The Stata Journal (2009)
9, Number 1, pp. 70–85

A general-purpose method for two-group randomization tests

Johannes Kaiser
Laboratory for Experimental Economics
University of Bonn
Bonn, Germany
johannes.kaiser@uni-bonn.de

Michael G. Lacy
Sociology Department
Colorado State University
Fort Collins, CO
michael.lacy@colostate.edu

Abstract. We outline a novel approach to calculate exact p-levels for two-sample randomization tests. The approach closely resembles `permute` in its applications, with the main difference being that the results are approximated only if the execution time needed to calculate exact p-levels would exceed a specified maximum. We demonstrate its use by deriving p-levels for the significance of Somers' D, the coefficient of variation, the difference in means and in medians, and the difference in two multinomials.

Keywords: st0158, tsrtest, mwtest, fptest, somersdtest, cvtest, vartest, randomization tests, Monte Carlo, two-sample problem

1 Introduction

In this article, we present a command, `tsrtest`, that implements a two-group exact randomization test with a test statistic of the user's choosing. By "two-group exact randomization test", we mean a hypothesis test for which the null hypothesis is that the observed value of the test statistic is consistent with a random assignment of cases to the two groups. This is the same null hypothesis tested by `permute` (see [R] **permute**),[1] but `tsrtest` offers, as feasible, an exact p-value derived from examining the test statistic for all distinct assignments of cases to groups rather than the sample of such assignments generated by `permute`. `tsrtest` thus generalizes Kaiser's (2007) `permtest2`, which offers an exact randomization test for the difference of two means, by allowing the test statistic to be any r-class result returned by a user-written or an official Stata command. `tsrtest` has the further advantage of using a much more efficient algorithm than `permtest2`. `tsrtest` combines flexibility with algorithmic efficiency sufficient to feasibly calculate exact two-group tests for relatively large sample sizes.

2 The algorithm and its efficiency

The efficiency of `tsrtest` in implementing a randomization test rests on its generating all assignments of the N cases to two groups, without regard to order within the group,

1. Compare with the views of others (e.g., Edgington and Onghena [2007, 289]) who wish to distinguish the underlying logic of permutation and randomization tests. This distinction is not at issue here, where our goal is simply to offer a new software routine.

rather than generating all permutations of the group or response variable vector, as would be done by a conventional permutation routine, such as the one implemented in `permute` or `permtest2`. Permutation of the response or explanatory variable is algorithmically inefficient for randomization tests when the variable vector being permuted is a two-category group variable because it generates many redundant arrangements of the data, given that the order of cases within groups is irrelevant for any two-group test statistic known to the authors. The extent of redundancy is dramatic: for a sample comprising two groups of sizes n_1 and n_2, there are $(n_1+n_2)!$ permutations of the group membership vector but only $(n_1+n_2)!/(n_1!n_2!)$ distinct assignments of the N cases to the two groups; so a full permutation approach involves generating group assignments and calculating the test statistic $n_1!n_2!$ times more than necessary. For example, with $n_1 = 10$ and $n_2 = 20$, there are 3.0×10^7 unique assignments of the 30 cases into two groups, a large but tractable number for an exact solution, but there are $N! = 2.7 \times 10^{32}$ permutations of the response vector, which is clearly beyond what is possible with current computational equipment. Thus the combination-oriented approach taken here makes it feasible to obtain exact p-values in many situations in which small sample sizes make asymptotic approaches untrustworthy, but in which a full permutation approach could not work.

3 Flexibility of the current approach

Like `bootstrap` (see [R] **bootstrap**) or `permute`, the `tsrtest` command is more flexible than a special purpose command like `permtest2`, which gives only one test statistic. It also is more flexible in allowing many more exact tests than can be obtained by relying on exact options available for some official Stata commands (e.g., `median` [see [R] **ranksum**]) but not others (e.g., `ttest` [see [R] **ttest**]). `tsrtest` permits the test statistic to be anything calculated by a user-written or an official Stata command and returned as an r-class result. As noted by Manly (2006), one virtue of the randomization approach to hypothesis testing is that it frees the analyst from using a test statistic whose null distribution is known or asymptotically well-approximated, allowing one to choose or devise a test statistic that optimally measures the phenomenon of interest. The flexibility of `tsrtest` fits well with this generic desired feature of randomization tests.

4 A simple example using tsrtest

Before offering a more complete description of the functioning and syntax of `tsrtest`, we present a simple illustration comparing it to the exact analogue to the two-group t test implemented in `permtest2`. Suppose we wish to obtain a nonasymptotic p-value for a test of the null hypothesis that the difference in mean miles per gallon of domestic (U.S.) cars versus other (foreign) cars is no different than what might occur from random assignment of the domestic and foreign labels within the sample; the alternative hypothesis is that the difference in mean miles per gallon for domestic cars is larger than expected under the randomization hypothesis. Assume further that

interest lies only in more expensive cars, those costing $7,000 (USD) or more, of which there are 10 domestic and 6 foreign. Considering that miles per gallon can well be heteroskedastic and nonnormal, relying on the robustness properties of a conventional t test at this small sample size would be dubious, and so we choose to address the question by using `permtest2` or `tsrtest`. Because `permtest2` uses the Pitman–Fisher approach, with the test statistic defined as simply the difference in means between samples, we follow the same approach in using `tsrtest`.

▷ Example

A short user-written Stata program is created to calculate the test statistic, which is called by `tsrtest` similarly to how any user-written command might be called by `permute`:

```
// Program to calculate test statistic
capture program drop meandiff
program meandiff, rclass
args y group
summarize `y´ if `group´ == 0
local mean0 = r(mean)
summarize `y´ if `group´ == 1
return scalar diff = r(mean) - `mean0´
end
//
sysuse auto
set seed 123456789
keep if price > 7000
tsrtest foreign r(diff): meandiff mpg foreign
//
// Comparison to -permtest2- and a conventional -ttest-
permtest2 mpg, by(foreign) exact
ttest mpg, by(foreign)  // dubious application of ttest
```

In the preceding application, `tsrtest` gives the following results:

```
. tsrtest foreign r(diff): meandiff mpg foreign
Two-sample randomization test for theta=r(diff) of meandiff mpg foreign by foreign

Combinations:   8008 = (16 choose 6)
Assuming null=0
Observed theta: 4

Minimum time needed for exact test (h:m:s):  0:00:01
Mode: exact

progress: |.....................................|

 p=0.03072 [one-tailed test of Ho:  theta(foreign==1)<=theta(foreign==0)]
 p=0.97602 [one-tailed test of Ho:  theta(foreign==1)>=theta(foreign==0)]
 p=0.05832 [two-tailed test of Ho:  theta(foreign==1)==theta(foreign==0)]
```

The p-value of 0.03072 is identical to what `permtest2` gives; however, `tsrtest` gives a different two-tailed value because `tsrtest`, unlike `permtest2`, does not assume symmetry of the upper and lower tails in calculating two-tailed p-values, but instead separately counts values in the upper and lower tails.[2]

◁

The preceding problem required about one second (in Stata/SE 9, using Windows XP on a machine with one Intel processor running at 3.4 GHz), while `permtest2` required about six seconds. Superior performance of `tsrtest` is achieved even though `permtest2` has the advantage of implementing all the calculations in Mata, does not have to call another Stata program, and does not have the overhead of flexibility that `tsrtest` has. For this small problem, the difference between the performance of the two programs would not matter to the user, but with even a somewhat larger sample, the permutation approach of `permtest2` becomes intractable.

▷ Example

When the preceding comparison is expanded to cars costing more than \$6,000 (USD), i.e.,

```
keep if price > 6000
tsrtest foreign r(diff): meandiff mpg foreign
permtest2 mpg, by(foreign) exact
```

there are 14 domestic cars and 9 foreign cars. The run time for `tsrtest` increased to 95 seconds, a change in proportion to ${}_{23}C_9$ from ${}_{16}C_6$, but `permtest2` did not even finish within 12 hours because its permutation task increased in proportion to $23!/16! = 1.2 \times 10^9$. Were this problem increased further, say, by attempting the preceding test with the full sample of 52 domestic and 22 foreign cars, the execution time would be excessive even for `tsrtest`, and the program would revert to an approximate solution, with a p-value based on some fixed number (10,000 by default) of permutations of the values of the explanatory variable:

```
. tsrtest foreign r(diff): meandiff mpg foreign
Two-sample randomization test for theta=r(diff) of meandiff mpg foreign by foreign

Combinations:   3.64867747887e+18 = (74 choose 22)
Assuming null=0
Observed theta: 4.946

Minimum time needed for exact test (h:m:s):  1.42e+11:17:11
Reverting to Monte Carlo simulation.
Mode: simulation (10000 repetitions)

progress: |.....................................|

 p=0.00020 [one-tailed test of Ho:  theta(foreign==1)<=theta(foreign==0)]
 p=0.99970 [one-tailed test of Ho:  theta(foreign==1)>=theta(foreign==0)]
 p=0.00020 [two-tailed test of Ho:  theta(foreign==1)==theta(foreign==0)]
```

◁

2. Both programs will give the same results if sample sizes are equal. If not, p-values from `tsrtest` are preferable because they are free of the symmetry assumption.

5 Algorithm and functioning of tsrtest

When `tsrtest` performs an exact two-group randomization test, it first calculates the test statistic on the observed data, with the observed assignment of cases to the two values of the group variable. Using a Mata program, `tsrtest` then selects all the possible combinations of choosing n_1 cases out of the total N, where n_1 represents the observed number of cases in the smaller group. Taking each such combination as representing one of the possible assignments of the cases to the two values of the group variable, `tsrtest` then calls the user's command to calculate the test statistic for comparison with the test statistic computed on the observed data distribution.

Because this is a combinatorial problem in the strict sense, i.e., because the order of the cases within the group does not matter, the Mata code uses a combinatorial algorithm (Gentleman 1975). Although many combinatorial algorithms exist, Gentleman's is simple (i.e., it does not require recursion), relatively short, and requires only one call.

In the current application, one call generates the indices of each possible combination of the N cases taken n_1 at a time, which corresponds to one possible arrangement of the data in the group variable vector that has n_1 of the cases allocated to group 1 and the others to group 2. Next the Mata program calls the user's command with the current combination used to designate group membership. The difference of the current value of the test statistic relative to the null value is compared with that same difference for the test statistic computed on the observed data. If the current difference exceeds the observed difference, the count of positive differences is increased; similarly, for differences equal to or less than the observed difference, the count of positive differences is decreased. When all combinations have been generated, the combinatorial algorithm terminates, and these counts are used to report a p-value. Users interested in more detail should examine the Mata code, available in text form in the `tsrtest` command.

As implied above, even this relatively efficient approach to an exact randomization test can exceed what can be completed in a reasonable amount of time. For this reason, before attempting the full combinatorial approach to a randomization test, `tsrtest` runs the user's command on the observed data 200 times, and it uses those results to estimate the total execution time for the exact solution. If this estimated time exceeds the default or user-supplied maximum time and if the user has not selected an option to force an exact solution, `tsrtest` reverts to a simulation approach, noting that fact for the user. Here it performs a permutation simulation of the randomization p-value by randomly shuffling the observed group assignment variable vector for some number of repetitions by using the Mata `jumble()` function (see [M-5] **sort( )**); for each shuffling, `tsrtest` calls the user's command and counts the position of the test statistic relative to the observed value, as the exact approach would have done.

The majority of the execution time of `tsrtest` arises from the actual calculations performed within the command it calls, not from the overhead of generating combinations or from calling a command and passing it parameters. For the example in section 4 using `meandiff`, commenting out the body of that program and substituting a dummy assignment of an r-class value reduced total execution time by about 40%, whereas a

similar example involving the use of `tab1` (see [R] **tabulate oneway**) (see the example in section 7.3 using `diffmulti`) reduced execution time by more than 90%. Thus care exercised in choosing a test statistic, and in selecting an existing Stata command or in implementing a user-written command, can offer considerable time efficiencies when using `tsrtest`.

6 Full description of syntax

6.1 Syntax

`tsrtest` *groupvar expr* [*if*] [*in*] [`using` *filename*] [`, quiet nodots nodrop reps(#) simsec(#) nullvalue(#) exact overwrite`]: *command*

6.2 Arguments

This syntax is similar to that of `permute`. In fact, one could use `permute` to approximate the results of this exact test, but with `tsrtest`, one would obtain a true exact test based on a complete enumeration of all possible arrangements.

`tsrtest` expects its first argument, *groupvar*, to be the grouping variable, which can be any dichotomous variable. In the process of calculating *p*-levels, this variable is reassigned but is restored when the program terminates . The second argument, *expr*, specifies the test statistic, e.g., `r(mean)` (also see [R] **saved results** and [P] **return**), which is generated by the command specified after the colon. It is possible to limit the observations included with `if` or `in`.

Additionally, one has the possibility of generating a `.dta` file that contains in its first line the observed value of the test statistic and in its subsequent lines all values of the test statistics from the computed combinations; this can be achieved by specifying `using` *filename*.

6.3 Options

`quiet` suppresses information about hypotheses and results.

`nodots` suppresses the display of dots.

`nodrop` specifies to not drop the observations excluded by `if` or `in`. By default, every observation not included by `if` or `in` gets (temporarily) dropped.

`reps(#)` stipulates that # random group assignments be performed if simulating. The default is `reps(10000)`.

`simsec(#)` stipulates that an exact test be used if its estimated execution time would not exceed # seconds. If the estimated execution time exceeds # seconds, `tsrtest` will revert to Monte Carlo simulation. The default is `simsec(1000)`.

`nullvalue(#)` specifies that the null value of the test statistic is the real number #. The default is `nullvalue(0.0)`.

`exact` forces the calculation of exact p-values, even if the estimated execution time would exceed the time specified in `simsec()`.

`overwrite` specifies to overwrite the results file indicated with `using`.

6.4 Saved results

`tsrtest` saves the following in `r()`:

Scalars

`r(repetitions)`	number of repetitions done if Monte Carlo simulation was executed; if exact solution was done, reverts to missing
`r(simulated)`	`1` if results came from Monte Carlo simulation, `0` if exact p-values were obtained from full randomization solution
`r(missing)`	fraction of all combinations for which calculation of test statistic gave a missing value
`r(twotail)`	fraction of all combinations for which absolute value of test statistic minus null value was greater than or equal to that same quantity for data distribution of original sample
`r(uppertail)`	fraction of all combinations for which value of test statistic minus null value was greater than or equal to that for observed data of original sample
`r(lowertail)`	fraction of all combinations for which value of test statistic minus null value was less than or equal to that for observed data of original sample
`r(obsvStat)`	value of test statistic computed on observed data distribution of original sample
`r(combinations)`	number of combinations of possible assignments of cases to two groups, without regard to order, i.e., ${}_{n_1+n_2}C_{n_1}$. If the exact solution was obtained, this is the actual number of data arrangements on which the p-values are based; if a simulated solution was obtained, this is the number that would have been required for an exact solution.

7 Applications of tsrtest

7.1 Somers' D

As detailed by Newson (2002), Somers' D has various uses and linkages to statistics of interest. In the current situation, we use it to measure a difference in location for an ordinal response with a binary explanatory variable, and we use `tsrtest` to give an exact test of no difference.

▷ Example

Suppose we are interested in gender differences in beliefs about the effect of mothers' work situations on their relationships with their children, and more particularly whether such gender differences hold up among younger, relatively educated people in high

prestige occupations (where we might expect that such differences would be small). The `ordwarm2` dataset, drawn from the 2006 United States General Social Survey (see Davis, Smith, and Marsden [2006]), contains the statement "a working mother can establish just as warm and secure a relationship with her children as a mother who does not work", which is coded as 4 equals "strongly agree" down to 1 equals "strongly disagree". To test the hypothesis of no difference in location between women and men against the hypothesis that men are more likely to disagree, we need a Stata program to calculate Somers' D and return it as an r-class result. Although this could be done by writing a small wrapper for Newson's `somersd` to make an r-class result from the e-class result given by `somersd`, this is not the best approach, because economy of calculation effort is crucial and `somersd` does many things besides calculate Somers' D. Consequently, we have created a stripped-down program to calculate Somers' D, taking advantage of the identity between Somers' D with a binary explanatory variable and the rank-biserial correlation coefficient (demonstrated by Newson [2008]; see related material in Cureton [1956]), which is a simple function of the difference in mean rank across groups:

```
program somd, rclass
args y group g1 g2
// Assumes response variable has been converted to ranks
  quietly {
    count
    local n = r(N)
    summarize `y´ if `group´==`g1´, meanonly
    local y1=r(mean)
    summarize `y´ if `group´==`g2´, meanonly
    local y2=r(mean)
    local d=(2/`n´)*(`y2´-`y1´)
 }
  return scalar d = `d´
end
```

Before running the example, we convert the response variable to ranks by using Stata's built-in `egen rank()` function. We then run `tsrtest` with `somd` as the command to be called:

```
sysuse ordwarm2
// younger, relatively well-educated and prestigious
keep if yr89 & (ed >=16) & (prst >= 70) & (age < 50)
// convert response variable to ranks
egen rwarm=rank(warm)

tab2 warm male
tsrtest male r(d), nullvalue(0): somd rwarm male 0 1
```

(*Continued on next page*)

We obtain the following results:

Mother has warm relationship	Gender: 1=male 0=female Women	Men	Total
D	0	4	4
A	4	9	13
SA	4	3	7
Total	8	16	24

```
Two-sample randomization test for theta=r(d) of somd warm male 0 1 by male

Combinations:   735471 = (24 choose 8)
Assuming null=0
Observed theta: -.4375

Minimum time needed for exact test (h:m:s):  0:01:58
Mode: exact

progress: |.....................................|

 p=0.98161 [one-tailed test of Ho:  theta(male==0)<=theta(male==1)]
 p=0.05242 [one-tailed test of Ho:  theta(male==0)>=theta(male==1)]
 p=0.07711 [two-tailed test of Ho:  theta(male==0)==theta(male==1)]
```

◁

These exact results (elapsed time = 122 seconds) indicate that among people of relatively high education and prestige and aged 50 years or younger, there is some evidence ($p = 0.052$) against the randomization hypothesis of no difference. By contrast, a more conventional normal theory test, using the jackknife standard error given by Newson's `somersd`, gives a considerably stronger one-sided $p = 0.012$ for this small sample.[3]

7.2 Difference in relative variation

Paleontologists and zoologists often are interested in comparing the amount of variation in some quantitative trait across two samples of animals. There are various approaches to such tests, all of which compare the level of variation between the two samples while adjusting for differences in the means between the two samples (Plavcan and Cope 2001). Such mean adjustments are necessary since one cannot, to take a classic example, meaningfully compare the standard deviation of tail length in a sample of mice with that of a sample of elephants, because the scale difference would dominate any comparison of variability. One common approach is to base the test on a comparison of the coefficients of variation (CVs) of the two samples, where $\text{CV} = 100s/\overline{Y}$, or equivalently, to compare the standard deviations after rescaling the data in each sample by dividing each score

3. However, if one specifies both the `transf(z)` and `tdist` options with `somersd`, the p-value adjusts to a more reasonable value of 0.063. See Newson (2007) for a discussion.

by its sample mean. One method of testing CVs simply assumes normality of the underlying trait and treats the ratio of squared CVs as an F statistic, whereas other approaches assume normality of the underlying trait distributions but use a Monte Carlo approach to generate the sampling distribution. See Donnelly and Kramer (1999) for the former approach and also for a comprehensive survey of tests of relative variation; see Cope and Lacy (1995) for an example of the latter approach. Because paleontologists often possess small samples and cannot be certain of normality, randomization tests are useful, so this situation provides another apt illustration of the use of `tsrtest`.

Example

Consider the data reported by Plavcan and Cope (2001, 210), in which they compare the variation in skull length (mm) in a sample of 10 pygmy marmosets and 10 orangutans. These two primates vary dramatically in typical size, as seen in the descriptive statistics reported (for the marmosets versus for the orangutans), but the CV values suggest a modest difference in relative variation (1.74 versus 2.73). A relevant randomization test for a difference in relative variation, then, would address whether the CV in the orangutan sample exceeds that in the marmoset sample beyond what could result from random assignment of the (mean-adjusted) individual values to the two species. To conduct this test with `tsrtest`, the data in each of the two samples were first rescaled by dividing each score by its sample mean, which gives each observed sample a standard deviation equivalent to its CV. A user-written Stata program is used to compute the test statistic, the difference of the sample standard deviations, which is then called by `tsrtest`. The file `copeplavcan.dta` has to reside in the current working directory for this example to work.

```
use copeplavcan.dta

// Difference of two standard deviations
capture program drop sddiff
program sddiff, rclass
   args y group g1 g2
   quiet summarize `y´ if `group´ ==`g1´
   local sd1 = r(sd)
   quiet summarize `y´ if `group´ ==`g2´
   return scalar sddiff = r(sd) - `sd1´
end
// rescale by mean
summ lengthskull if orang ==1
local CVorang = 100 * r(sd)/r(mean)
local df1 = r(N) - 1
summ lengthskull if orang ==0
local CVmarmo = 100 * r(sd)/r(mean)
local df2 = r(N) - 1
bysort orang: egen m = mean(lengthskull)
gen lenadj = lengthskull/m
bysort orang: summ lenadj

tsrtest orang r(sddiff): sddiff lenadj orang  0 1
```

The preceding code required approximately 22 seconds to run on the Windows machine cited in section 4, using Stata 9.2, and gave the following results:

```
Two-sample randomization test for theta=r(sddiff) of sddiff lenadj orang  0 1 by
> orang

Combinations:   184756 = (20 choose 10)
Assuming null=0
Observed theta: .0099

Minium time needed for exact test (h:m:s):  0:00:15
Mode: exact

progress: |....................................|

 p=0.09663 [one-tailed test of Ho:  theta(orang==0)<=theta(orang==1)]
 p=0.90338 [one-tailed test of Ho:  theta(orang==0)>=theta(orang==1)]
 p=0.19326 [two-tailed test of Ho:  theta(orang==0)==theta(orang==1)]
```

◁

The relatively large p-value of 0.09663 suggests that these data provide weak evidence against the randomization hypothesis of no difference. In this case, the conventional normal theory test based on $F = (\mathrm{CV}_2/\mathrm{CV}_1)^2$ with $\mathrm{df}_1 = n_1 - 1$ and $\mathrm{df}_2 = n_2 - 1$ gave a p-value = 0.0978, which is quite close to the exact p-value obtained here. Such a good result, of course, need not occur for all datasets.

7.3 Difference in two multinomials

A final example application of `tsrtest` involves an omnibus test for the difference of two multinomials, a test that could conventionally be done as a χ^2 test, or, in a small sample, by using an extended form of Fisher's exact test, as implemented in the `exact` option of `tab2`. Although the calculation of Fisher's exact test is very fast, there is no simple and clear measure of association to which it corresponds once we go beyond the odds ratio underlying its use for a 2×2 table. For a $k \times 2$ table, `tsrtest` allows us to implement a test statistic that does correspond to an understandable measure of association, and it allows a test that explicitly involves an explanatory and a response variable.

▷ Example

As an empirical example, consider data on religious preference and the geographical region of residence among persons in the United States, as derived from the 2006 General Social Survey (Davis, Smith, and Marsden 2006). More particularly, consider comparing the regional distribution of U.S. residents who identify as Muslim and those who identify as Hindu, for which the observed data are

```
. use gss06religregion, clear
(General Social Surveys, 1972-2006: [Cumulative File], Dataset 0001)
. keep if relig == 7 | relig ==9 //hindu and muslim
(4482 observations deleted)
. tab2 region relig, chi2 exact
```

REGION OF INTERVIEW	RS RELIGIOUS PREFERENCE HINDUISM	MOSLEM/IS	Total
MIDDLE ATLANTIC	1	5	6
E. NOR. CENTRAL	2	4	6
W. NOR. CENTRAL	0	1	1
SOUTH ATLANTIC	6	2	8
E. SOU. CENTRAL	0	1	1
MOUNTAIN	0	2	2
PACIFIC	2	2	4
Total	11	17	28

```
          Pearson chi2(6) =   8.4349   Pr = 0.208
           Fisher´s exact =                 0.209
```

As an alternative to either the asymptotic χ^2 test or Fisher's exact test (which give surprisingly similar results even in this sparse table), suppose we define a more intuitively meaningful measure of the difference between the two distributions of regional location, namely, the sum of the absolute differences in the row proportions of the sample distributions

$$d = \sum_{i=1}^{k} | \widehat{\pi}_{i1} - \widehat{\pi}_{i2} |$$

where k is the number of rows, and $\widehat{\pi}_{ij}$ is the proportion of persons in the jth column that fall in the ith row. In the context of a randomization test (also true for the χ^2 test or Fisher's exact test), the row and column marginals are fixed, so an equivalent and less calculation-intensive statistic can be used for testing purposes:

$$d = \sum_{i=1}^{k} | \widehat{\pi}_{i1} - \widehat{\pi}_{i\cdot} |$$

where $\widehat{\pi}_{i\cdot}$ represents the marginal row proportion for the ith row, which remains constant across all random allocations to the columns. This statistic can be calculated with a short program:

(*Continued on next page*)

```
capture program drop diffmulti
program diffmulti, rclass
// Calculates sum of absolute difference of first column versus
// marginal proportions.
   args y x m xcol1 n1 nyval v1 v2 v3 v4 v5 v6 v7
// y and x are the response and explanatory variables,
// m is a one column matrix of the marginal relative //
// frequencies, xcol1 is the value of the x variable for the
// first column, n1 is the first column marginal frequency,
// nyval is the number of possible values for y, and v1-v7
// is a list of the possible values of y
   local sumd = 0
   forval i = 1/`nyval´  {
      quietly count if `x´ == `xcol1´ & `y´ == `v`i´´
      local sumd = `sumd´  + abs(r(N)/`n1´- `m´[`i´,1] )
   }
   return scalar d = `sumd´
end
```

This program is called by `tsrtest` with precalculation of the relevant program parameters,

```
// Get marginals, and column one sum.
tab1 region, matcell(marg)
mat marg = marg/r(N)  // marginal relative freq
count if relig == 7
local n1 = r(N)
// Get value list for response variable
levelsof region
local vals = r(levels)
// count values of y
local nval = wordcount("`vals´")
tab1 region, matcell(marg)
mat marg = marg/r(N)  // marginal relative freq
local rows = rowsof(marg)
count if relig == 7  // first column total
local n1 = r(N)
tsrtest relig r(d), exact: diffmulti region relig ///
                           marg  7 `n1´ `nval´ `vals´
```

and gives the following results (execution time = 11,485 seconds):

```
Two-sample randomization test for theta=r(d) of diffmulti region relig marg  7 1
> 1 7 2 3 4 5 6 8 9 by relig

Combinations:   21474180 = (28 choose 11)
Assuming null=0
Observed theta: .5974

Minimum time needed for exact test (h:m:s):  3:15:03

WARNING: This will take *very* long! If this is not what you intended to do, hit
> BREAK and repeat the command without specifying the ´exact´ option!

Mode: exact

progress: |.....................................|

 p=0.15717 [one-tailed test of Ho:  theta(relig==7)<=theta(relig==9)]
 p=0.84416 [one-tailed test of Ho:  theta(relig==7)>=theta(relig==9)]
 p=0.15717 [two-tailed test of Ho:  theta(relig==7)==theta(relig==9)]
```

◁

Here a randomization test using a meaningful but nonstandard test statistic gave more evidence against the null hypothesis of no difference than did either the conventional χ^2 test or the Fisher's exact test, but it did not yield a sufficiently small p-value to meet conventional standards for rejecting the null hypothesis. (The two-tailed and one-tailed tests are identical here, because there is no sense of direction for the difference of two multinomial distributions.) From a programming efficiency point of view, the current example also is instructive. An earlier version of `diffmulti` used `tab1` to obtain frequency counts, which was quite a slow method. Changing to `count` (see [D] **count**) to obtain frequencies reduced execution time by about 70%, again showing the importance of judicious approaches to calculation in the user's program.

7.4 Other applications

There are countless ways of applying the randomization test to test statistics of two independent samples. One possible application is the ranksum test for the equality of medians by Mann and Whitney (1947), also known as the Mann–Whitney U test. Stata already provides an asymptotic implementation of this test (see [R] **ranksum**) because exact p-levels can be tedious to derive even for moderate sample sizes (Narayanan and Watts 1996). It has, however, been demonstrated (e.g., by Edwardes [2000]) that under certain conditions the asymptotic test can yield much smaller and thus misleadingly liberal p-values. By calculating the exact p-values through a randomization test, you can avoid this pitfall. According to Siegel and Castellan (1988, 155), p-levels of the Mann–Whitney test for the equality of medians can be computed exactly through a two-sample randomization test for differences in means applied to the ranks of the variable observed. The proceeding is largely identical to the example outlined in section 4 except that the ranks of the values in the combined sample are used instead of the values themselves.

For convenience, the supplemental program `mwtest` provides the Stata user with a routine to carry out this test. Similar routines are provided for calculation of the randomization test difference in means (`fptest`), Somers' D (`somersdtest`), the relative variation (`sddiff`), and the standard deviation of samples that are centered around their means (`vartest`).

8 Conclusion

This article presented a new Stata command, `tsrtest`, that allows users to efficiently conduct a wide variety of two-sample randomization tests. `tsrtest` was presented in examples involving two-sample tests for difference of means, ordinal location, relative variation, and multinomial distributions.

9 References

Cope, D. A., and M. G. Lacy. 1995. Comparative application of the coefficient of variation and range-based statistics for assessing the taxonomic composition of fossil samples. *Journal of Human Evolution* 29: 549–576.

Cureton, E. E. 1956. Rank-biserial correlation. *Psychometrika* 21: 287–290.

Davis, J. A., T. W. Smith, and P. V. Marsden. 2006. General social surveys, 1972–2006 [cumulative file] (ICPSR 4697). Ann Arbor, MI. Inter-university Consortium for Political and Social Research.

Donnelly, S. M., and A. Kramer. 1999. Testing for multiple species in fossil samples: An evaluation and comparison of tests for equal relative variation. *American Journal of Physical Anthropology* 108: 507–529.

Edgington, E. S., and P. Onghena. 2007. *Randomization Tests*. 4th ed. Boca Raton, FL: Chapman & Hall/CRC.

Edwardes, M. D. deB. 2000. Implications of random cut-points theory for the Mann–Whitney and binomial tests. *Canadian Journal of Statistics* 28: 427–438.

Gentleman, J. F. 1975. Algorithm AS 88: Generation of all nCr combinations by simulating nested Fortran DO loops. *Applied Statistics* 24: 374–376.

Kaiser, J. 2007. An exact and a Monte Carlo proposal to the Fisher–Pitman permutation tests for paired replicates and for independent samples. *Stata Journal* 7: 402–412.

Manly, B. F. J. 2006. *Randomization, Bootstrap and Monte Carlo Methods in Biology*. 3rd ed. Boca Raton, FL: Chapman & Hall/CRC.

Mann, H. B., and D. R. Whitney. 1947. On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics* 18: 50–60.

Narayanan, A., and D. Watts. 1996. Exact methods in the NPAR1WAY procedure. In *Proceedings of the Twenty-first Annual SAS Users Group International Conference*.

Newson, R. 2002. Parameters behind "nonparametric" statistics: Kendall's tau, Somers' D and median differences. *Stata Journal* 2: 45–64.

———. 2007. Robust confidence intervals for Hodges–Lehmann median difference. 2007 UK Stata Users Group meeting. Downloadable from http://ideas.repec.org/p/boc/usug07/01.html.

———. 2008. Identity of Somers' D and the rank biserial correlation coefficient. http://www.imperial.ac.uk/nhli/r.newson/miscdocs/ranksum1.pdf.

Plavcan, J. M., and D. A. Cope. 2001. Metric variation and species recognition in the fossil record. *Evolutionary Anthropology* 10: 204–222.

Siegel, S., and N. J. Castellan Jr. 1988. *Nonparametric Statistics for the Behavioral Sciences*. 2nd ed. Columbus, OH: McGraw–Hill.

About the authors

Johannes Kaiser is a PhD student in the Laboratory for Experimental Economics at the University of Bonn, Germany.

Michael Lacy is an associate professor in the Sociology Department at Colorado State University in Fort Collins, Colorado.