

Paper SD-151

In the Pursuit of Balanced Groups: A SAS® Macro for an Adaptive Randomization Test

Tyler Hicks, University of Kansas, Lawrence, KS
 Jeffrey D. Kromrey, University of South Florida, Tampa, FL

ABSTRACT

In adaptive randomized experiments, researchers verify that random allocation yielded equivalent experimental groups before proceeding with data analysis. Given a precise definition of balanced groups (e.g., Cohen's $d < 0.5$ for a continuous covariate), researchers may plan to keep reshuffling participants until balance is achieved. However, reshuffling can invalidate standard parametric tests, such as t -tests, F -tests, and χ^2 -tests. This paper describes a non-parametric *analog* of the two-sample t -test, called an adaptive randomization test, which can preserve Type I error control in adaptive randomized experiments. Although such tests are well established in mainstream statistics (Morgan & Rubin, 2015), they are not readily available in SAS making them difficult to implement. This paper provides an overview of adaptive randomization tests, presents a macro for doing them in base SAS, and concludes with an executed example of the macro.

Keywords: SAS Macro, RCT, RESHUFFLING, ADAPTIVE RANDOMIZATION TEST, BASE SAS, SAS/STAT

INTRODUCTION

R.A. Fisher (1925) championed randomization of units into groups. This was an innovation for which he had to fight to convince his contemporaries to implement (Hacking, 1988). Many were worried it might produce imbalanced allocations. With k independent covariates, the chance of at least one covariate showing a statistically significant difference, at significance level α , is $1 - (1 - \alpha)^k$. Given 10 covariates, there is almost a 40% chance of obtaining an imbalanced sample that is statistically significant at the 5% level (Morgan & Rubin, 2015). Yet, because randomization underpinned null hypothesis testing, Fisher's idea ultimately prevailed among the next generation of researchers (Howson & Urbach, 2006).

The internal validity of a randomized experiment can be enhanced with a blocking design (Ramachandran & Tsokos, 2009). In a randomized block design, units are deliberately divided into blocks (i.e., subgroups), such that units within a block are matched on prognostic baseline covariates. After blocking, units within each block are randomly allocated into control and treated groups. This design assures that experimental groups will be balanced on prognostic baseline covariates. Moreover, it preserves the validity of standard parametric tests (e.g., t -tests, F -tests, and χ^2 -tests), as long as the blocking is taken into account in the analysis (e.g., using a correlated-means t -test to account for paired observations).

Adaptive methods can also correct for imbalanced groups. In an adaptive randomized design, units are allocated into control and treated groups at the outset but experimenters check group equivalency on relevant covariates before proceeding with the experiment. When imbalance is detected, using a preset criterion (e.g., Cohen's $d < 0.5$), the units are reshuffled until a passable balance is reached. Blocking and adaptive methods may even be used concurrently in the same study to increase internal validity. Perhaps, blocking is used to assure balance on covariate X_1 while reshuffling is used on covariates X_2 and X_3 .

An important caveat about adaptive methods is that they, unlike block designs, invalidate the p -values of the standard parametric tests. When experimenters plan to assure balance by reshuffling, even if they got lucky and did not need to reshuffle, they should carefully consider the choice of analytic tool. For example, the classic t -test assumes that the researcher would *always* have stuck with the original random allocation no matter how imbalanced the allocation was on a covariate. When this assumption is violated the t -test will misbehave. It will be too conservative given an adaptive reshuffling protocol (i.e., the t -test will overcontrol Type I error rates).

OVERVIEW OF THE ADAPTIVE RANDOMIZATION TEST

The adaptive randomization test described in this section, which is a non-parametric analog of the two-sample t -test, is appropriate for adaptive randomized experiments. It is a special case of non-parametric permutation testing. Fisher (1925) marketed this specific test as a distribution-free analytic tool for independent data. His phrase *distribution-free*

is a somewhat misleading description. What Fisher really intended to communicate was that the requisite sampling distribution could be constructed using the sample data rather than theorized, if the data were randomized.

Permutation tests can evaluate a special null hypothesis, briefly:

$$H_0: \text{The independent variable has no effect.}$$

This null hypothesis is distinct because it can be stated without reference to any population parameters (e.g., $\mu_d = 0$). In general, permutation tests have three basic steps.

1. Compute a test statistic using the original set of units (e.g., difference between sample means).
2. Regroup units into all possible control and treated group permutations, and compute the new test statistic each time.
3. Calculate the permutation test's p -value, which is the proportion of test statistic values that equal or exceed the value of the test statistic from the original data.

Fisher's classic example of a permutation test suffices for illustration. Consider the case of the hypothetical tea-drinking lady who claimed that she could tell from taste alone whether the milk or tea had been poured into the cup first. Fisher proposed a randomized experiment to evaluate her claim. He prepared eight cups of tea. In four, he poured tea first (otherwise, the milk). He then presented the cups in a random order and asked her to guess which four had the milk poured in first. Other prognostic baseline covariates were held constant (e.g., size, temperature, etc.). After the lady tasted each of them, she made her final guess (i.e., she classified 4 of the 8 cups correctly).

The test statistic was the number of correct guesses. The null hypothesis was that her guesses were comparable to random guesswork. Her classification would have been the same regardless of the true order, if the null was correct. Since the order of the cups was randomized, Fisher adduced that there were 70 permutations possible, $[\binom{8}{4} = 70]$. She thus had a $1/70 = 0.0143$ chance of randomly guessing the correct order. So H_0 would be rejected at $\alpha = 0.05$ if she guessed perfectly. But in this instance, Fisher failed to reject the null hypothesis as she did not get the right order.

Fisher's classic example of a permutation test involved discrete data but the logic applies to continuous data as well. The difference is that with continuous data it may not be possible to analytically derive all relevant permutations, even with computers and combinatoric simplifications. A modern solution to this problem is to empirically simulate the requisite sampling density and substitute an approximate p -value for the unknown exact one. When the p -value is so approximated the test is called a randomization test rather than a permutation test. The steps to a randomization test are similar to the permutation test.

1. Compute a test statistic using the original set of units (e.g., difference between sample means).
2. Randomly scramble the units into new groups, and compute the new test statistic.
3. Carry out step 2 a large number of times to simulate the permutation distribution.
4. Calculate the randomization test's p -value, which is the proportion of test statistic values in the simulated permutation distribution that equal or exceed the value of the test statistic from the original data.

With a sufficiently large simulated sample of the permutation distribution ($n=1000, 5000, 10000$), the randomization test should yield a p -value that is practically equivalent to the p -value of the exact permutation test. Randomization tests are thus replacements for permutation tests, if one is willing to accept an approximate p -value.

An *adaptive* randomization test goes just one step further than a randomization test in order to take into account the study's reshuffling protocol. It has two steps between steps 3 and 4 of the randomization test above.

- 3a. Screen permutations in the simulated permutation distribution for imbalance on covariates, using a preset balance criterion, and then reshuffle those imbalanced permutations until balance is reached.
- 3b. Substitute the test statistic obtained from the final balanced permutation for the one computed from the unbalanced permutation.

The inclusion of these two steps differentiates an adaptive randomization test from a randomization test.

There are several advantages of an adaptive randomized test over a classic t -test of independent means. First and foremost, it preserves Type I error control in an adaptive randomized experiment (Morgin & Rubin, 2015). This makes it possible to improve the rigor of the study. Second, it can accommodate any sampling protocol (i.e., random, deliberate, or convenience). Its validity does not depend on how the original sample was collected. It just assumes

that after the units were collected, they were randomly allocated into control and treated conditions. Third, a null hypothesis without population parameters is transparent even to an audience without statistical training.

SOFTWARE LIMITATIONS

SAS does not currently provide an adaptive randomized test procedure. Thus, in order for SAS users to align their statistical analysis with their experimental design via an adaptive randomized test, extra work is required. Typically, simulation is needed to obtain the requisite sampling densities to conduct an adaptive randomization test. Unfortunately, the coding can be very labor intensive. Thus, in an effort to reduce the burden of programming an adaptive randomization test from scratch, this paper provides a SAS macro to conduct an adaptive randomization test.

SAS MACRO FOR AN ADAPTIVE RANDOMIZATION TEST

To execute an adaptive randomization test, four macros can be installed: a *screener* macro, a *subset* macro, a *shuffle* macro and a *main* macro. The main macro, which performs steps 1-4 described on page 2, internally calls on supporting macros, which perform steps 3a and 3b. Users only need to provide arguments for the main macro. The arguments for the supporting macros are automatically supplied when the main macro runs.

Users provide the main macro with five arguments: (a) The name of the dataset, (b) the name of the independent variable, (c) the name of the dependent variable, (d) the names of the covariate or covariates, (e) the requested number of randomizations to be used, and (f) the requested effect size value to demarcate balanced vs. imbalanced samples. A larger number of randomizations will yield more exact p -values yet they require greater execution times. A replication of 1000 or 5000 randomizations usually suffices. The main macro has four outputs: (a) sample size, (b) obtained effect, (c) the approximate two-tailed p -value, and (d) the number of randomizations used to obtain the p -value.

By way of introduction, please note that all variables in the analysis are assumed to be continuous (except the independent variable which is assumed to be dichotomous). The independent variable can either alphanumeric or numeric. Users can also list one or more continuous covariates. If one covariate is listed, Cohen's d is used to quantify imbalance. If multiple covariates are listed, Mahalanobis distance is used to quantify imbalance. This duality of effect measures allows the macro to run more efficiently. Users can also set the criterion for imbalance as high or low as they see fit. For example, setting the criterion to 0.5, a standard default, defines imbalance on covariates to be a Cohen's d or Mahalanobis distance at or above 0.5.

To validate the technical adequacy of this macro, a simulation study was conducted. The procedure was found to control Type I error rates in a series of repeated applications as intended. Specifically, when a 5% alpha level was requested, the test had a 5.3% Type I error rate in 1,000 repeated applications. This simulation study demonstrates that the macro works.

The macro code is below, followed by an example. It can handle multiple covariates and unbalanced group sizes.

```
*+-----+
| The Three Supporting Macros |
+-----+;
*SCREENER MACRO
This macro screens the randomization sampling distribution and selects
samples with imbalanced groups for removal. It detects the number of
covariates and adjusts screening. If there is one covariate Cohen's D is used
for screening. If there are multiple covariates Mahalanobis Distance is used
for screening.;

%MACRO SCREENER(DATA, N_COVARIATES);
  %IF &N_COVARIATES = 1 %THEN %DO;
    PROC TTEST DATA=&DATA; BY REPLICATE; CLASS NIV; VAR &COVARIATE;
    ODS OUTPUT STATISTICS=COVARY; RUN;
    DATA COVARY; SET COVARY; IF CLASS='Diff (1-2)';
    EFFECT=ABS(MEAN/STDDEV); IMBALANCED=0; RUN;
  %END;
  %IF &N_COVARIATES > 1 %THEN %DO;
    PROC CANDISC DATA=&DATA DISTANCE; BY REPLICATE; CLASS NIV; VAR
    &COVARIATE; ODS OUTPUT DIST=COVARYJ; RUN;
```

```

        DATA COVARY; SET COVARYJ; IF FromNIV=0; EFFECT=_1; IMBALANCED=1;
        RUN;
    %END;
%MEND SCREENER;

*SUBSET MACRO
After the screener macro identifies all the imbalanced samples in the initial
sampling distribution this macro removes them and puts them into a separate
dataset for a subsequent step;
%MACRO SUBSET (DATASET, VARIABLE, INDEPENDENT, DEPENDENT, COVARIATE,
N_COVARIATE);
    %IF &N_IMBALANCED > 0 %THEN %DO; *EXCUTES IF IMBALANCED SAMPLES;
        %DO SUB = 1 %TO &N_IMBALANCED;
            DATA NEED; SET &DATASET;
            IF &VARIABLE = &SUB;
                %SHUFFLE (NEED, EFFECT, NIV, &DEPENDENT, &COVARIATE,
                &N_COVARIATE);
                PROC APPEND BASE = SD5 DATA = TRYIT FORCE; %END;
            PROC DATASETS; DELETE NEED;
        %END;
%MEND SUBSET;

*SHUFFLE MACRO
This macro repeatedly reshuffles imbalanced samples until balanced is
achieved or the 100th attempt is reached;

%MACRO SHUFFLE(DATA, EFFECT, INDEPENDENT, DEPENDENT, COVARIATE, N_COVARIATE);
    %LET I = 0; *THIS LIMITS THE NUMBER OF LOOP EXECUTIONS;
    %IF &N_COVARIATE = 1 %THEN %DO;
        %DO %WHILE (&EFFECT > &CRITERION); *TELLS THE LOOP WHEN TO STOP;
            DATA TRYIT;
                SET NEED;
                RANVAR = RANUNI(0);
            PROC SORT DATA = TRYIT; *OBSERVATIONS RANDOMLY RESHUFFLED;
                BY RANVAR;
            DATA TRYIT; SET TRYIT;
                NIV = 0;
                IF _N_ > &RATIO*&SIZE THEN NIV = 1;
            PROC TTEST DATA=TRYIT; BY REPLICATE; CLASS NIV;
            VAR &COVARIATE; ODS OUTPUT STATISTICS=t_STATS; RUN;
            DATA t_STATS; SET t_STATS;
                IF CLASS='Diff (1-2)'; D=ABS(MEAN/STDDEV);
                IF &I > 100 THEN D = 0; *FORCE EXIT AFTER 100 TRIES;
            CALL SYMPUT('EFFECT',trim(left(put(D,8.5)))); RUN;
            %let i=%eval(&i+1); * Increment i with each loop execution;
            %put _local_;
            %put _global_;
        %END;
    %END;
    %IF &N_COVARIATE > 1 %THEN %DO;
        %DO %WHILE (&EFFECT > &CRITERION);
            DATA TRYIT; SET NEED;
                RANVAR = RANUNI(0);
            PROC SORT DATA = TRYIT;
                BY RANVAR;
            DATA TRYIT; SET TRYIT;
                NIV = 0; IF _N_ > &RATIO*&SIZE THEN NIV = 1;

```

```

PROC CANDISC DATA=TRYIT distance;
  BY REPLICATE; CLASS NIV; VAR &COVARIATE;
  ODS OUTPUT DIST=COVAR1; RUN;
DATA COVAR1; SET COVAR1;
  IF FromNIV=0; MD=_1 ; KEEP REPLICATE MD; RUN;
DATA COVAR; SET COVAR1;
  IF &I > 100 THEN MD = 0; *force exit after 100 tries;
CALL SYMPUT('EFFECT',trim(left(put(MD,8.5))));
RUN;
%let i=%eval(&i+1); * Increment i with each loop execution;
%END;
%MEND SHUFFLE;

*Main MACRO for the adaptive randomization test. It computes the p-value and
simulates the sampling distribution;
%MACRO AdaptiveRandomizationTest (DATASET, INDEPENDENT, DEPENDENT, COVARIATE,
N_REP=1000, CRITERION=.5);
  %PUT _LOCAL_; %PUT _GLOBAL_;
  *ods output off;
  ods graphics off;
  ods exclude all;
  ods noresults;
  *Delimits data set to only DV, IV, and Covariates;
  DATA &DATASET; SET &DATASET;
  KEEP &INDEPENDENT &DEPENDENT &COVARIATE; RUN;
  *Executes listwise deletion;
  DATA &DATASET; SET &DATASET;
  IF CMISS(OF _ALL_) THEN DELETE; RUN;
  *Ensures IV is numeric;
  PROC MEANS DATA=&DATASET; BY &INDEPENDENT; VAR &DEPENDENT; OUTPUT
OUT=DECOY MEAN=M;
  DATA DECOY; SET DECOY; IF _N_=1 THEN NIV=0; IF _N_=2 THEN NIV=1;
KEEP &INDEPENDENT NIV;
  PROC SORT DATA=&DATASET; BY &INDEPENDENT;
  DATA &DATASET; MERGE &DATASET DECOY; BY &INDEPENDENT; RUN;
  *Extracts number of COVARIATE;
  DATA COVARIATE; SET &DATASET; KEEP &COVARIATE;
  DATA _NULL_; SET COVARIATE; CALL
SYMPUTX('N_COVARIATE',COUNT(CATX('09'X,OF _ALL_), '09'X)+1); STOP; RUN;
  *Extracts treated and control ratio AND sample size;
  PROC MEANS DATA=&DATASET N MEAN; VAR NIV; OUTPUT OUT=GROUP_RATIO
MEAN=RATIO N=SIZE;
  DATA _NULL_; SET GROUP_RATIO; CALL SYMPUTX('RATIO' ,RATIO); CALL
SYMPUTX('SIZE',SIZE); RUN;
  *Extracts obtained test statistic;
  PROC TTEST DATA=&DATASET; CLASS NIV; VAR &DEPENDENT; ODS OUTPUT
STATISTICS=STATS; RUN;
  DATA STATS; SET STATS;
  IF CLASS='Diff (1-2)' ;
  D=ABS(MEAN/STDDEV); CALL SYMPUTX('Obtained_D' ,D); RUN;
  *Simulate Randomization Sampling Density;
  DATA &DATASET; SET &DATASET; DUMMY_ID = _N_; RUN;
  PROC SURVEYSELECT DATA=&DATASET OUT=SD1 SEED=123 METHOD=SRS
SAMPRATE=&RATIO REPS=&N_REP; RUN;
  DATA SD1; SET SD1; NIV=1; RUN; PROC SORT DATA=SD1;
  BY REPLICATE DUMMY_ID;

```

```

PROC SURVEYSELECT DATA=&DATASET OUT=SD2 SEED=321 METHOD=SRS
  SAMPRATE=1 REPS=&N_REP; RUN;
DATA SD2; SET SD2; NIV = .; PROC SORT DATA=SD2;
  BY REPLICATE DUMMY_ID;
DATA SD3; MERGE SD2 SD1; BY REPLICATE DUMMY_ID;
DATA SD3; SET SD3; IF NIV=. THEN NIV=0; DROP DUMMY_ID; RUN;
PROC SORT DATA=SD3; BY REPLICATE; RUN;
*Screen sampling density for imbalanced samples;
  %SCREENER(DATA= SD3,N_COVARIATES=&N_COVARIATE);
*Count the number of imbalanced samples in the sampling density;
  DATA COVARY; SET COVARY; IF EFFECT > &CRITERION; IMBALANCED=1;
  KEEP REPLICATE IMBALANCED;RUN;
  PROC MEANS DATA=COVARY SUM; VAR IMBALANCED; OUTPUT OUT=TOTAL
  SUM=N_ImbalancedSamples; RUN;
  DATA _NULL_; SET TOTAL; CALL SYMPUTX('N_IMBALANCED',
  N_ImbalancedSamples); RUN;
*Exchange imbalanced samples with balanced samples;
  PROC RANK DATA=COVARY OUT=COVARY DESCENDING TIES=DENSE; VAR
  REPLICATE; RANKS Rep2; RUN;
  PROC SORT DATA=COVARY; BY REPLICATE;
  DATA SD4; MERGE SD3 COVARY; BY REPLICATE;
  DATA SD5a; SET SD4; IF REP2 NE .;
  DATA SD6; SET SD4; IF REP2 = .;
  %SUBSET(SD5a, REP2, NIV, &DEPENDENT, &COVARIATE, &N_COVARIATE);
  RUN;
*Adaptive randomization test;
  DATA SD8; SET SD6 SD5;
  PROC SORT DATA=SD8; BY REPLICATE;
  PROC TTEST DATA=SD8; BY REPLICATE; CLASS NIV; VAR &DEPENDENT;
    ODS OUTPUT STATISTICS=SD9; RUN;
  DATA SD9; SET SD9; IF CLASS='Diff (1-2)';
    D=ABS(MEAN/STDDEV); COUNT=0; RUN;
  DATA SD10; SET SD9; IF D > &Obtained_D THEN COUNT=1;
  PROC MEANS DATA=SD10 N MEAN noprint; VAR COUNT;
    OUTPUT OUT=RESULTS N=REPS MEAN=PVALUE;
*ods output on;
  ods graphics on;
  ods exclude none;
  ods results;
*Report output;
  TITLE 'RESULTS';
  DATA RESULTS; SET RESULTS;
  TESTSTATISTIC=&Obtained_D; SIZE=&SIZE; RUN;
  DATA OUTPUT; SET RESULTS;
  FILE PRINT notitles header =HD;
  PUT @5 SIZE @20 TESTSTATISTIC @39 pvalue @60 REPS;
  RETURN;
  HD: PUT @1 'Adaptive Randomizaion test' // @5 'Sample Size' @20
  'Obtained Effect' @39 'Two-Tailed p-value' @60 'Simulated N'
%MEND AdaptiveRandomizationTest; RUN;

```

EXAMPLE OF THE MACRO IN ACTION

Suppose that an intervention makes no difference on a continuous outcome (i.e., the null hypothesis is true) but an experimenter ignorant of this truth conducted a randomized clinical trial on a sample to test the intervention's promise. Her data had 30 records (control=15; treated=15). To increase her internal validity, she used an adaptive

randomization protocol to ensure group equivalency on three covariates. She used a multivariate effect size measure (i.e., Mahalanobis Distance [M]) to quantify imbalance using PROC CANDISC.

```
PROC CANDISC DATA=<Your dataset> DISTANCE;
    CLASS <Your Independent Variable>; VAR <Your Covariates>; RUN;
```

She defined imbalance on the three covariates to be $M > 0.5$. She obtained imbalanced samples the first two tries but got a balanced sample on the third attempt. After proceeding with the experiment using the final balanced allocation, she computed a slightly harmful effect ($d = -0.114$).

The following SAS code inputs the researcher's hypothetical dataset into SAS for analysis. In this data step, the variable ID is an identification number for each observation, INTERVENTION is an alphanumeric variable to differentiate the treatment group from the control group, OUTCOME is the outcome variable for the study, and the three NU variables (NU1 – NU3) are the three covariates to be monitored.

```
DATA ONE;
INPUT ID INTERVENTION $ OUTCOME NU1 NU2 NU3;
CARDS;
01 A -1.53958 2.0832 -3.19903 -17.0714
02 A 0.48471 0.7187 -0.12244 0.9840
03 A 0.34744 -5.7920 0.02811 -6.5629
04 A 0.05601 0.2451 -3.42413 -7.2665
05 A -1.64109 -14.600 -4.30257 -4.9164
06 A 1.02381 6.1346 -1.24883 4.6000
07 A -1.08931 -2.5810 1.28031 3.5874
08 A 1.09135 0.6173 6.14315 9.0597
09 A 0.51387 -5.3571 3.71445 -15.0572
10 A 0.34340 2.8248 1.30870 -1.2044
11 A 0.75228 -3.8451 2.90331 -0.8255
12 A -1.60946 1.9709 -2.60033 0.3364
13 A -0.58792 -1.5083 -5.41507 -9.4322
14 A -1.66991 -1.9983 0.19122 -10.4639
15 A 0.14042 0.0655 0.07779 -9.8183
16 B 0.90140 8.8166 6.60298 -3.1495
17 B 0.71912 9.6628 0.96187 6.4426
18 B 0.59267 4.1575 -4.35257 17.9101
19 B 0.33669 -5.4280 -4.12148 4.3510
20 B 0.20156 -1.7704 -0.15107 9.7133
21 B -0.33710 0.8993 2.24059 -7.8984
22 B 1.50408 -2.4540 6.89627 -4.6669
23 B -0.66232 1.1283 -1.38161 -1.7281
24 B 0.02802 12.7186 4.09207 0.5962
25 B -1.68220 0.2223 -4.25337 -15.2137
26 B -0.07880 0.1158 -0.41087 -5.3116
27 B -0.29250 -8.1577 4.35347 -9.5985
28 B -1.20356 -2.6962 1.72018 -21.7518
29 B -0.47698 3.3853 1.19385 -5.1315
30 B -1.30648 -7.9942 1.70587 -14.7338
;
```

To request an adaptive randomization test one can enter the following arguments into the main macro:

```
%AdaptiveRandomizationTest (DATASET=ONE, INDEPENDENT=INTERVENTION,
DEPENDENT=OUTCOME, COVARIATE=NU1 NU2 NU3, N_REP=1000, CRITERION=0.5); RUN;
```

The above code requests a randomization sampling distribution with 1000 iterations and an effect size of 0.50 as a balance criterion. Because three covariates are specified in the macro call, the macro will use Mahalanobis' Distance as the effect size index. When this procedure was executed the following information output was obtained, although slight deviation in a replication study can be expected if repeated due to simulation error in the sampling distribution:

RESULTS			
Adaptive Randomization test			
Sample Size	Obtained Effect	Two-Tailed p-value	Simulated N
30	0.1136419008	0.706	1000

Based on this output, the researcher declares the data to be statistically insignificant at the 5% significance level. The obtained test statistic is consistent with the null hypothesis (i.e., the obtained effect is the product of random sampling error).

ADVANTAGES OF IMPLEMENTING AN ADAPTIVE RANDOMIZATION TEST

It is backwards to conform the design of a randomized experiment to a statistical test. If it is recognized that the control and treated groups are not equivalent on an important prognostic baseline data before commencement of the experiment, then it seems pedantic to require researchers to continue the experiment to meet the assumptions of a statistical test. A more satisfactory solution seems to be to reshuffle units to balance the groups and then implement a different statistical test. If researchers are agreeable to this plan, then they should give serious consideration to the use of an adaptive randomization test.

Adaptive randomization tests, such as the one executed by this macro, are perhaps most useful in small sample analysis. Exact quantification of an effect in small sample analysis is difficult as 95% confidence intervals can be imprecise and point estimates can contain substantial prediction error. Yet, even with only limited data, researchers can still proceed to evaluate the null hypothesis with a statistical test. The successful rejection of a null hypothesis only becomes a more remarkable feat with limited data. Such a test can help researchers decide if the costs of collecting a big-sample to investigate the effect will likely be repaid or not. Hypothesis testing thus holds a prominent position in small sample research, and it will likely continue to do so long into the foreseeable future (Mohr, 1990).

The assumption of random selection is typically violated in small sample analysis. For example, researchers planning a pilot study often operate on a low budget and so may opt to perform convenience or deliberate sampling rather than random sampling. Randomization tests in general, and adaptive randomization tests in particular, are applicable irrespective of sampling protocol (convenience, deliberate, random). They only assume random assignment into control and treated groups occurred.

CONCLUSION

Researchers can capitalize upon adaptive randomization tests to statistically evaluate the null hypothesis, if random reshuffling was performed to balance groups (or was planned). The above macro makes it easy for researchers to put abstract statistical theory into practice in base SAS. Such testing can help researchers identify effects to be pursued in further research.

Although applications of adaptive randomization tests are relatively rare in social and applied science, adaptive randomization experiments are believed to be very common in social and applied science. However, it is impossible to obtain accurate statistics as many researchers fail to report if they planned to randomly reshuffle again, if needed, to correct imbalanced groups. Perhaps, ready access to this adaptive randomization test macro will encourage researchers to seriously consider if an adaptive randomization testing is a good fit for their research.

Researchers are also encouraged to closely examine the logic of the main macro so that they can tweak it to handle more complicated analytic situations. Currently, it only handles a two-sample *t*-test situation but this is only a starting place. The macro can easily be remade to handle other modeling situations. For example, the main macro currently invokes PROC TTEST but if it was reprogrammed to invoke PROC ANOVA or PROC LOGISTIC instead it could handle multiple groups or dichotomous variables. The logic of the main macro is thus very versatile and, in theory, researchers can redeploy it to cope with a variety of analytic needs.

REFERENCES

- Fisher, R. A. (1925). *Statistical Methods for Research Workers*. Edinburgh, UK: Oliver & Boyd.
- Hacking, I. (1988). Telepathy: Origins of randomization in experimental design. *ISIS*, 79, 427-51.
- Howson, C., & Urbach, P. (2006). *Scientific reasoning: The Bayesian approach* (3rd ed.). Chicago: Open Court.

Mohr, L. B. (1990). *Understanding significance testing*. New York, NY: Sage.

Morgan, K. L., & Rubin, D. B. (2015). Rerandomization to Balance Tiers of Covariates. *Journal of the American Statistical Association*, 110, 1412-1421.

Ramachandran, K., & Tsokos, C. (2009). *Mathematical statistics with applications*. Burlington, MA: Elsevier Academic Press.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the first author at:

Tyler Hicks, Ph.D.
University of Kansas
122 West Campus Rd,
Joseph R. Pearson Hall, Rm 521.
Lawrence, Kansas 66045
Email: tahicks@ku.edu

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.