I Z A Institute
of Labor Economics
Initiated by Deutsche Post Foundation

DISCUSSION PAPER SERIES

IZA DP No. 12845

The Romano-Wolf Multiple Hypothesis Correction in Stata

Damian Clarke
Joseph P. Romano
Michael Wolf

DECEMBER 2019

I Z A Institute
of Labor Economics

Initiated by Deutsche Post Foundation

DISCUSSION PAPER SERIES

IZA DP No. 12845

The Romano-Wolf Multiple Hypothesis Correction in Stata

Damian Clarke
University of Santiago de Chile and IZA

Joseph P. Romano
Stanford University

Michael Wolf
University of Zurich

DECEMBER 2019

ISSN: 2365-9793

IZA – Institute of Labor Economics

Schaumburg-Lippe-Straße 5–9
53113 Bonn, Germany

Phone: +49-228-3894-0
Email: publications@iza.org

www.iza.org

ABSTRACT

The Romano-Wolf Multiple Hypothesis Correction in Stata

When considering multiple hypothesis tests simultaneously, standard statistical techniques will lead to over-rejection of null hypotheses unless the multiplicity of the testing framework is explicitly considered. In this paper we discuss the Romano-Wolf multiple hypothesis correction, and document its implementation in Stata. The Romano-Wolf correction (asymptotically) controls the familywise error rate (FWER), that is, the probability of rejecting at least one true null hypothesis in 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 Stata command rwolf that implements this correction, and provide a number of examples based on a wide range of models. We document and discuss the performance gains from using rwolf over other multiple correction procedures that control the FWER.

JEL Classification: C12, C15, C63, C87

Keywords: bootstrap, familywise error rate, multiple hypothesis testing, permutation methods, rwolf, step-down procedure

Corresponding author:
Damian Clarke
Universidad de Santiago de Chile
Av. Libertador Bernardo O'Higgins 3363
Estación Central
Santiago de Chile
Chile

E-mail: damian.clarke@usach.cl

1 Introduction

Advances in computational power and statistical programming languages such as Stata mean that frequently, the testing of additional hypotheses in an analysis is virtually costless in terms of running time. Although the computational costs of running an additional analysis are negligible, the costs of collecting new data with which to test a hypothesis are often considerable. This generally 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, in frequentist hypothesis testing, as the number of hypotheses considered in a given circumstance 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 paper 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 in the family of hypotheses under test.[1] The procedure is noteworthy given that in addition to controlling the FWER, it also offers considerably more power compared to earlier multiple hypothesis correction procedures such as Holm (1979); 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 one 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 paper; a more detailed overview can be found in Romano et al. (2010).

In this paper 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 (FDR), defined as the expected proportion of true null hypotheses rejected among all hypotheses rejected. Details of a number of such procedures, such as Benjamini and Hochberg (1995)'s 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); Newson (2010). In general FDR techniques are less conservative than FWER techniques, and are particularly common in genetic or bio-chemical applications where a huge amount of null hypotheses are considered. An illustrative applied discussion is provided in Anderson (2008).

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 paper, we will equate "control of the FWER" with "asymptotic control of the FWER".

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 et al. 2015), early childhood interventions (Gertler et al. 2014), and the evaluation of conditional cash transfers (Attanasio et al. 2014) to name but a few. In line with the frequency of use of this procedure, this paper provides a program, `rwolf`, to allow for its implementation simply in Stata.[2] Along with the theory underlying this program, we provide here a number of demonstrations chosen to illustrate the broad range of situations to which the Romano-Wolf multiple hypothesis correction, and the `rwolf` ado, is suited.

In what remains of this paper, we first provide a very 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 a number of examples, both based on simulated as well as real data sets in Section 4. We briefly 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 this 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:

$$\Pr_{\theta^0}\{\text{reject } H | H \text{ is true}\}. \tag{1}$$

The test could be a two-sided test, such as $H : \theta = \theta^0$ versus $H' : \theta \neq \theta^0$, 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 Equation (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, which fulfills (at least asymptotically):

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

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'.

2. An earlier version of this program is available as Clarke (2016).
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 $\theta^0 = 0$.

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 Equation (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 familywise error rate (FWER) is then defined as:

$$\text{FWER} := \Pr\{\text{Reject at least one } H_s \text{ with } s \in I\}. \tag{3}$$

To account for multiple hypothesis testing, we seek to control the FWER at a given rate of no greater than α. Note that by definition, if all the null hypotheses are false, the FWER is equal to zero.

A traditional solution has been to implement the Bonferroni correction. The Bonferroni procedure 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, p. 350)); however, it has often low power to detect false null hypotheses, particularly when the number of hypotheses is large. Indeed, a procedure that has greater power, while still maintaining control of the FWER was described in Holm (1979). This is a "step-down" procedure which 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 $p_{(1)} \leq \alpha/S$, 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, $H_{(s)} > \alpha/(S-s+1)$, and then stops. (If the 'stopping criterion' is never met, the procedure rejects all S hypotheses.) Obviously, the Holm procedure rejects all hypotheses rejected by the Bonferroni procedure but potentially also rejects some further hypotheses in addition.

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. Clearly, 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.

4. This is often illustrated with a simple case assuming (i) independence of individual p-values, p_s, and (ii) all S null hypotheses being true, in which case the probability of falsely rejecting at least one true null hypothesis is equal to $1-(1-\alpha)^S$.
5. Strong control of the FWER means that the FWER will be no greater than α regardless which of the null hypotheses are true or not. This contrasts with *weak* control of the FWER, which refers to the case where there is a guarantee that the FWER does not exceed α *only* in the case that all null hypotheses are true. Unless otherwise noted, in the remainder of the paper, control of the FWER is equated with strong control.

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 1 of this paper), assumes a certain *subset pivotality* condition, which is used in establishing FWER control. This assumption can be violated in certain applications and is thus undesirable.[6] The procedure is available for Stata, as provided in Reif (2017), with additional discussion in Jones et al. (2019).

More recently, the Romano-Wolf multiple hypothesis correction has been proposed, as described in Romano and Wolf (2005b,a) as well as in the subsection below. This procedure also uses resampling and step-down 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 to 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 (2005b)'s "Studentized StepM Procedure" (Algorithms 4.1–4.2). Here we describe the algorithms implemented, before turning to the precise syntax in the following subsection.

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. The Romano Wolf procedure can also be implemented using permutation methods rather than bootstrap methods (Romano and Wolf 2005a); however, note that permutation tests of regression coefficients can result in rates of Type I error which exceed the nominal size, and so these methods are likely not ideal for such applications (DiCiccio

6. If subset pivotality does not hold, the Westfall-Young procedure only offers weak control of the FWER.

and Romano 2017). Nevertheless, the `rwolf` command can certainly 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 are all two-sided of the type $H_s' : \theta_s \neq 0$.[8] A Studentized test statistic for H_s is given by[9]

$$t_s := \frac{\widehat{\theta}_s}{\widehat{\sigma}_s}. \tag{4}$$

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

$$t_s^{*,m} := \frac{\widehat{\theta}_s^{*,m} - \widehat{\theta}_s}{\widehat{\sigma}_s^{*,m}}. \tag{5}$$

Note that these statistics $t_s^{*,m}$ are centered around zero 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

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 pre-processing 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 non-zero value, $H_s : \theta_s = \theta_s^0$, Equation (4) should be changed to
$$t_s := \frac{\widehat{\theta}_s - \theta_s^0}{\widehat{\sigma}_s}.$$
10. It is important to point out that Equation (5) is also valid if the test is for a nonzero value, $H_s : \theta = \theta_s^0$, and therefore needs *not* to be changed in such a case.

test statistic (hereafter $t_{(1)}$), and $H_{(S)}$ refers to that with the smallest (labelled $t_{(S)}$). In what follows, we denote by $\max_{t,j}^{*,m}$ the largest value of the vector $\left(t_{(j)}^{*,m}, \ldots, t_{(S)}^{*,m}\right)$:

$$\max_{t,j}^{*,m} := \max\left\{t_{(j)}^{*,m}, \ldots, t_{(S)}^{*,m}\right\} \qquad \text{for } j = 1, \ldots, S \text{ and } m = 1, \ldots, M, \tag{6}$$

and, for a given j, we denote by $\hat{c}(1-\alpha, j)$ 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, $\max_{t,1}^{*,m}$ denotes the maximum value of $(t_{(1)}^{*}, t_{(2)}^{*}, t_{(3)}^{*}, t_{(4)}^{*})$ whereas $\max_{t,2}^{*,m}$ denotes the maximum value of the subvector $(t_{(2)}^{*}, t_{(3)}^{*}, t_{(4)}^{*})$, that is the vector of test statistics corresponding to the the three 'least significant' hypotheses only; and so on. At last, we simply have $\max t_{(4)}^{*,m} = t_{(4)}^{*,m}$. An important consequence is that $\hat{c}(1-\alpha, j)$ is weakly decreasing in j, that is, $\hat{c}(1-\alpha, j) \geq \hat{c}(1-\alpha, j+1)$, for $j = 1, \ldots, S-1$.

Based on the above, the principal step-down multiple testing procedure at a significance level of α (based on Algorithm 3.1 from Romano and Wolf (2016)) can be summarized as follows.

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

As was the case with the procedure of Holm (1979), this correction is a step-down procedure: $\hat{c}(1-\alpha, j)$ 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 $\max_{t,j}^{*,m}$ 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 defined 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, p. 158):

$$p := \frac{\#\{t^{*,m} \geq t\} + 1}{M+1}; \tag{7}$$

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 $\max_{t,j}^{*,m}$, following Algorithm 4.2 of Romano and Wolf (2016).

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

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 Equation (7) (or Equation (8), if `noplusone` is specified) for comparison.

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

where *depvars* refers to the multiple outcome variables which are being considered. There are two ways for this command to be used. First, either `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 pre-computed 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` and `pointestimates(`*numlist*`)`, `stderrs(`*numlist*`)` and `stdests(`*varlist*`)` should be indicated.

11. Another (somewhat less conservative) common definition is:
$$p := \frac{\#\{t^{*,m} \geq t\}}{M}. \tag{8}$$
Either option can be implemented in the `rwolf` command, as described in the following section.

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 modes of use is provided in Section 4 of this paper.

3.1 Options

Standard Options

`indepvar(`*varlist*`)` Indicates the independent (treatment) variable which is included in multiple hypotheses tests. This will typically be a single independent variable, however it is possible to indicate various independent (treatment) variables which are included in the same model, and 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 are performed (that is, the baseline models). Standard regression-based estimation commands permitted by Stata can be specified, including `logit`, `probit`, `ivregress`, `regress` and `areg`. If IV estimation is desired, `ivregress` must be indicated, rather than `ivreg2` or alternative IV regression models. This option must be specified, unless the `nobootstraps` option is indicated (see below).

`controls(`*varlist*`)` Indicates additional controls which 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. If specified, a single scalar value should be indicated for each of the multiple hypotheses tested, and these should be listed in the same order that variables are listed as *depvars* in the command syntax. In the case that multiple *indepvars* are specified, null parameters should be specified grouped first by *indepvars* 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. If this option is not used, it is assumed that each null hypothesis is that the parameter is equal to 0.

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

`reps(#)` Perform # bootstrap replication; default is `reps(100)`. Where possible, a

much larger number of replications should be indicated 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 estimated.

`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.

`onesided(`*string*`)` Indicates that p-values based on one-sided tests should be calculated. Unless specified, p-values based on two-sided tests are provided, corresponding to the null that each parameter is equal to 0 (or the values indicated in `nulls()`). In `onesided(`*string*`)` *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, variables y1_bl, y2_bl and y3_bl should be created with baseline values, and `bl(_bl)` should be specified.

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

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

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

`noplusone` Calculate the Resampled and Romano-Wolf adjusted p-values without adding one to the numerator and denominator (that is, according to Equation (8) rather than Equation (7)).

`nodots` Suppress replication dots.

`holm` Along with standard output, additionally provide p-values corresponding to the Holm (1979) correction. These will be 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. These graph the distribution of $\max_{t,j}^{*,m}$ from Equation (6) for each j in $1, \ldots, S$. Examples of such a graph are provided in Section 4.

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

`*` Any additional options which 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 bootstrap replications do not need to be estimated 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*`)`. This option 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. This option may not be specified if `indepvar()` and `method()` are specified. For all standard implementations, `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. This option may not be specified if `indepvar()` and `method()` are specified. For all standard implementations, `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. This option may not be specified if `indepvar()` and `method()` are specified. For all standard implementations, `indepvar()` and `method()` should be preferred.

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

3.2 Returned Objects

`rwolf` is an eclass program, and returns a number of 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. In the case that multiple *independent* variables are considered, a scalar for each p-value corresponding to each independent variable–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. In the case that the `holm` option is indicated, p-values according to Holm (1979) will be returned in column 4 of this matrix. Once again, in the case that 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 ten 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 is simulated as:

$$y_i^s = \beta_0^s + \beta_1^s Treat_i + \varepsilon_i^s \quad \text{for } s = 1, \ldots, 10, \tag{9}$$

for observation i. For each outcome i, the ten stochastic error terms ε_i^s are drawn from a multi-variate normal (MVN) distribution, following:

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

with $\rho \geq 0$, and the independent variable of interest, $Treat_i$ is generated as $Treat_i := \mathbb{1}_{\{U_i > 0.5\}}$, where $\mathbb{1}$ denotes the indicator function and $U_i \sim U(0, 1)$. This is a highly stylized setting, but 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 $\rho = 0.25$. In this simulation, we follow the data generating process described above where first we simulate $Treat_i$ (as `treat`), and then generate ε_i as a draw from a 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 obs 100
number of observations (_N) was 0, now 100
. gen treat  = runiform()>0.5
. foreach num of numlist 1(1)10 {
  2.     gen c`num´     = rnormal()
  3. }
. local r=0.25
. #delimit ;
delimiter now ;
. mat 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
. mat 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.     gen y`num´ = 1 + 0*treat + epsilon`num´
  5. }
.
. 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

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

12. Further discussion of this is provided in Gould (undated). 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 $c \sim \mathcal{N}(0, I)$ can be transformed following $\varepsilon := \mu + Lc$, resulting in a vector $\varepsilon \sim \mathcal{N}(\mu, LL')$. Thus if $\Sigma = LL'$ is the desired covariance matrix, the Cholesky decomposition can be taken to give $L = cholesky(\Sigma)$, and the resulting L used to generate the correlated random draws with desired mean μ and covariance matrix Σ.

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

Once we have simulated the $N \times S$ matrix Y in the above code, we apply the Romano-Wolf step-down 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 arguments, will be performed by resampling observational units with replacement. By specifying the `nodots` and `holm` arguments, we request that no dots be displayed on the Stata output to indicate the degree to which the resample procedure has advanced, and also for `rwolf` to return p-values corrected using Holm's procedure (which will be saved in the `e(return) list`).

The command returns a list of p-values associated with each of the multiple outcomes. The column "Model p-values" 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-values" lists the resampling-based p-values as per Equation (7)), 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, a number 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 final column displays the Romano-Wolf adjusted p-values, where we note that now no null hypothesis is rejected (even at $\alpha = 0.10$).

This simulated example above provides one example of a multiple hypothesis correction based upon a known data generating process (DGP). In order 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 Table 1.[13] Here we consider a series of models where each of the β_1^s terms are equal to 0 (presented in Panel A), a series where half of the β_1^s terms are equal to zero, and the other half are equal to 0.5 (presented in panel B),[14] and finally a series where each of the β_1^s terms are equal to 0.5 (panel C). Across columns, we vary the degree of correlation between outcomes, from $\rho = 0$ in the

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

first two columns, to $\rho = 0.75$ in the final two columns. In each case (where relevant) we report the empirical FWER and the rate of hypotheses correctly rejected. The nominal levels for the FWER are set at $\alpha = 0.05$ and $\alpha = 0.10$, respectively.

Table 1: Simulated Error and Rejection Rates

	$\rho = 0$		$\rho = 0.25$		$\rho = 0.50$		$\rho = 0.75$	
	5%	10%	5%	10%	5%	10%	5%	10%
Panel A: All $\beta_1 = 0$								
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 $\beta_1 = 0.5$								
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
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 C: All $\beta_1 = 0.5$								
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

Error rates are documented for individual tests, and familywise over all outcomes. Romano-Wolf error rates are displayed at the end of each panel, compared with uncorrected error rates based on bootstrap inference, and error rates based on Holm's procedure. When considering rates of rejection of (false) null hypotheses, "R-W" refers to Romano-Wolf. 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 $M = 5,000$.

Table 1 is in the spirit of Tables 1–3 in Romano and Wolf (2005a). Here we briefly discuss the performance of the `rwolf` command, and note a number of important features of the Romano-Wolf hypothesis correction. In particular, we always present the performance criteria for three testing procedures: the ones corresponding to the naïve case, where no correction for multiple hypothesis testing is made, the ones corresponding to Holm's procedure, and the ones corresponding to the Romano-Wolf procedure. In panel A, all values for $\beta_1^s = 0$, and as such we need not present the rate of hypotheses correctly rejected. In examining empirical FWERs, it is clear in the uncorrected case the FWER greatly exceeds 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. When undertaking the Holm correction, we now 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 in the case of the Romano-Wolf procedure. Indeed, in each case the empirical FWER is very close to desired nominal rate of 0.05 or 0.10, respectively.

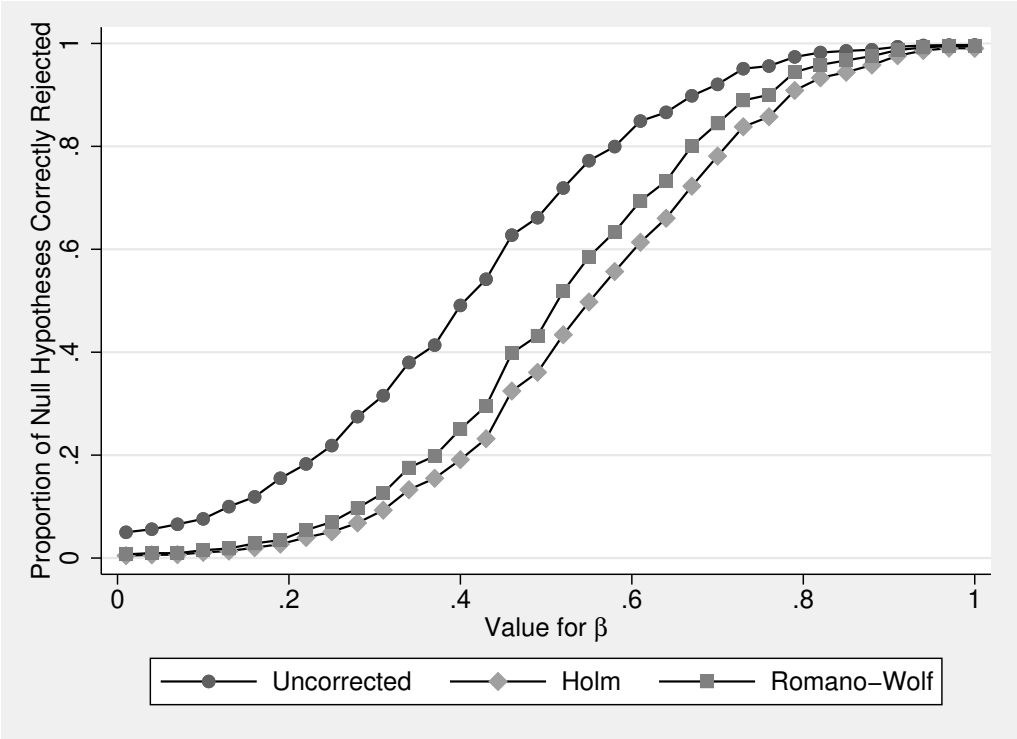
In panels B and C, we can additionally examine the relative power of the Holm and Romano-Wolf procedures for rejecting false null hypotheses. In this case once again we observe in panel B that the FWER without multiple hypothesis correction substantially exceeds desired error rates, but is successfully controlled at no more than α with both the Holm and Romano-Wolf procedures. And in turning to the final three rows of panel B we observe the relative power of each procedure. The naïve case of course allows us to reject a large proportion of false null hypotheses (at the cost of the FWER). 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 B, we observe that we reject 56% of false null hypotheses using the Romano-Wolf procedure, versus only 43% using Holm's procedure, a 30% improvement in rejection rate. A similar pattern is observed in panel C where all null hypotheses are false. Initially when the correlation between outcomes is 0, Holm and Romano-Wolf's procedures have similar power, but to the degree that ρ increases, the Romano-Wolf procedure becomes relatively more powerful.

In Table 1 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. In Figure 1 we consider how these rates of rejection vary with β_1^s values. In this figure we consider a particular example, namely that corresponding to the panel C (where all $\beta_1^s \neq 0$), and where $\rho = 0.5$. In this figure 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 Equation (9). 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 in the case that 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 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

15. As discussed in the Procedures section, 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.

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%.

Figure 1: Comparative power to reject false null hypotheses.



Notes to Figure 1: 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 Equation (9), 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.

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 OLS 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 a number of such extensions, we turn to an alternative example below, based

on a simple instrumental variables regression example. This example is a standard Stata 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 2SLS 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`. Here 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, as per the example. In this case we use 10,000 bootstrap replications (bootstrapping on observational units), and request a graph of the null distributions used in testing to 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 zero. 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.

```
. 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

Dependent 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 refers to the coefficient on `hsngval` in the second stage of the IV regression. For the sake of illustration, we multiply two outcomes by -1, such that the sign on $\hat{\beta}_1$ estimated in each IV regression is greater than zero. Along with the syntax described previously, the implementation of one-sided tests requires the use of the argument `onesided()`. If `onesided(negative)` is specified,

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

the null will be $H_0 : \beta_1 \leq 0$, that is, negative values will provide more support for the null, whereas 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.

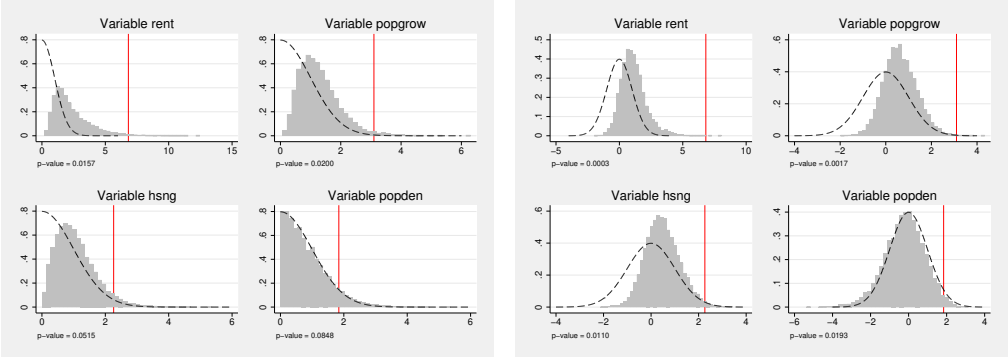
```
. replace popden=-popden
(50 real changes made)
. replace hsng  =-hsng
(50 real changes made)
. rwolf rent popden popgrow hsng, indepvar(hsngval) method(ivregress) /*
>  */ iv(faminc i.region) reps(10000) graph onesided(negative) nodots  /*
>  */  controls(pcturban)
Bootstrap replications (10000).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

Romano-Wolf step-down adjusted p-values

Dependent 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	0.0003	0.0003
popden	0.0327	0.0193	0.0193
popgrow	0.0010	0.0012	0.0017
hsng	0.0118	0.0060	0.0110

In each of these examples, we have specified the `graph` option, which means that we request as output the null distributions used to calculate the p-value in each case. Examining these distributions is useful insofar as it 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 case of 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 that maximum coefficient estimated across each outcome. These null distributions become increasingly less demanding in the top right and bottom left panels as 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, however 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 which is 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.

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



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

Notes to Figure 2: 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, however now based on the one-tailed test where each null is that $\beta_1 \leq 0$.

4.2 A Non-Standard Studentized Example

Each of the previous examples 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 set-ups, if the user wishes to pass the command the already bootstrapped (or permuted) statistics and standard errors which have been calculated from the underlying models of interest. In what remains of this section, we document this flexibility, using a bootstrap approach where our statistics of interest are a series of correlations.

To do so, 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 "Non-standard" example, it is only non-standard in the way it interacts with the syntax of `rwolf`,

given that the multiple tests are based on a number of 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 a number of other state-level measures for the 48 mainland US states plus Hawaii. These data — consisting simply of state-level measures of a number of variables of interest — are provided in Table 6.4 (p. 197) of Westfall and Young (1993) as well as in the replication materials for this paper. The precise variables considered are the "SATdev" (generated as 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 which 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. In order to construct null distributions following Equations (5)–(6), 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 order the hypotheses in terms of their relative significance. It additionally requires the results from M bootstrap replicates, where in each case 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]:

$$se_{\hat{\rho}} := \sqrt{\frac{1-\hat{\rho}^2}{N-2}}. \tag{10}$$

To generate the various components that `rwolf` uses to implement the multiple hypothesis correction we first open the data and define the pairs to be tested one-by-one in locals `var1` and `var2`. The idea in these locals is that we wish to calculate the correlation between the first two variables in each local, the second two variables in each local, and so forth, until reaching the tenth two variables in each local. We calculate these correlations one by one in the `foreach` loop in the below code. To do so we loop over elements of the local `var1`, and take 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 a series of ten empty variables `rho1`–`rho10` and `std1`–`std10` which will later be populated by resample-based correlation estimates and their standard errors.

```
. use "SATgenerated", clear
```

17. In Appendix 2 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.

```
. 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.     qui 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.     qui gen rho`num´=.
  3.     qui gen std`num´=.
  4. }
.
```

A bootstrap procedure is then implemented, based on 5,000 resamples. We initially expand the data-set 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 towards the end of each iteration, as 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.     qui keep if sat!=.
  4.     bsample
  5.     tokenize `var2´
  6.     local s=1
  7.     foreach var of varlist `var1´ {
  8.         qui 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.         qui replace rho`s´=`cor`s´´ in `m´
 16.         qui replace std`s´=`std`s´´ in `m´
 17.     }
 18. }
```

Once we have stored the original estimates and 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 the command the `varlist` consisting of resampled estimates, and then in the option `stdests()` the `varlist` consists of bootstrap standard errors. Finally, the original correlations and standard errors from the (original) data are passed as `numlists` in `pointestimates()` and `stderrs()`. The `graph` option requests for a graph to be produced documenting the null distributions used in each test, and `noplusone` suggests that the p-value should be calculated following Equation (8) rather than following the 'standard' Equation (7).

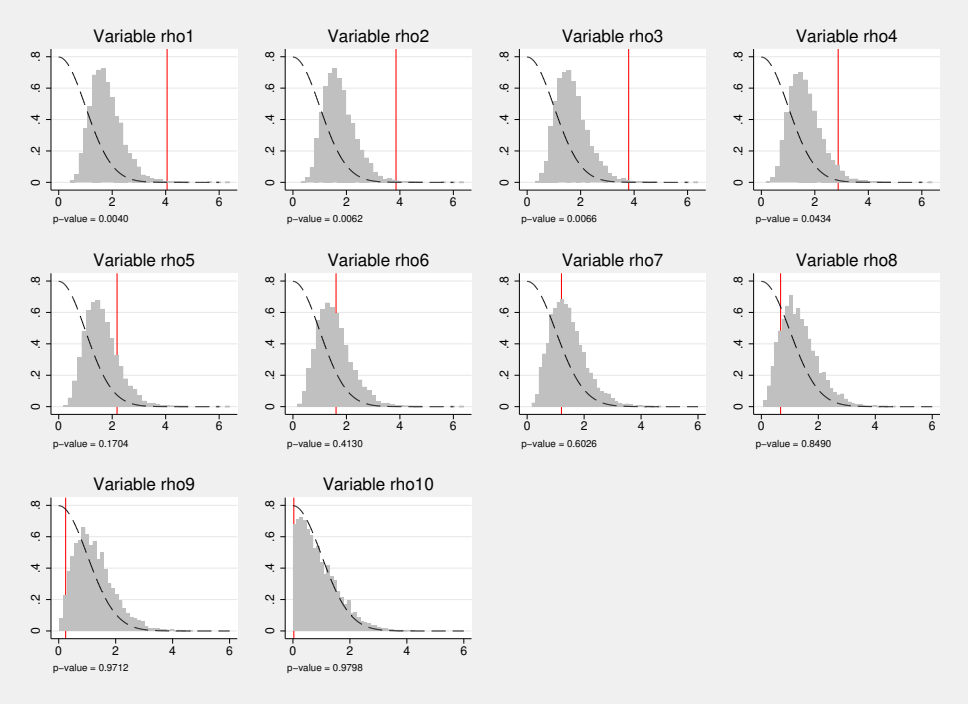
```
. local allcorrs `c1´ `c2´ `c3´ `c4´ `c5´ `c6´ `c7´ `c8´ `c9´ `c10´
. local allserrs `s1´ `s2´ `s3´ `s4´ `s5´ `s6´ `s7´ `s8´ `s9´ `s10´
.
. #delimit ;
delimiter now ;
. rwolf rho1 rho2 rho3 rho4 rho5 rho6 rho7 rho8 rho9 rho10,
> nobootstraps stdests(std1 std2 std3 std4 std5 std6 std7 std8 std9 std10)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

The output of this command is provided above. In the case of this "non-standard" implementation, no "Model p-value" is returned given that `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) and consists 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 hypothesis in the case that standard cut-offs (such as $\alpha = 0.01$, $\alpha = 0.05$ or $\alpha = 0.10$) were used.

Finally, we also present the graph of null distributions, based on Equation (6), for

each hypothesis in Figure 3. As implied by the multiple hypothesis testing algorithm, we observe that the null distributions become less demanding as 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 to Figure 3: Each panel documents the null distributions used to calculate the Romano-Wolf adjusted p-values (following Equation (6)) for each of the ten outcome variables of interest. The histogram in each panel plots the step-down 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

This paper describes the Romano-Wolf multiple hypothesis correction which is a flexible and versatile procedure to (asymptotically) control the familywise error rate (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 paper documents 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 to more traditional multiple testing procedures, such as the Bonferroni procedure and the Holm procedure.

We document the syntax of the command, and provide a number of illustrative examples, both with simulated and with real data. We document how this command can be used very simply 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, as the command interacts directly with Stata's estimation commands to implement the p-value adjustment. We also document a more complex case, where the statistics of interest are not based on regression models, and where multiple independent variables are also considered. It is envisaged that this code can be used in a wide 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 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(484): 1481–1495.

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.

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 (Methodological)* 57(1): 289–300.

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

Bowley, A. L. 1928. The Standard Deviation of the Correlation Coefficient. *Journal of the American Statistical Association* 23(161): 31–34.

Clarke, D. 2016. RWOLF: Stata module to calculate Romano-Wolf stepdown p-values for multiple hypothesis testing. Statistical Software Components, Boston College Department of Economics. 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. *JASA* 112(519): 1211–1220.

Efron, B. 1979. Bootstrap methods: another look at the jackknife. *Annals of Statistics* 7: 1–26.

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(6187): 998–1001.

Gould, W. Undated. 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(2): 65–70.

Jones, D., D. Molitor, and J. Reif. 2019. What do Workplace Wellness Programs do? Evidence from the Illinois Workplace Wellness Study. *The Quarterly Journal of Economics* 134(4): 1747–1791.

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

———. 2005b. Generalizations of the Familywise Error Rate. *The Annals of Statistics* 33(3): 1138–1154.

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. *The Journal of Econometrics* 187(1): 293–311.

Newson, R., and The ALSPAC Study Team. 2003. Multiple-test procedures and smile plots. *Stata Journal* 3(2): 109–132.

Newson, R. B. 2010. Frequentist q-values for multiple-test procedures. *Stata Journal* 10(4): 568–584.

Reif, J. 2017. WYOUNG: Stata module to perform multiple testing corrections. Statistical Software Components, Boston College Department of Economics. 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(1): 75–104.

Romano, J. P., and M. Wolf. 2005a. Exact and Approximate Stepdown Methods for Multiple Hypothesis Testing. *Journal of the American Statistical Association* 100(469): 94–108.

———. 2005b. Stepwise Multiple Testing as Formalized Data Snooping. *Econometrica* 73(4): 1237–1282.

———. 2016. Efficient computation of adjusted p-values for resampling-based stepdown multiple testing. *Statistics and Probability Letters* 113: 38–40.

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

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 Santiago 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.

Appendices

1 Westfall and Young's "Free Step-Down Resampling Method"

For background related to the multiple hypothesis testing procedures discussed in Section 2 of this paper, we describe the step-down resampling procedure of Westfall and Young (1993) here. This procedure is described as Algorithm 2.8 in Westfall and Young (1993, pp. 66–67) and we largely follow their notation, however adapted to fit the notation laid out in section 2 of this paper.

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

1. Begin with a counter, $COUNT_s = 0$ 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 minima:

$$\begin{aligned} q_S^* &= p_S^* \\ q_{S-1}^* &= \min(q_S^*, p_{S-1}^*) \\ q_{S-2}^* &= \min(q_{S-1}^*, p_{S-2}^*) \\ &\vdots \\ q_1^* &= \min(q_2^*, p_1^*) \end{aligned}$$

4. If $q_s^* \leq p_s$, then increment $COUNT_s$ by 1 unit.
5. Repeat steps 2–4 N times, and compute $\tilde{p}_s = COUNT_s/N$
6. Enforce monotonicity using successive maximization:

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

The vector of adjusted p-values $(p_{WY,1}^{\text{adj}}, p_{WY,2}^{\text{adj}}, \ldots, p_{WY,S}^{\text{adj}})$ 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*.

2 Regression-Based Examples

Below we replicate the example from section 4.2, however here rather than calculating correlations directly, regressions are estimated (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 equation 10, 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.     qui 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.     qui gen rho`num´=.
  3.     qui gen 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.     qui keep if sat!=.
  4.     bsample
  5.     tokenize `var2´
  6.     local xvar=1
  7.     foreach var of varlist `var1´ {
  8.         qui 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.         qui replace rho`num´=`bta`num´´ in `n´
 16.         qui 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;

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
```