 or 0.10, respectively. In panel B of table 1 (where 5 of the 10 hypotheses should not be rejected), we again observe that the FWER without multiple-hypothesis correction still substantially exceeds desired error rates but is successfully controlled at no more than α with both the Holm and Romano–Wolf procedures.

Table 2. Simulated rejection rates

	$\rho = 0$		$\rho = 0.25$		$\rho = 0.50$		$\rho = 0.75$	
	5%	10%	5%	10%	5%	10%	5%	10%
Panel A: Half β_1=0.5								
Rejected uncorrected	0.687	0.791	0.689	0.797	0.681	0.789	0.693	0.798
Rejected Holm	0.324	0.460	0.325	0.457	0.325	0.453	0.340	0.468
Rejected R–W	0.373	0.486	0.382	0.492	0.401	0.519	0.469	0.594
Panel B: All β_1=0.5								
Rejected uncorrected	0.683	0.792	0.689	0.794	0.681	0.788	0.694	0.797
Rejected Holm	0.384	0.547	0.406	0.558	0.409	0.552	0.432	0.564
Rejected R–W	0.416	0.558	0.436	0.576	0.458	0.593	0.519	0.651

NOTES: The rate of correctly rejected (false) null hypotheses is displayed, where rates refer to the proportion of all hypotheses rejected across all simulations where the null hypothesis is false. "Uncorrected" rejection rates are for cases where naïve p-values are used for each test, "Holm" rejection rates are based on Holm's p-value correction, and "R–W" rates are based on Romano–Wolf's p-value correction. The correlation between outcomes is varied across columns, and rejection rates at $\alpha = 0.05$ and $\alpha = 0.10$ are displayed. The number of repetitions is 1,000 per scenario, and the number of bootstrap resamples in each case is equal to $M = 5000$.

In table 2, we can examine the relative power of the Holm and Romano–Wolf procedures for rejecting false null hypotheses. In panel A of table 2, we return to the case considered in panel B of table 1, here considering the power of the tests instead. In this setting, the naïve case of no multiple-hypothesis correction allows us to reject a large proportion of false null hypotheses (at the cost of the FWER documented in table 1).

15. As discussed in section 2, where outcomes are uncorrelated, the probability of falsely rejecting at least one hypothesis is $1-(1-\alpha)^S$. In this case in particular, we would thus expect the proportion to be $1-(1-0.05)^{10} = 0.401$ and $1-(1-0.10)^{10} = 0.651$ at the 5% and 10% levels, very close to the empirically observed values.

In the case of Holm and Romano–Wolf, we observe relatively similar rates of correct rejection when the correlation between outcomes is 0, but as expected, as the correlation between outcomes grows, the Romano–Wolf procedure substantially improves relative to Holm's procedure. For example, in the final column of panel A, we reject 59.4% of false null hypotheses using the Romano–Wolf procedure versus only 46.8% using Holm's procedure, a 27% improvement in rejection rate. A similar pattern is observed in table 2, panel B (where all null hypotheses are false). Initially, when the correlation between outcomes is 0, Holm's and Romano–Wolf's procedures have similar power; however, to the degree that ρ increases, the Romano–Wolf procedure becomes relatively more powerful.

In table 2, we consider the relative power of testing procedures for rejecting false null hypotheses with a particular value for β_1^s, in this case, $\beta_1^s = 0.5$ for all cases where $\beta_1^s \neq 0$. The relative power of these procedures will depend on the actual value of β_1^s. Below, in figure 1, we consider how these rates of rejection vary with β_1^s values. In the figure, we consider the case corresponding to table 2 panel B (where all $\beta_1^s \neq 0$) and where $\rho = 0.5$. We present values of β_1^s varying from 0.01 to 1, in steps of 0.03. All other details follow the DGP in (8).

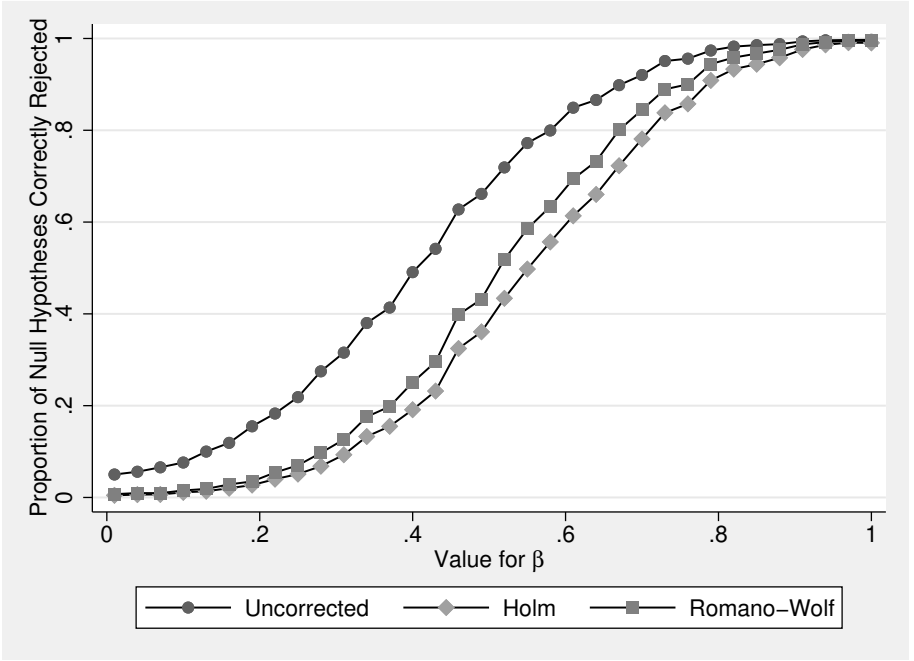


Figure 1. Comparative power to reject false null hypotheses.

NOTES: For each value of β_1 in $\{0.01, 0.04, 0.07, \ldots, 0.97, 1\}$, we conduct 1,000 simulations following the DGP laid out in (8), where each β_1^s is set to the value indicated on the x axis and $\rho = 0.5$. In each case, $N = 100$ observations are simulated, and we calculate p-values using the `rwolf` command with 200 bootstrap resamples.

For each value of β_1, we run 1,000 simulations, in each case calculating the unadjusted p-values, and the Holm and Romano–Wolf p-values using the `rwolf` command. Across the 1,000 simulations, we examine the proportion of null hypotheses correctly rejected. As expected, the power when not conducting any multiple-hypothesis correction is greatest (at the cost of exceeding the FWER if the null hypothesis were true). To the degree that the correlation between outcomes approaches $\rho = 1$, the power of the Romano–Wolf procedure will approach the power of the procedure with no multiple-hypothesis correction.

Of interest is the relative performance of Romano–Wolf's versus Holm's correction. Here, given that $\rho = 0.5$, the power of the Romano–Wolf correction dominates the power of the Holm correction across each value for β_1. This difference is particularly notable at values of β_1 between 0.4 and 0.6, and all disappear as β_1 becomes large, implying sufficient power to reject all null hypotheses regardless of the correction procedure. When $\beta_1 = 1$, each of the procedures results in rejection rates of around 100%.

The simulated examples based on `rwolf` displayed previously provide a standard implementation where multiple outcome variables are regressed on a single independent (treatment) variable, each using an ordinary least-squares regression. However, the standard command syntax of `rwolf` allows for many extensions of this baseline implementation. This includes the use of alternative estimation methods (for example, IV regression, probit, and other Stata estimation commands), the implementation of one-sided tests, or the use of alternative bootstrap routines (for example, block and stratified bootstraps). To document several such extensions, we turn to an alternative example below, modeled after a simple IV regression example based on system data, as described in the help file of Stata's `ivregress`. Here we extend to a case with multiple outcomes and examine both a one- and a two-sided test.

We begin with a (default) two-sided test, where we follow the implementation of the two-stage least-squares estimate from the `ivregress` help file.[16] We use the same specification, where along with the outcome variable `rent`, we also consider three other variables: `popden`, `popgrow` and `hsng`. We do not make any claim to causality or consistency of the resulting estimates. These are simply shown as an illustration of the `rwolf` command with an IV regression. In each case, the independent variable of interest is `hsngval`, and this is instrumented with the variables indicated in the `iv()` option. We include `pcturban` as an additional control. We use 10,000 bootstrap replications (bootstrapping on observational units) and request a graph of the null distributions used in testing be reported (as discussed below).

The setup and output of this Romano–Wolf correction is displayed below. In this example, the p-values from the original IV models are quite low, suggesting evidence against the null that each coefficient on the variable `hsngval` is 0. The Romano–Wolf correction displayed in the final column results in inflated p-values (as designed), though the adjusted p-values are still relatively low.

16. The implementation there is `ivregress 2sls rent pcturban (hsngval = faminc i.region)`.

```
. set seed 120113031
. use http://www.stata-press.com/data/r13/hsng, clear
(1980 Census housing data)
. rwolf rent popden popgrow hsng, indepvar(hsngval) method(ivregress)
> iv(faminc i.region) reps(10000) graph nodots
> controls(pcturban)
Bootstrap replications (10000). This may take some time.

Romano-Wolf step-down adjusted p-values

Independent variable:  hsngval
Outcome variables:   rent popden popgrow hsng
Number of resamples: 10000
```

Outcome Variable	Model p-value	Resample p-value	Romano-Wolf p-value
rent	0.0000	0.0157	0.0157
popden	0.0654	0.0848	0.0848
popgrow	0.0019	0.0119	0.0200
hsng	0.0236	0.0120	0.0515

Below, we document the same procedure but here conducting one-sided hypothesis tests. In each case, the null hypothesis in these tests will be of the form $H_0 : \beta_1 \leq 0$, versus the alternative $H_1 : \beta_1 > 0$, where β_1 is the coefficient on `hsngval` in the second stage of the IV regression. For the sake of illustration, we multiply the two outcomes by -1, such that the sign on $\widehat{\beta}_1$ estimated in each IV regression is greater than 0. Along with the syntax described previously, the implementation of one-sided tests requires the use of the option `onesided()`. If `onesided(negative)` is specified, the null will be $H_0 : \beta_1 \leq 0$; that is, negative values will provide more support for the null. On the other hand, if `onesided(positive)` is specified, the null will be $H_0 : \beta_1 \geq 0$ in each case, such that positive values will provide more support for the null.

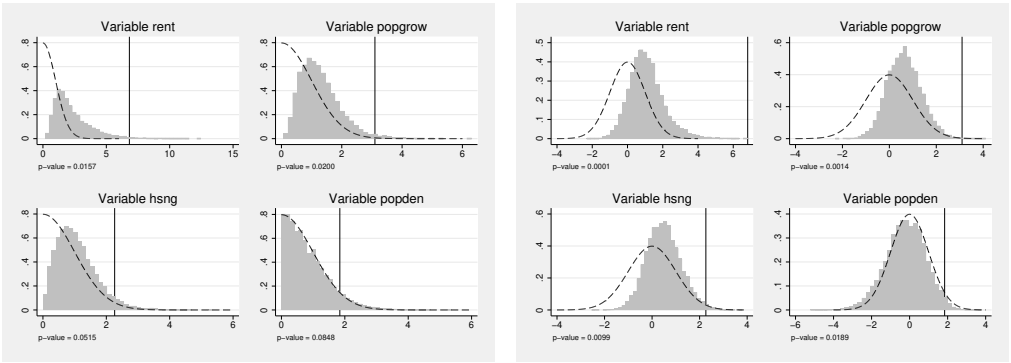
```
. replace popden=-popden
(50 real changes made)
. replace hsng  =-hsng
(50 real changes made)
. rwolf rent popden popgrow hsng, indepvar(hsngval) method(ivregress)
> iv(faminc i.region) reps(10000) graph onesided(negative) nodots
> controls(pcturban)
Bootstrap replications (10000). This may take some time.

Romano-Wolf step-down adjusted p-values

Independent variable:  hsngval
Outcome variables:   rent popden popgrow hsng
Number of resamples: 10000
```

Outcome Variable	Model p-value	Resample p-value	Romano-Wolf p-value
rent	0.0000	0.0001	0.0001
popden	0.0327	0.0189	0.0189
popgrow	0.0010	0.0009	0.0014
hsng	0.0118	0.0050	0.0099

In each example, we specified the `graph` option to request as output the null distributions used to calculate the p-value in each case. Examining these distributions allows us to empirically observe how much more demanding the Romano–Wolf correction is compared with an uncorrected test.

In figure 2 panel (a), we display these distributions in the two-sided case. Here the histogram presents the absolute value of each (Studentized) estimate from the bootstrap replications where the null is imposed, and the black dotted line presents an exact half-normal distribution. The actual Studentized value of the regression coefficient is displayed as a solid vertical line. In the top left panel, the first null distribution is considerably more demanding than the theoretical distribution, given that it accumulates the maximum coefficient estimated across each outcome. These null distributions become increasingly less demanding in the top right and bottom left panels because previously tested variables are removed from the pool to form the null distributions. Finally, in the bottom right corner (for the least significant variable), the null distribution is based on bootstrap replications from only this variable, and as such, the null distribution closely approximates the theoretical half-normal distribution.

In figure 2 panel (b), we present the same null distributions but now based on the one-sided tests. Here the histogram documents the maximum values across the multiple variables in each bootstrap replication, and the black dotted line presents the theoretical normal distribution. Once again, when we consider outcomes for which more significant relationships are observed, the empirical distribution used to calculate the corrected p-value is considerably more demanding than the black dotted distribution, which would be used under no correction and a normality assumption. In the case of the least significant variable (`popden`), these two distributions are similar given that the null distribution is based only on Studentized regression estimates of the single regression where this is the outcome variable.



(a) Two-sided tests (b) One-sided tests

Figure 2. Null distributions for one- and two-sided tests with IV models.

NOTES: Panel (a) documents the null distributions used to calculate the Romano–Wolf adjusted p-values for each of the four outcome variables of interest in the `ivregress` command, using a two-tailed test where the null is that $\beta_1 = 0$ in each case. Panel (b) documents the null distributions for the same regressions but now based on the one-tailed test where each null is that $\beta_1 \leq 0$.

4.2 A nonstandard Studentized example

Each previous example has been based on the simple `rwolf` command syntax where a single independent variable is regressed on multiple outcome variables. This is frequently sufficient for a large number of implementations, such as cases where a single experimental treatment is assigned, and various outcomes are measured. In this case, the `rwolf` command can be implemented in one line and takes care of everything, including the full process of bootstrap draws, estimation of regression coefficients and standard errors, as well as the generation of null distributions and p-values. However, Romano–Wolf p-values can also be calculated for more complex setups, if the user wishes to pass the command the already-bootstrapped (or permuted) statistics and standard errors that have been calculated from the underlying models of interest. We document this flexibility in what remains of this section, using a bootstrap approach where our statistics of interest are a series of correlations.

We use an example and data documented in Westfall and Young (1993), which was also previously used to demonstrate the effectiveness of the Romano–Wolf procedure in Romano and Wolf (2005a). Although we refer to this as a nonstandard example, it is only nonstandard in the way it interacts with the syntax of `rwolf`, given that the multiple tests are based on several independent variables and as such do not allow for a single independent variable to be indicated using the `indepvar()` option.

This example considers pairwise correlations between state-average standardized Scholastic Aptitude Test (SAT) scores and several other state-level measures for the 48 mainland U.S. states plus Hawaii. These data—consisting simply of state-level measures of several variables of interest—are provided in table 6.4 of Westfall and Young (1993, 197) as well as in the replication materials for this article. The variables considered are `satdev` (the residuals from a regression of state-level SAT scores on the square root of the percent of students taking the exam in a given state), the student/teacher ratio in the state, the teacher salary, the percent of the population that is black, and the crime rate of each state.

In this case, the statistics of interest refer to simple correlations between pairs of variables of interest, and p-values will be corrected for the fact that we are testing 10 hypotheses. To construct null distributions following (4) and (5), the `rwolf` command requires an estimate of the original parameter of interest in each case (the pairwise correlation) along with a standard error. These values are used to construct t_s (as defined in section 2.2) for each of the s multiple hypotheses and to order the hypotheses in terms of their relative significance. The command additionally requires the results from M bootstrap replicates, where a resampled estimate of each statistic and its standard error is provided. We document such a case below, where in each of the multiple hypotheses the parameter of interest is the correlation between variables ρ (which can be simply calculated using `corr` in Stata) and its standard error, which, assuming normality, is calculated as (Bowley 1928; Zar 2010)[17]

$$\mathrm{SE}_{\widehat{\rho}} := \sqrt{\frac{1-\widehat{\rho}^2}{N-2}} \tag{9}$$

To generate the various components that `rwolf` uses to implement the multiple-hypothesis correction, we first open the data and define in locals `var1` and `var2` the corresponding pairs to be tested. The idea in these locals is that we wish to calculate the correlation between the first variable listed in `var1` and the first variable listed in `var2`, the correlation between the second variable listed in `var1` and the second variable listed in `var2`, and so forth, through the correlation between the last variable listed in `var1` and the last variable listed in `var2`. We calculate these correlations one by one in the `foreach` loop in the code below. We loop over elements of the local `var1`, and we take corresponding elements one by one from local `var2` using Stata's `tokenize` command.[18] These correlations and standard errors from the original data are then saved, respectively, as locals `c‘i’` and `s‘i’` to be later passed to the `rwolf` command. Finally, we generate two series of 10 empty variables, `rho1`–`rho10` and `std1`–`std10`, which will later be populated by resample-based correlation estimates and their standard errors.

17. In appendix B, we document how this test can be implemented using regression rather than correlation.

18. Using the `tokenize` command means that we can refer to the variables in `var2` as `‘1’`, `‘2’`, ..., `‘10’` in each iteration of the loop.

```
. use "satgenerated", clear
. set seed 13032019
. local var1 satdev salary black satdev satdev ratio ratio satdev salary ratio
. local var2 black crime crime ratio crime crime black salary black salary
. tokenize `var2´
. local i=1
. foreach var of varlist `var1´ {
  2.     quietly corr `var´ ``i´´
  3.     local c`i´=r(rho)
  4.     local s`i´=sqrt((1-r(rho)^2)/(r(N)-2))
  5.     local ++i
  6. }
. foreach num of numlist 1/10 {
  2.     quietly generate rho`num´=.
  3.     quietly generate std`num´=.
  4. }
```

A bootstrap procedure is then implemented based on 5,000 resamples. We initially expand the dataset to have 5,000 observations to store each of the bootstrap estimates; however, prior to calculating the correlations and standard errors in each replicate, we work only with the 49 state-level observations with SAT data. Each replicate is implemented in the main `forvalues` loop below. Within each of these 5,000 iterations, we first issue a `preserve` command to later `restore` the data toward the end of each iteration; this way, the bootstrap resample (`bsample`) begins with the original data in each iteration. Within each loop, lines 7–12 calculate the bootstrap correlations and corresponding standard errors, which are then filled in line by line in the variables `rho1`–`rho10` and `std1`–`std10` at the end of each loop.

```
. local M=5000
. set obs `M´
number of observations (_N) was 49, now 5,000
. forvalues m=1/`M´ {
  2.     preserve
  3.     quietly keep if sat!=.
  4.     bsample
  5.     tokenize `var2´
  6.     local s=1
  7.     foreach var of varlist `var1´ {
  8.         quietly corr `var´ ``s´´
  9.         local cor`s´ = r(rho)
 10.         local std`s´ = sqrt((1-r(rho)^2)/(r(N)-2))
 11.         local ++s
 12.     }
 13.     restore
 14.     foreach s of numlist 1/10 {
 15.         quietly replace rho`s´=`cor`s´´ in `m´
 16.         quietly replace std`s´=`std`s´´ in `m´
 17.     }
 18. }
```

Once we have stored the original estimates plus their standard errors and have variables containing resampled estimates along with their resampled standard errors, we can simply request the multiple-hypothesis correction from `rwolf`, as laid out in section 3. This is implemented below.

We pass to `rwolf` the *varlist* consisting of resampled estimates. The `stdests(`*varlist*`)` option consists of bootstrap standard errors. Finally, the original correlations and standard errors from the (original) data are passed as *numlist*s in `pointestimates()` and `stderrs()`. The `graph` option requests that a graph be produced documenting the null distributions used in each test, and `noplusone` requests that the p-value be calculated following (7) rather than the standard (6).

```
. local allcorrs `c1´ `c2´ `c3´ `c4´ `c5´ `c6´ `c7´ `c8´ `c9´ `c10´
. local allserrs `s1´ `s2´ `s3´ `s4´ `s5´ `s6´ `s7´ `s8´ `s9´ `s10´
. #delimit ;
delimiter now ;
. rwolf rho1 rho2 rho3 rho4 rho5 rho6 rho7 rho8 rho9 rho10,
> nobootstraps stdests(std1 std2 std3 std4 std5 std6 std7 std8 std9 std10)
> pointestimates(`allcorrs´) stderrs(`allserrs´) graph noplusone;

Romano-Wolf step-down adjusted p-values

Outcome variables:   rho1 rho2 rho3 rho4 rho5 rho6 rho7 rho8 rho9 rho10
Number of resamples: 5000
```

Outcome Variable	Model p-value	Resample p-value	Romano-Wolf p-value
rho1	.	0.0000	0.0040
rho2	.	0.0010	0.0062
rho3	.	0.0002	0.0066
rho4	.	0.0008	0.0434
rho5	.	0.0158	0.1704
rho6	.	0.1468	0.4130
rho7	.	0.1640	0.6026
rho8	.	0.4790	0.8490
rho9	.	0.8220	0.9712
rho10	.	0.9798	0.9798

```
. #delimit cr
delimiter now cr
```

In this nonstandard implementation, no `Model p-value` is returned because `rwolf` itself does not estimate the original correlations, simply working with the estimates provided from the above code. In this case, we observe the `Resample p-value` (which is not corrected for multiple-hypothesis testing), which is the results of comparing each original estimate with the null distribution, and the `Romano-Wolf p-value`, where the multiple-hypothesis correction has been implemented as indicated in section 2.2. These results are similar to those observed in Romano and Wolf (2005a, table 4) and would result in rejecting the same hypotheses if standard cutoffs (such as $\alpha = 0.01$, $\alpha = 0.05$, or $\alpha = 0.10$) were used.

In figure 3, we present the graph of null distributions, based on (5), for each hypothesis. As implied by the multiple-hypothesis testing algorithm, the null distributions become less demanding while moving from the most significant variable (top left-hand panel) to the least significant variable (left-hand panel in the final row).

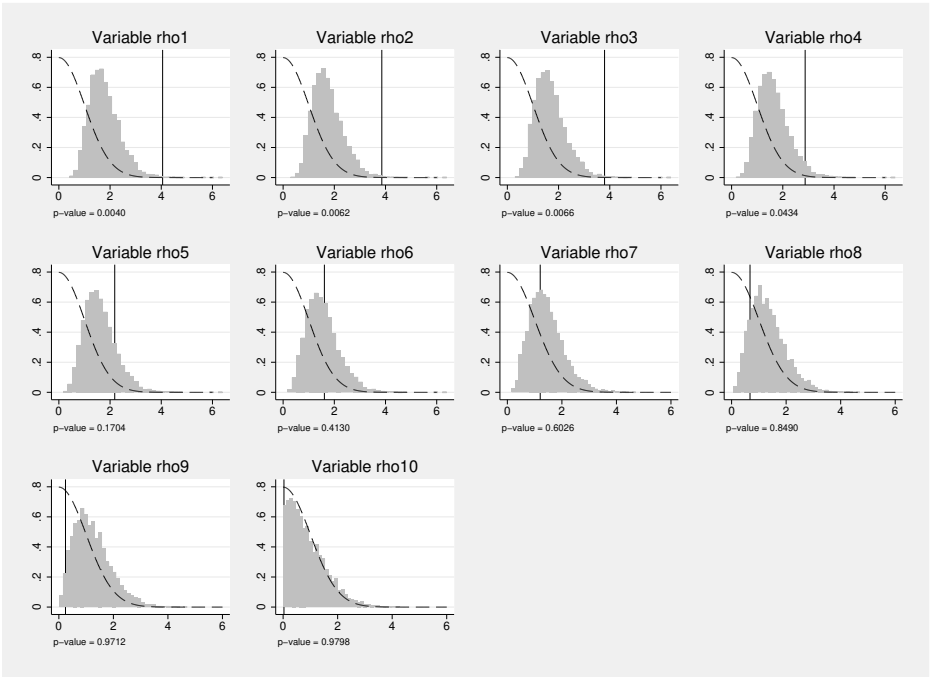


Figure 3. Null distributions and original t statistics from the SAT-deviation data.
NOTES: Each panel documents the null distributions used to calculate the Romano–Wolf adjusted p-values [following (5)] for each of the 10 outcome variables of interest. The histogram in each panel plots the stepdown resampled null distribution, the dashed line represents the theoretical half-normal, and the solid vertical line represents the original t statistic corresponding to each correlation.

5 Conclusions

In this article, we described the Romano–Wolf multiple-hypothesis correction, a flexible and versatile procedure to (asymptotically) control the FWER when testing a family of hypotheses at the same time, which occurs frequently in applied work in economics, finance, and many other fields. The article documented the `rwolf` command, which returns (multiple-testing) adjusted p-values that a) do not suffer from inflated rates of type I error and b) take into account the dependence structure of test statistics via resampling. The latter feature, together with the stepwise nature of the procedure, results in improved ability to correctly reject false null hypotheses (that is, power)

compared with more traditional multiple-testing procedures, such as the Bonferroni procedure and the Holm procedure.

We documented the syntax of the command and provided several illustrative examples, with both simulated data and real data. We documented how this command can be easily used in cases where multiple dependent variables are regressed on a single independent variable (in a broad class of regression models). In this case, implementing the Romano–Wolf multiple-hypothesis correction in Stata is a one-line endeavor, because the command interacts directly with Stata's estimation commands to implement the p-value adjustment. We also documented a more complex case, where the statistics of interest are not based on regression models and where multiple independent variables are also considered.

We envision this code being used in a variety of circumstances where multiple-hypothesis testing occurs, avoiding the well-known and undesirable pitfalls of the phenomenon interchangeably called "data mining", "cherry picking", or "p-hacking".

6 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 20-4
. net install st0618     (to install program files, if available)
. net get st0618         (to install ancillary files, if available)
```

7 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.

Attanasio, O. P., C. Fernández, E. O. A. Fitzsimons, S. M. Grantham-McGregor, C. Meghir, and M. Rubio-Codina. 2014. Using the infrastructure of a conditional cash transfer program to deliver a scalable integrated early child development program in Colombia: Cluster randomized controlled trial. *British Medical Journal* 349: g5785. https://doi.org/10.1136/bmj.g5785.

Benjamini, Y., and Y. Hochberg. 1995. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society, Series B* 57: 289–300. https://doi.org/10.1111/j.2517-6161.1995.tb02031.x.

Bonferroni, C. E. 1935. Il calcolo delle assicurazioni su gruppi di teste. In *Studi in Onore del Professore Salvatore Ortu Carboni*, 13–60. Rome: Tipografia del Senato.

Bowley, A. L. 1928. The standard deviation of the correlation coefficient. *Journal of the American Statistical Association* 23: 31–34. https://doi.org/10.2307/2277400.

Clarke, D. 2016. rwolf: Stata module to calculate Romano–Wolf stepdown p-values for multiple hypothesis testing. Statistical Software Components S458276, Department of Economics, Boston College. https://ideas.repec.org/c/boc/bocode/s458276.html.

Davison, A. C., and D. V. Hinkley. 1997. *Bootstrap Methods and Their Application*. Cambridge: Cambridge University Press.

DiCiccio, C. J., and J. P. Romano. 2017. Robust permutation tests for correlation and regression coefficients. *Journal of the American Statistical Association* 112: 1211–1220. https://doi.org/10.1080/01621459.2016.1202117.

Efron, B. 1979. Bootstrap methods: Another look at the jackknife. *Annals of Statistics* 7: 1–26. https://doi.org/10.1214/aos/1176344552.

Gertler, P., J. Heckman, R. Pinto, A. Zanolini, C. Vermeersch, S. Walker, S. M. Chang, and S. Grantham-McGregor. 2014. Labor market returns to an early childhood stimulation intervention in Jamaica. *Science* 344: 998–1001. https://doi.org/10.1126/science.1251178.

Gould, W. 1995. FAQ: Stata 6: Simulating multivariate normal observations. https://www.stata.com/support/faqs/statistics/multivariate-normal-observations/.

Holm, S. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics* 6: 65–70.

Jones, D., D. Molitor, and J. Reif. 2019. What do workplace wellness programs do? Evidence from the Illinois Workplace Wellness Study. *Quarterly Journal of Economics* 134: 1747–1791. https://doi.org/10.1093/qje/qjz023.

Lehmann, E. L., and J. P. Romano. 2005a. *Testing Statistical Hypotheses*. 3rd ed. New York: Springer.

———. 2005b. Generalizations of the familywise error rate. *Annals of Statistics* 33: 1138–1154. https://doi.org/10.1214/009053605000000084.

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.

Liu, L. Y., A. J. Patton, and K. Sheppard. 2015. Does anything beat 5-minute RV? A comparison of realized measures across multiple asset classes. *Journal of Econometrics* 187: 293–311. https://doi.org/10.1016/j.jeconom.2015.02.008.

Newson, R. B. 2010. Frequentist q-values for multiple-test procedures. *Stata Journal* 10: 568–584. https://doi.org/10.1177/1536867X1101000403.

Newson, R. B., and the ALSPAC Study Team. 2003. Multiple-test procedures and smile plots. *Stata Journal* 3: 109–132. https://doi.org/10.1177/1536867X0300300202.

Reif, J. 2017. wyoung: Stata module to perform multiple testing corrections. Statistical Software Components S458440, Department of Economics, Boston College. https://ideas.repec.org/c/boc/bocode/s458440.html.

Romano, J. P., A. M. Shaikh, and M. Wolf. 2010. Hypothesis testing in econometrics. *Annual Review of Economics* 2: 75–104. https://doi.org/10.1146/annurev.economics.102308.124342.

Romano, J. P., and M. Wolf. 2005a. Exact and approximate stepdown methods for multiple hypothesis testing. *Journal of the American Statistical Association* 100: 94–108. https://doi.org/10.1198/016214504000000539.

———. 2005b. Stepwise multiple testing as formalized data snooping. *Econometrica* 73: 1237–1282. https://doi.org/10.1111/j.1468-0262.2005.00615.x.

———. 2016. Efficient computation of adjusted p-values for resampling-based stepdown multiple testing. *Statistics & Probability Letters* 113: 38–40. https://doi.org/10.1016/j.spl.2016.02.012.

Westfall, P. H., and S. S. Young. 1993. *Resampling-Based Multiple Testing: Examples and Methods for p-Value Adjustment*. New York: Wiley.

Zar, J. H. 2010. *Biostatistical Analysis*. 5th ed. New York: Pearson.

About the authors

Damian Clarke is an associate professor of economics at Universidad de Chile.

Joseph P. Romano is a professor of statistics and economics at Stanford University.

Michael Wolf is a professor of economics at the University of Zurich.

A Westfall and Young's "Free Stepdown Resampling Method"

For background related to the multiple-hypothesis testing procedures discussed in section 2 of this article, we describe the stepdown resampling procedure of Westfall and Young (1993) here. This procedure is described as algorithm 2.8 in Westfall and Young (1993, 66–67), and we largely follow their notation but adapt it to fit the notation laid out in section 2 of this article.

This procedure begins with S multiple hypotheses, each associated with its own (unadjusted) p-value. These p-values are labeled such that $p_1 \leq p_2 \leq \cdots \leq p_S$. It then proceeds as described below.

1. Begin with a counter, $\text{COUNT}_s = 0$ for each $s = 1, \ldots, S$.

2. Using a bootstrap sample, generate a vector of analogous p-values, $(p_1^*, p_2^*, \ldots, p_S^*)$. These will not necessarily follow the same ordering as the original p-values.

3. Define the successive minimums:

$$\begin{aligned} q^*_S &= p^*_S \\ q^*_{S-1} &= \min(q^*_S, p^*_{S-1}) \\ q^*_{S-2} &= \min(q^*_{S-1}, p^*_{S-2}) \\ &\vdots \\ q^*_1 &= \min(q^*_2, p^*_1) \end{aligned}$$

4. If $q^*_s \leq p_s$, then increment COUNT_s by one unit.

5. Repeat steps 2–4 N times, and compute $\widetilde{p}_s = \text{COUNT}_s/N$.

6. Enforce monotonicity using successive maximization:

$$\begin{aligned} p^{\text{adj}}_{\text{WY},1} &= \widetilde{p}_1 \\ p^{\text{adj}}_{\text{WY},2} &= \max(p^{\text{adj}}_{\text{WY},1}, \widetilde{p}_2) \\ &\vdots \\ p^{\text{adj}}_{\text{WY},S} &= \max(p^{\text{adj}}_{\text{WY},S-1}, \widetilde{p}_S) \end{aligned}$$

The adjusted p-values $(p^{\text{adj}}_{\text{WY},1}, p^{\text{adj}}_{\text{WY},2}, \ldots, p^{\text{adj}}_{\text{WY},S})$ in the vector are the p-values corrected for multiple-hypothesis testing, where subscript WY refers to the Westfall–Young procedure. These p-values provide strong control of the FWER under the assumption of subset pivotality.

B Regression-based examples

Below, we replicate the example from section 4.2, except that here we estimate the regressions rather than calculating correlations directly (see, for example, line 2 of the first loop, and line 8 of the final principal loop). Given that t statistics calculated from each regression are identical to those calculated using the estimate of the correlation and its standard error in (9), the Studentization can be similarly performed by regression. This is documented below, where, apart from the regressions themselves, all other details follow those described in section 4.2.

```
. use "satgenerated", clear
. set seed 13032019
. local var1 satdev salary black satdev satdev ratio ratio satdev salary ratio
. local var2 black crime crime ratio crime crime black salary black salary
. tokenize `var2´
. local i=1
. foreach var of varlist `var1´ {
  2.     quietly reg `var´ ``i´´
  3.     local c``i´´=_b[``i´´]
```

```
  4.     local s`i´=_se[``i´´]
  5.     local ++i
  6. }
. foreach num of numlist 1/10 {
  2.     quietly generate rho`num´=.
  3.     quietly generate std`num´=.
  4. }
. local N=5000
. set obs `N´
number of observations (_N) was 49, now 5,000
. forvalues n=1/`N´ {
  2.     preserve
  3.     quietly keep if sat!=.
  4.     bsample
  5.     tokenize `var2´
  6.     local xvar=1
  7.     foreach var of varlist `var1´ {
  8.         quietly reg `var´ ``xvar´´
  9.         local bta`xvar´ = _b[``xvar´´]
 10.         local std`xvar´ =_se[``xvar´´]
 11.         local ++xvar
 12.     }
 13.     restore
 14.     foreach num of numlist 1/10 {
 15.         quietly replace rho`num´=`bta`num´´ in `n´
 16.         quietly replace std`num´=`std`num´´ in `n´
 17.     }
 18. }
. local allcorrs `c1´ `c2´ `c3´ `c4´ `c5´ `c6´ `c7´ `c8´ `c9´ `c10´
. local allserrs `s1´ `s2´ `s3´ `s4´ `s5´ `s6´ `s7´ `s8´ `s9´ `s10´
. #delimit ;
delimiter now ;
. rwolf rho1 rho2 rho3 rho4 rho5 rho6 rho7 rho8 rho9 rho10,
> nobootstraps stdests(std1 std2 std3 std4 std5 std6 std7 std8 std9 std10)
> pointestimates(`allcorrs´) stderrs(`allserrs´) graph noplusone varlabels;

Romano-Wolf step-down adjusted p-values

Outcome variables:   rho1 rho2 rho3 rho4 rho5 rho6 rho7 rho8 rho9 rho10
Number of resamples: 5000
```

Outcome Variable	Model p-value	Resample p-value	Romano-Wolf p-value
rho1	.	0.0000	0.0046
rho2	.	0.0010	0.0074
rho3	.	0.0016	0.0084
rho4	.	0.0054	0.0470
rho5	.	0.0268	0.1740
rho6	.	0.1428	0.3928
rho7	.	0.1350	0.5848
rho8	.	0.4844	0.8480
rho9	.	0.8242	0.9726
rho10	.	0.9788	0.9788

```
. #delimit cr
delimiter now cr
```