```
/*SAS PROGRAM RANDOMPCA - S.J. Tonsor April 13, 2003
This program can be used by anyone. In exchange I expect to be cited
in any resulting publications and notified of any use.

This program tests for significant relationships among variables
using randomization.  It tests for significant correlations/covariances
and significant eigenvalues and eigenvector coefficients from Principal
Components Analyses.  It does this by shuffling the actual trait values
Among the observations in a data set.  For a data set with n observations
and m traits, m data sets are created, each containing one of the
variables.   For each of the m data sets, the order of the observations
is randomly permuted. The data sets are then re-merged, but in the order
of their independent random permutations.  This has the effect of
randomly associating one of the actual measured values of trait 'a'
with a randomly drawn actually measured value of trait 'b' and assigning them
to an observation. The covariance matrix and principle component eigenvectors
and eigenvalues are produced by PROC PRINCOMP. This is repeated NITERAT
(value set by user) times, with the values output to a data set. The
values are used with proc univariate to calculate randomization-based
99% confidence intervals on the null hypothesis distributions of each
of the elements of the "output out=" data set from proc princomp. The
confidence limits are compared to the actual values, and a table is
produced in the SAS output window, reporting significance or lack thereof.
THE OUTPUT TABLE CONTAINS THE ACTUAL VARIABLE VALUE FLANKED BY THE LO
AND HI 99% CONFIDENCE LIMITS, FOLLOWED BY ** WHEN SIGNIFICANT AND ..
WHEN NOT SIGNIFICANT.


CAUTION SHOULD BE EXERCISED IN INTERPRETING THE CONFIDENCE INTERVALS
ON THE EIGENVECTOR COEFFICIENTS WHEN THE MAGNITUDE OF THE EIGENVALUE
IS RELATIVELY SMALL, EVEN IF SIGNIFICANT.  WHEN EIGENVALUES FOR TWO
PCs ARE CLOSE IN VALUE, THEY CAN SWITCH IN MAGNITUDE, HENCE RANK,
BETWEEN RESAMPLE ITERATIONS.  THE PC IS ALSO PRONE TO REFLECTION.
THE CONFIDENCE INTERVALS ON THE COEFFICIENTS CAN BECOME VERY LARGE
AND PRONE TO TYPE II ERROR.  SEE Perez-Neto et al. 2003 Ecology 84(9):2347-2363
FOR A DISCUSSION OF THIS ISSUE.


Up to 9 treatments can be handled (these could of course also be
populations), with separate hypothesis testing by treatment.  Up to 99
traits can be handled.  The number of observations per treatment can be
as large as the user needs it to be (you do have to tell the program
the max number of observations per treatment/population).

The first module is a user-settings module.  This is the
only place in the program in which the user needs to make changes, indicating
number of treatments, traits and iterations, as well as directories and file
names for input and output.  The input and output directories must be the
same.

NOTE THAT THE INPUT FILE IS SPECIFIED AS COMMA-DELIMITED. YOU CAN EASILY CONVERT
ASCII OR EXCEL FILES TO COMMA-DELIMITED FORMAT USING "SAVE AS" .CSV IN EXCEL.
This avoids problems SAS sometimes has with ordinary excel or txt files, in
which SAS sometimes reads two variable values as one string.

I RECOMMEND RUNNING FIRST WITH NITERAT=5 TO DEBUG FILENAMES, ETC.  IF YOU DO
THIS, DELETE THE OUTPUT FILES, OR CHANGE THE FILENAMES IN THE "SIDEWARD=" AND
"FORWARD=" SPECIFICATIONS OF THE USER SETTINGS MODULE, before running a full
set of iterations.  JUST ABOVE THE USER SETTINGS MODULE, THE PROGRAM SHUTS
OFF THE PRINTING OF NOTES AND SOURCE CODE TO THE LOG.  IF YOU ARE HAVING
TROUBLE GETTING THIS PROGRAM TO RUN, RESTART SAS AND PUT COMMENTS AROUND
THIS LINE OF CODE. NOTE ALSO THAT THE LOG FILE IS REDIRECTED TO A DISC
FILE TO PREVENT BUFFER OVERFLOWS.  IF YOU WANT TO EXAMINE IT, LOOK AT THE
LAST LINE OF THE USER SETTINGS MODULE FOR THE NAME OF THE LOG DISK FILE.

NOTE1: SAS V6.12 (the latest for Mac) running in a 9.2 shell in Mac OS X
is a little buggy, and does not manage to handle SAS work files correctly
when they are iteratively accessed.  It also does not seem to handle random
number generation quite right (some values are drawn more often than expected
at random).  I do not recommend running this in SAS 6.12 in a Mac OS 9.2
shell in Mac OS X.  I have tried this with all versions of OS X through
```

```
10.2.4.  I have not tried native 9.2.  Early work on a Mac running 9.2
shows that SAS v8 in MS Windows has much more informative error messages
with some types of program errors.

NOTE2: The confidence limits on the randomization-based eigenvector
coefficients that are reported by this program, at least with my data,
are quite large, so large  that they are not useful.  They seem to be
correct and to reflect the random construction of trait relationships.
I recommend the use of bootstrap confidence limits for description and
inference associated with these coefficients (a program called BOOTPCA
is currently being debugged and should be available very soon).

It would be fairly easy for a proficient SAS programmer to substitute other
procedures for PROC PRINCOMP, eg for  randomization-based factor or
cluster analyses. I will undoubtedly do some of these myself, and make them
available. A copy of any such modifications by others would be appreciated.
If you use this program, please cite me. Please also tell me if you used it,
and provide feedback, even if you don't publish anything from its use.

Mea culpa for any errors, but you run this program at your own risk and I
take no responsibility for the consequences, although I would be happy
to discuss any problems you encounter.

Steve Tonsor
*/


/*options source notes mprint mlogic mtrace;*/
options nosource nonotes;
/*SAS macro commands establishing the scope of some variables- no user input necessary*/
%GLOBAL NITERAT;
%GLOBAL NTRAITS;
%GLOBAL NSAMPLES;
%GLOBAL SEEDNO;
%let seedno=10908655;

/*1- USER SETTINGS MODULE*/

/*IMPORTANT **** USER MUST SET THESE VARIABLE VALUES **** IMPORTANT*/
%let NITERAT=5;   /*There is no limit to the number of iterations that can be run*/
%let NTRTMT=5;       /*A maximum of 9 treatments can be included in the data set*/
%let ntraits=14;    /* ntraits, the number of traits has a maximum of n=99 */
%let nsamples=280;  /*maximum number of observations per treatment- there is no limit to the
                      number you can have, and the data set does not have to be balanced
                      but this number does have to be bigger than the number of obs per trtmt. */
/*1.1 SET FILEPATH SUBMODULE
  allows easy modification of filepath/directory information between platforms.
  The macro variable "prefix" precedes all file names in the program.  One need
  therefore only change the pathname associated with "prefix" in this module
  in order to change all file names in the program*/
 %GLOBAL prefix;
 %let prefix=C:\Documents and Settings\Administrator\Desktop\Five Carbon\;
/*END 1.1 SET FILEPATH SUBMODULE*/

/*1.2 SET FILENAME SUBMODULE*/
%let direct=FiveC_4PCA.csv;
                   /*specifies the directory and filename in which data is stored*/
                   /* the variables must be in this order:
                       trtmt geno trait1 ... traitn
                     The variables need not have these names. This prog renames them.
                   */
%let sideward=PCACIout;
          /*specifies the directory and filename to which the program will write conf intervals*/
%let forward=RanPCAout;
          /*specifies the directory and filename to which the program will output from PRINCOMP*/

PROC PRINTTO LOG="&prefix.RANDPCA_LOG"; RUN; /*redirects log ouput to prevent overflows*/

/*END 1- USER SETTINGS MODULE */
```

```
/*2 MACRO MAKLIST
  creates a text string: trait1 trait2...traitn, where n=ntraits
*/
%macro maklist;
   %local ENN;
   %do ENN=1 %to &ntraits;
     trait&ENN
   %end;
%mend maklist;
/*END 2 MACRO MAKLIST*/

/*3 MACRO OUTLIST
  creates a text string: trait1 trait2...traitn, where n=ntraits
*/
%macro outlist;
   %local EMM;
   %do EMM=1 %to &ntraits;
     trait&EMM. F10.3
   %end;
%mend outlist;
/*3 END MACRO OUTLIST*/

/*4 MACRO ZDSLST
  creates text string: z1...zN, where zN is the randomized ordering of the Nth trait.
*/
%macro zdslst;
%local zloop;
 %do zloop=1 %to &ntraits;
     z&zloop
 %end;
%mend zdslst;
/*END 4 MACRO ZDSLST*/

/*4 MACRO EVDSLST
  creates text string: rev1 ... revN, where revN is the randomized trait ds for
  the highest-valued treatment.
*/
%macro evdslist;
%local evlup;
 %do evlup=1 %to &ntrtmt;
     rev&evlup.com
 %end;
%mend evdslist;
/*END 4 MACRO EVDSLST*/

/*4B CILIST MACRO
  creates a formatted list for c.i. put statement
*/
%macro cilist;
 %do ciloop=1 %to &ntraits;
   PL&ciloop.LO  F8.2 PL&ciloop.HI  F8.2
 %end;
%mend;
/*END 4B CILIST MACRO*/


/*5 MACRO SIGVAR
  creates variables indicating p<0.01 based on randomizations.
  these are used in macro SIGTEST
*/
 %macro sigvar;
  %local ikk;
  %do ikk=1 %to &ntraits;
   sigtrt&ikk
  %end;
 %mend sigvar;
/*END 5 MACRO SIGVAR*/

/*6 MACRO SIGOUT
```

```
  creates list of variables, each followed by the results of the randomization-
  based significance test
*/
  %macro sigout;
    %local iii;
    %do iii=1 %to &ntraits;
     PL&iii.LO trait&iii PL&iii.HI sigtrt&iii
    %end;
  %mend sigout;
/*END 6 MACRO SIGOUT*/

/*6A DS PCACONC ESTABLISHMENT
  pconc will be used to concatenate output from the multiple runs
  of PROC PRINCOMP.  This establishes the data set with one line of
  missing values.  This line is deleted from the data set at
  a later point.*/
data pcaconc;
  %macro pconc;
    %local iii;
    %do iii=1 %to &ntraits;
     .
    %end;
  %mend pconc;
  put ". . . %pconc";
run;
/*6A DS PCACONC ESTABLISHMENT*/


/*7 INPUT MODULE */
data datset;
infile "&prefix.&direct" delimiter=',' firstobs=2;
input trmt geno %maklist;
 retain lastrt trtmt 0;
  if (lastrt ne trmt) then trtmt = trtmt + 1;
 lastrt = trmt;
proc sort; by trtmt; run;
/* END 7 INPUT MODULE */


/*8 CREATE SORT VARIABLE MACRO
 creates a ds for each trtmt level, containing a new ordered indexing variable
 called SORT.  Sort will be used to match-merge with the random numbers produced
 in module RANDOMID.
*/
%macro sortvar;
%do treat=1 %to &ntrtmt;
 data data&treat; set datset;
 retain sort&treat 0;
  if (trtmt = &treat) then do;
   SORT&treat = SORT&treat + 1;
   SORT = SORT&treat;
  end;
  else delete;
 run;
 proc sort; by trtmt sort;
 run;
%end;
%mend sortvar;
%sortvar;
/*END 8 CREATE SORT VARIABLE MACRO*/

/*9 TRAITDS  MACRO
creates a separate ds for each trait (z) in each trtmt (ev)
*/
%macro traitds;
 %do zloop =1 %to &ntraits;
  %do evloop = 1 %to &ntrtmt;
   data ev&evloop.z&zloop; set data&evloop;
    keep trtmt geno sort trait&zloop;
    if (trtmt ne &evloop) then delete;
   run;
  %end;
 %end;
%mend;
%traitds;
/*END TRAITDS MACRO*/

/*10 RANDPCA MACRO*/
%macro randpca;
/*10.0 ITERATION LOOP */
/*  repeatedly randomizes associations between traits,
  then performs PROC PRINCOMP and outputs to 'output out=' ds.
*/
%do runno = 1 %to &NITERAT;

/*10.0.1 RANDOMID LOOP
creates a ds for each trait in each trtmt, containing the variable SORT,
which is randomly ordered, each obs with a unique value betw 1 and nsamples.
*/
 %do evloop = 1 %to &ntrtmt;
  %do zloop=1 %to &ntraits;
    Proc Plan SEED=&seedno;
     factors sort=&nsamples random/noprint;
     output out= rev&evloop.z&zloop;
    run;
   data _null_;
   /*file "&prefix.seedvals" mod;*/
     seedy=&seedno ;
     call ranuni(seedy,drawn);
     draw=int(drawn*1000000000);
     call symput('seedno',draw);
    /* put drawn draw "&seedno";*/
   run;
  %end;
 %end;
/*END 10.0.1 RANDOMPOT LOOP*/

/*10.0.2 MERGE LOOP*/
/*merges each ds from the TRAITDS macro (ev&evloop.z&zloop) with one of the corresponding
id-ordered data set created in the RANDOMID loop (rev&evloop.z&zloop). It deletes any "extra"
lines (i.e. lines with a sort variable but no trait value), then sorts by the randomly ordered
SORT variable created in the RANDOMID loop. After a merged ds has
been created for each trait, the set of trait ds are merged into a single ds for the treatment,
called revNcom, where N=trtmt.
*/
%do evloop=1 %to &ntrtmt;
 %do zloop=1 %to &ntraits;
   data z&zloop;
     merge ev&evloop.z&zloop rev&evloop.z&zloop;
     if (trait&zloop = .) then delete;
   run;
    proc sort; by sort; run;
 %end;
 data rev&evloop.com;
  merge %zdslst;
 run;
%end;
/* END 10.0.2 MERGE LOOP */

/*10.0.3 TRTMT CONCATENATION MODULE
  puts the data sets from MERGE LOOP for all treatments into a common data set*/
data rancomb; set %evdslist;
runno=&runno;
run;
proc sort; by trtmt; run;
/*proc print; title "rancomb: the combined randomized traits";
  var trtmt sort trait1 trait2 trait3 trait4; run; */
/* END 10.0.3 CONCATENATION MODULE */

/*10.0.4 JOB TRACKER MODULE
   prints the current run number to the output window, enabling the monitoring of analysis
   progress.
```

```
*/
data prinrun; set rancomb;
 retain lincount 0;
 if (lincount ne 0) then delete;
 lincount = lincount + 1;
run;
proc print;
  title "RUN NUMBER";
  var runno;
run;
/*END 10.0.4 JOB TRACKER MODULE */

/* 10.0.5 PRINCOMP MODULE*/
/*performs PCA on each of the treatments separately, then outputs
  stats to a file.
*/
PROC PRINCOMP DATA = rancomb OUTSTAT=RANPCA NOPRINT;
 Var %maklist;
 BY trtmt;
RUN;
/*END 10.0.5 PRINCOMP MODULE*/


/*10.0.6 PCA OUTPUT CONCATENATION MODULE
data pcaconc;
 %if (&runno ='1') %then set ranpca;
 %else
set pcaconc ranpca;
run;
proc sort data=pcaconc; by _type_ trtmt _name_; run; */
/*END 10.0.6 PCA OUTPUT CONCATENATION MODULE */

/*10.0.7 DISK FILE CONCATENATION MODULE*/
data ranpca; set ranpca;
FILE "&prefix.&forward" MOD;
 runno=&runno;
 if ((_TYPE_="MEAN") or (_TYPE_="N") or (_TYPE_="STD")) then delete;
 if (_TYPE_="EIGENVAL") then _NAME_="xxxx";
 if (_TYPE_='.') then delete;
 put runno TRTMT  _TYPE_  _NAME_  %outlist;
run;
/*END 10.0.7 DISK FILE CONCATENATION MODULE*/

%end;
/*END 10.0 ITERATION LOOP */

/*10.1 PCA RESULTS INPUT MODULE */
data pcaconc;
INFILE "&prefix.&forward";
 input runno trtmt _TYPE_ $ _NAME_ $ %maklist;
RUN;
/*END 10.1 PCA RESULTS INPUT MODULE*/

/*10.2 UNIVARIATE MODULE
calculates the 99th percentile for the null hypothesis distribution for each
variable included in the correlation matrix produced by proc princomp.
*/
 proc sort data=pcaconc;  by _TYPE_ TRTMT _NAME_;
 PROC UNIVARIATE plot NOPRINT;
 var  trait1-trait&ntraits;
    output out=bootpctl  PCTLPTS  = 0.5 99.5
                         PCTLPRE  = PL1-PL&ntraits
                         PCTLNAME = LO HI;
    by _TYPE_ TRTMT _NAME_;
 run;
/*END 10.2 UNIVARIATE MODULE*/

%mend;
/* END 10 RANDPCA MACRO */
%randpca;
```

```
/*11 CONFIDENCE INTERVAL ASCII FILE MODULE  */
data bootpctl; set bootpctl;
 retain obsindx 0;
 FILE "&prefix.&sideward";
 If  (obsindx < 1) then put "99 pct confidence intervals from randompca output";
 put _TYPE_ TRTMT _NAME_ %cilist;
 obsindx = obsindx + 1;
run;
/*END 11 CONFIDENCE INTERVAL ASCII FILE MODULE */

/*12 ACTUAL PRINCOMP MODULE
  performs PCA on each of the treatments separately, then outputs
  stats to a file.  This is the actual PCA, to which the
  randomized confidence intervals will be compared.
*/
PROC PRINCOMP DATA = datset OUTSTAT=OUTPCA NOPRINT;
 Var %maklist;
 BY TRTMT;
RUN;
/*END 12 ACTUAL PRINCOMP MODULE*/

/*13 COMPARE TO CI MODULE
   Compares the actual value with the upper and lower C.I. bounds
   to determine significance, then outputs a table of significances.
*/
data outpca; set outpca;
 if ((_TYPE_="MEAN") or (_TYPE_="N") or (_TYPE_="STD")) then delete;
 if (_TYPE_="EIGENVAL") then _NAME_="xxxx";
run;
proc sort data=outpca; by _TYPE_ TRTMT _NAME_; run;
data sistered; merge outpca bootpctl; by _TYPE_ TRTMT _NAME_;
 length %sigvar $ 8;
 %macro sigtest;
   %local jjj;
   %do jjj=1 %to &ntraits;
     if ((PL&jjj.LO > trait&jjj ) or (PL&jjj.HI < trait&jjj )) then sigtrt&jjj = "**";
     else sigtrt&jjj = "..";
   %end;
 %mend sigtest;
 %sigtest;
run;

proc print;
 title "Significance tests based on randomization-generated 99% confidence intervals";
 var _TYPE_ TRTMT _NAME_ %sigout ;
run;
/*END 13 COMPARE TO CI MODULE */
```