Code for permutation test of eigenvalues of a mixed model estimate of $\gamma$

This SAS/IML code permutes the fitness measurements and then rotates the data based on the original estimate of $\gamma$. The permuted and rotated data is then passed back through a mixed model with the same error structure as that used to obtain the original estimate of $\gamma$, and the F statistics from the original double regression are then compared to the distribution of F values from the permutations. Although this code is for a 4-dimensional $\gamma$ matrix estimated in a mixed model reflecting our experimental design(s), it is straightforward to modify for different numbers of traits and designs with different random effect structures. The %rand_gen macro is adopted from [http://www2.sas.com/proceedings/sugi27/p251-27.pdf](http://www2.sas.com/proceedings/sugi27/p251-27.pdf) modified to sort the dependent variable within each tank within each permutation replicate, so fitness is permuted within each tank.

```sas
data yourdata2 (keep = W x1 x2 x3 x4 tank);
set yourdata1;
%rand_gen(indata=yourdata2,outdata=outrand,
  depvar=W,numreps=1000,seed=43)
proc iml;
    ga = {}; /* Paste gamma estimate here */
    gam = (I(4)+1)#ga; /* doubles the quadratic coefficients */
    vech = eigvec(gam);
    samp = ; /*Paste sample size here */
    Y = j((samp*1001), 5, 0); /*columns in Y should be number of traits + 1*/
    do i = 0 to 1000;
      use outrand;
      read all var {x1} where(replicate = i) into Z1;
      read all var {x2} where(replicate = i) into Z2;
      read all var {x3} where(replicate = i) into Z3;
      read all var {x4} where(replicate = i) into Z4;
      read all var {tank} where(replicate = i) into Z5;
      read all var {W} where(replicate = i) into Z6;
      Z = Z1||Z2||Z3||Z4;
      Y1 = Z*inv(vech');
      Z = Z1||Z2||Z3||Z4;
      Y1 = Z*inv(vech');
      Y2 = Z6||Y1;
      l = (((i-1)*samp)+1);
      h =((i-1)*samp)+samp;
      ind = {l h};
      Y[(((i)*samp)+1):(((i)*samp)+samp), 1:5] = Y2;
    end;
    create ca from Y[colname={'W' 'can1' 'can2' 'can3' 'can4'}];
    append from Y;
    close ca;
    data outrand2 (keep = tank replicate);
set outrand;
```

data can;
merge ca outrand2;
ods select none;
run;
ods output Tests3 = pva(keep = replicate effect FValue);

proc glimmix data = can method = MSPL ;
by replicate;
class tank;
model W = can1|can1 can2|can2 can3|can3 can4|can4;
random can1 can2 can3 can4 / type = chol sub = tank;
run;
ods select all;
data pval;
set pva;
if effect = "can1" then delete;
if effect = "can2" then delete;
if effect = "can3" then delete;
if effect = "can4" then delete;
data can1;
set pval;
if effect = "can2*can2" then delete;
if effect = "can3*can3" then delete;
if effect = "can4*can4" then delete;
data can2;
set pval;
if effect = "can1*can1" then delete;
if effect = "can3*can3" then delete;
if effect = "can4*can4" then delete;
data can3;
set pval;
if effect = "can1*can1" then delete;
if effect = "can2*can2" then delete;
if effect = "can4*can4" then delete;
data can4;
set pval;
if effect = "can1*can1" then delete;
if effect = "can2*can2" then delete;
if effect = "can3*can3" then delete;
proc iml;
use can1;
read all var {FValue} into c1;
use can2;
read all var {FValue} into c2;
use can3;
read all var {FValue} into c3;
use can4;
read all var {FValue} into c4;
prop = j((nrow(c1)-1), 4, 0);
do i = 0 to 0;
est1 = c1[(i+1), 1];
est2 = c2[(i+1), 1];
est3 = c3[(i+1), 1];
est4 = c4[(i+1), 1];
end;
do i = 1 to (nrow(c1)-1);
if c1[(i+1)] > est1 then prop[i, 1] = 1;
else prop[i, 1] = 0;
if c2[(i+1)] > est2 then prop[i, 2] = 1;
else prop[i, 2] = 0;
if c3[(i+1)] > est3 then prop[i, 3] = 1;
else prop[i, 3] = 0;
if c4[(i+1)] > est4 then prop[i, 4] = 1;
else prop[i, 4] = 0;
end;
canonical_1 = prop[+ , 1]/nrow(prop);
canonical_2 = prop[+ , 2]/nrow(prop);
canonical_3 = prop[+ , 3]/nrow(prop);
canonical_4 = prop[+ , 4]/nrow(prop);
print canonical_1 canonical_2 canonical_3 canonical_4;
run;