```
/*BOOTPCA  STEVE TONSOR  DECEMBER 7, 2004
This program bootstraps relationships among traits by randomly sampling
the actual observations in a data set with replacement.  For a data set with n
observations, n random selections are made, each observation independently chosen
from among the n actual observations. The correlation matrix and principle component
eigenvectors and eigenvalues are produced by PROC PRINCOMP. This routine is repeated
ninterat times (the user sets this value), with the values output to a ASCII disk file.
(You can use this file in your own SAS program to make confidence intervals of any
width you choose.  However, you can also let the macro produce 95% confidence intervals.)
BOOTPCA then uses proc univariate to produce bootstrapped-based 95% confidence
intervals on the estimates of pc eigenvalues, trait correlations, and trait eigenvector
coefficients, which are then printed to a SAS output window.  Note that in the output
window, for k traits, the traits will be named 'trait1' through 'traitk', and m
treatments will be named simply 1 - m.  The eigenvalue confidence intervals will be
listed in m lines, one per treatment.  The k eigenvalues will be listed as 'trait1' through
'traitk', although they are actually eigenvalue 1 through eigenvalue k.
Contact Steve Tonsor with any questions regarding the rather cumbersome output
[tonsor@pitt.edu; 412-624-5491].
PLEASE ACKNOWLEDGE ME IF YOU USE THIS PROGRAM or any derivative of it.
*/
PROC PRINTTO LOG="BOOTPCA_LOG"; RUN;/*You will need to delete this file when done*/

/*THE FOLLOWING LINE OF OPTIONS CAN BE ACTIVATED FOR DEBUGGING IF YOU
  ARE MODIFYING THE PROGRAM*/
/*options source notes mprint mlogic mtrace;*/
/*THE FOLLOWING LINE SHOULD BE ACTIVE WHEN RUNNING THE FULL NUMBER OF ITERATIONS.
  YOU MAY WANT TO INACTIVATE IT AND RUN WITH NITERAT=50 OR SOMETHING FOR A FIRST
  RUN-THROUGH TO SEE THAT EVERYTHING IS WORKING CORRECTLY.  IF YOU DO THIS, MAKE
  SURE YOU DELETE ANY FILES THAT ARE CREATED BEFORE PROCEEDING WITH THE FULL RUN.
  OTHERWISE THE PROGRAM WILL ADD THE NEW OUTPUT TO THE OLD RATHER THAN OVERWRITING
*/
options nosource nonotes; /* <- MOVE COMMENT MARK TO ARROW PREVENT LOG OVERFLOW*/


/*SAS macro commands establishing the scope of some variables- no user input necessary*/
%GLOBAL NITERAT;
%GLOBAL NTRAITS;
%GLOBAL NSAMPLES;
%GLOBAL SEEDNO;
%let seedno=108563;

/*1- USER SETTINGS MODULE*/
/*IMPORTANT **** USER MUST SET THESE VARIABLE VALUES **** IMPORTANT*/
%let NITERAT=1000; /*No limit to the number of iterations that can be run*/
%let NTRTMT=5;    /*Max 9 treatments/populations 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 plus one-
                      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 treatment in the treatment with the
                      highest number of observations */
%let datastrt=2; /*The program ignores the header line(s) and starts reading
                   data at the specified line number in the data input file,
                   So tell it the line number on which to start.*/

/*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 the filepath. This works even between Windows and Mac
  operating systems.*/
 %GLOBAL prefix;
 %let prefix=C:\Documents and Settings\Administrator\Desktop\sjt\Five Carbon\Analyses 12-7-04\;
/*END 1.1 SET-FILEPATH SUBMODULE*/

/*1.2 SET-FILENAME SUBMODULE
  You will need to change at least the "direct" name in this list.  The rest
  can be left as is, but you can change them if you want to use filenames
  that are more meaningful to you.*/
%let direct=ForAmos-DEC6-04;
                   /*specifies the directory and filename in which
                     data is stored*/
                   /* the variables must be in this order:
                       trtmt trait1 ... traitn
                     The variables need not have these names.
                     The program renames them.
                   */
%let sideward=BootPCIout;
          /*specifies the directory and filename to which the program will
            write conf intervals*/
%let forward=BootPCAout;
               /*specifies the directory and filename to which the program will
                 output from PRINCOMP*/
/*END 1.2 SET-FILENAME SUBMODULE*/

/*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*/

/*CRIT VALUE OUTPUT LIST MACRO
  creates a list of upper and lower c.i. variable names
  for the ntraits. This is used in outputting  the critical values to
  the file specified as "sidwards" in the user specifications section.
*/
%macro critout;
```

```
%local zloop;
 %do zloop=1 %to &ntraits;
  PL&zloop.2_5 F9.3  PL&zloop.97_5  F9.3
 %end;
%mend critout;
/*END CRIT VALUE OUTPUT LIST MACRO */


/*4 MACRO EVBOOT DSLST
  creates text string: ev1boot ... evNboot,
  where evNboot is the bootstrapped ds for
  the nth treatment.
  Also defines the treament sample sizes
  (calculated below)as global macro variables
*/
%macro evdslist;
%local evlup;
 %do evlup=1 %to &ntrtmt;
     ev&evlup.boot
 %end;
 %do evlup=1 %to &ntrtmt;
    %global nsamp&evlup;
 %end;
%mend evdslist;
/*END 4 MACRO EVBOOT DSLST*/

/*6 MACRO SIGOUT
  creates list of variables, each followed by
  the upper and lower bootstrapped c.i.s
*/
  %macro sigout;
    %local iii;
    %do iii=1 %to &ntraits;
     trait&iii PL&iii.2_5 PL&iii.97_5
    %end;
  %mend sigout;
/*END 6 MACRO SIGOUT*/

/*7 INPUT MODULE */
data datset;
infile "&prefix.&direct" firstobs=&datastrt delimiter=',';
input trmt $  %maklist;
 length lastrt $ 8;
 retain lastrt 'xuppux';
 retain 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*/

/* BOOTPCA MACRO*/
%macro bootpca;
/*9 ITERATION LOOP
  repeatedly randomly draws an observation number, outputs the obs to a ds,
  then performs PROC PRINCOMP and outputs to an 'output out=' ds.
*/
%do runno = 1 %to &NITERAT;
/*10 BOOTOBS LOOP
  creates a ds for each trtmt, with each observation sampled with
  replacement from the base ds. These are resampled in each run.
*/
%local evloop;
%do evloop = 1 %to &ntrtmt;

 data ev&evloop.boot;
    retain seed -1 i 0;
   do eye = 1 to n;
    rando = ranuni(-1);
    CHOICE = int(rando * n ) + 1;
    set data&evloop point=choice nobs=n;
    i = i + 1;
    output;
    if (i ge n ) then stop;
   end;
  run;
 %end;
/*END BOOTOBS LOOP*/

/*MERGE the bootstrapped datasets from the ntrtmts*/
data bootcomb; set %evdslist;
  if (TRTMT ='') then delete;
run;
proc sort; by trtmt; run;

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

```
/* PRINTOUT MODULE */
data bootpca; set bootpca;
retain firstobs 0;
 runno = &runno;
 if ((_TYPE_="MEAN") or (_TYPE_="N") or (_TYPE_="STD")) then delete;
 if (_TYPE_="EIGENVAL") then do;
    _NAME_="xxxx";
    _TYPE_="EIGNV";
 end;
 if (_TYPE_='.') then delete;
 FILE "&prefix.&forward" mod;
if ((runno = 1) and (firstobs =0)) then do;
 put
 "RANDOMPCA PRINTOUT phenotypic associations randomized by CO2 treatment";
end;
put runno trtmt _TYPE_ _NAME_  %outlist;
firstobs = firstobs + 1;
run;
/* END PRINTOUT MODULE */

%end;


data bootpca ;
INFILE "&prefix.&forward" firstobs=2 missover;
 input runno trtmt  _TYPE_ $ _NAME_ $  %maklist;
RUN;
proc sort; by _type_ trtmt _name_; run;

/* UNIVARIATE MODULE
calculates the 95th percentile for the null hypothesis distribution for each
variable included in the correlation matrix produced by proc princomp.
*/
 PROC UNIVARIATE NOPRINT plot normal PCTLDEF=2;
 var  trait1-trait&ntraits;
    output out=bootpctl  PCTLPTS= 2.5 97.5 PCTLPRE=PL1-PL&ntraits;
    by _TYPE_ trtmt _NAME_;
 run;
/*END UNIVARIATE MODULE*/

%mend;
/* END BOOTPCA MACRO */
%bootpca;

/*CONFIDENCE INTERVAL ASCII FILE MODULE */
data bootpctl; set bootpctl;
 retain obsindx 0;
 FILE "&prefix.&sideward";
 If  (obsindx < 1) then put "95 pct confidence intervals from bootpca output";
 put _TYPE_ trtmt _NAME_  %critout;
 obsindx = obsindx + 1;
run;
/*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
   Outputs the actual values with the upper and lower C.I. bounds.
*/
data outpca; set outpca;
 if ((_TYPE_="MEAN") or (_TYPE_="N") or (_TYPE_="STD")) then delete;
 if (_TYPE_="EIGENVAL") then do;
    _NAME_="xxxx";
    _TYPE_="EIGNV";
 end;
run;
proc sort data=outpca; by _TYPE_ TRTMT _NAME_; run;
data sistered; merge outpca bootpctl; by _TYPE_ TRTMT _NAME_;
 keep _TYPE_ trtmt _NAME_ %sigout;
run;

proc print;
 title "Confidence intervals based on bootstrap-generated 95% confidence intervals";
 var _TYPE_ TRTMT _NAME_ %sigout ;
run;
/*END 13 COMPARE TO CI MODULE */
```