%ArrayPerm: A SAS® Macro for Permutation Analysis of Microarray Data
Deqing Pei, M.S., Wei Liu, M.S., Cheng Cheng, Ph.D.
St. Jude Children’s Research Hospital, Memphis, TN

ABSTRACT
Microarray has become a common tool for identifying genes of interest that are differentially expressed under different biological conditions. Software packages such as Spotfire, GeneSpring, and Xcluster are commonly used in research laboratories. These packages provide a set of analysis tools for accessing, visualizing, normalizing and filtering data with modest learning curve for biologists without statistical background. However, wide applications of DNA microarray technology in gene discovery, disease diagnosis, pharmacogenomics etc. require more sophisticated statistical analysis methods beyond simply looking for genes up- or down-regulated, thereby requiring solid data modeling on the basis of valid statistical methods. When the expression data or their transformations cannot be assumed to distribute normally, or the investigator wants to assess the reliability of gene-selection procedures, permutation tests are often desired. Software packages mentioned above do not have flexibility or capability for implementing permutation tests under sophisticated models such as general linear models, generalized linear models, GEE, Cox regression models, etc. Here, we present a SAS macro to facilitate permutation analysis on a broad spectrum of statistical models that can be chosen according to specific objectives of microarray experiment. This macro is a very flexible tool for one to implement permutation tests quickly and easily. It can provide such flexibility that every step of the analysis can be fine-tuned by appropriate programming.

INTRODUCTION
Microarray data are summarized in a matrix format where the genes are in rows and different individuals or subjects correspond to columns. Contrary to traditional statistical set-ups, the number of subjects is much smaller than the number of genes. The dimension and complexity of raw gene expression data obtained by oligonucleotide chips and spotted arrays create challenging data analysis problems, especially when permutation tests under sophisticated statistical models are desired. For example, the Affymetrix oligonucleotide human array U133A contains approximately 22,000 probesets, which means that 22000 separate statistical tests are needed using whatever chosen procedure in each round of permutation. For 1000 permutations, this implies that we need to run 22000 x 1000 tests with any chosen SAS procedure (e.g., PROC MIXED, PROC PHREG). To perform permutation tests on such complicated data within a reasonable amount of time requires plenty of computing resource, such as high CPU speed and large memory. The macro described in this paper provides a quick and flexible way to produce permutation p values under any chosen statistical model. Methods to maximize the performance of the SAS system via batch mode submission and to save results in a defined SAS library are also presented.

ARRAY SYNTAX
%ArrayPerm(dataset=, seed=, nperm=, varlist=%str(), identify=, response=, ProcName=, stmts=%str(), ODStable=, TestStat=, Pvalue=)

MACRO PARAMETERS
varlist= specifies a list of variables that you want to keep for future analyses. For example, varlist =%str (id cmsh2 signal lsignal spot), a string of variable names separated by a space
dataset= specifies the name of the macro input dataset
seed= a seed number for the random number generator, a prime number is recommended
nperm= specifies the number of permutations to perform
identify= specifies the name of variable that will be used to identify each unique individual probeset
(gene spot). It is always a good idea to create numeric identifier instead of using character string of probeset name available in the data set.

**response=** specifies the name of variable that will be used as response variable in the model, could be original expression signal, log-transformed expression signal or transformed signal by any other method.

**ProcName=** provides the name of SAS procedure that you will call for the analysis, such as PROC MIXED, PROC PHREG or whatever procedures chosen based on the specific aim and design of the microarray experiment.

**stmts=** provides model statement appropriate for the procedure that you are calling, e.g. stmts=%str(model time*censor(0,2,3)=lsignal;) for PROC PHREG.

**ODStable=** specifies the name of ODS table where you can find values of test statistics. For two samples or multiple comparisons in the mixed model procedure, the ODS table name is `test3`; for survival type analysis, you can call PROC PHREG and the ODS table name is `ParameterEstimates`. The easiest way to find out the names of ODS tables is to use the ods trace output option. This option displays the name and other information associated with each piece of output as it is produced.

**TestStat=** specifies the variable name of test statistics in ODS table. For example, two samples or multiple comparisons in the mixed model procedure, the variable name in the ODS table `test3` is `fvalue`; for survival type analysis, the variable name in the ODS table `ParameterEstimates` is `chisq`.

**Pvalue=** specifies the variable name of test p value in ODS table. For two samples or multiple comparisons in the mixed model procedure, the variable name in the ODS table `test3` is `probf`; for survival type analysis, the variable name in the ODS table `ParameterEstimates` is `probchisq`.

**TECHNICAL DETAILS**
Permutation tests are generally carried out in following five steps:

- **a)** Analyze the problem: identify the null (NULL) and alternative (ALT) hypotheses of interest.
- **b)** Choose appropriate test statistic (TS) that discriminates between the NULL and the ALT.
- **c)** Compute the TS for the original labeling of the observations.
- **d)** Full enumeration of all possible permutations quickly becomes infeasible as sample sizes increases because the number of possible sample combinations becomes very large. Therefore, this macro performs permutation based on randomly rearranging the observations for a pre-specified number of times (e.g., 1,000). The samples’ class labels are randomly reassigned and the TS for the rearranged data are recomputed. This process is repeated for a desired number of permutations.
- **e)** Make a decision, reject or not to reject the NULL based on the permutation distribution of TS and the extremity of the original TS (from the un-permuted data) in this distribution.

**STEP 1: ANALYZE THE PROBLEM**
The NULL and ALT hypotheses need to be identified based on each specific aim of the microarray experiment. Then an appropriate test statistic should be chosen.

**STEP 2: COMPUTE THE TS FOR THE ORIGINAL LABELING OF THE OBSERVATIONS**

```sas
DATA OLD (KEEP=&VARLIST) NEW (KEEP=&IDENTIFY &RESPONSE);
SET &DATASET; OUTPUT OLD;
OUTPUT NEW;
```

In order to generate permuted data set for future analysis, we need to create a new data set that contains only the variables that will be used to identify each unique individual probeset and the variables that will be used as response variables (original expression signal or log-transformed expression signal or transformed expression signal by another method) in the model. The original dataset that is used to compute the test statistic for the un-permuted observations also need to be saved.
The ODS statement turns off the standard “line printer” ODS destination.

The ODS statement output desired ODS table to the SAS data set.

The ODS statement turns on the standard “line printer” ODS destination.

For the procedure chosen according to the specific aim of experiment, ODS table name that contains values of test statistics need to be identified and output to a SAS dataset named as OUTOLD. The easiest way to find out the names of ODS tables is to use the ODS trace output option. This option displays the name and other information associated with each piece of output as it is produced.

Original variables representing the test statistic in the output data set OUTOLD are renamed for future convenience. A new variable named as count with start value set to zero is created in this step. This variable will be used to keep tracking the number of times that test statistic from rearranged data set is more extreme or equal to the observed test statistics in the future.

Now randomly rearrange (permute) the samples’ class labels and re-compute the TS from the permuted. To do this, we need to generate random numbers and rearrangements first, then generate the permuted data set by sorting the data containing only the probeset (identifier), the response variables (expression level data) and the random numbers. Last, we merge this sorted data back to the original data by identifier, which will produce the permuted data set.
In each round of permutation, a new permutated data will be created and overwrite the old one. The test statistics will be computed for each permuted dataset. Whenever the permuted test statistic is more extreme or equal to the observed test statistics, the variable COUNT will increase by 1; this way, we can keep track of the number of times that the permuted test statistic is more significant than the observed test statistics. Test statistics computed from each permuted dataset will be output into a new data set. The dataset in which the original test statistic was saved will be merged with this newly created data set. We compare the original test statistics with the permuted TS and increase COUNT by 1 if the permuted TS is more extreme than the original TS. The count will be saved in COUNT variable.

**STEP 4: COMPUTE PERMUTATION P VALUE**

```
*DATA D.PERMPVALUE;
SET COUNT1;
P_VALUE = (COUNT) / (&NPERM);
RUN; QUIT;
```

* D is the libname of the project you need to create for each specific project, you can choose whatever name you prefer to reflect project. The name D is just my preference. PERMPVALUE is the name of dataset where permutation p values were saved

Permutation p value will be defined as proportion of number of times that the test statistic from the permuted datasets are more extreme or equal to the observed test statistic in the total number of permutations.

**STEP 5: BATCH MODE SUBMISSION TO MAXIMIZE THE PERFORMANCE OF THE SAS SYSTEM**

i. You will have to format your microarray data into a structure that is suitable for the chosen model procedure based on specific aim of microarray experiment, sort data by identifier of each individual probeset.

ii. Set up library where all temporary and final results will be stored. It is always a good idea to set up individual SAS library for each individual project where interim or final results will be stored.

iii. Provide choice for all parameters required in the Macro based on your dataset. Test-run your program on smaller dataset (for an example, using 10 probesets, 10 permutation runs), make sure that program runs smoothly without any error message. Then fine-tune the program, specify the final data set and the number of permutations that you want to run.

iv. Save the program, close SAS window. From Windows Explorer, find the SAS program saved in project folder and right click on the chosen fine-tuned program. You can choose batch mode submission from the Windows Explorer. All results (notes in log window will be saved in a file with .log extension name and all results will be saved in the specified project library).

v. Have fun with browsing and playing result in the SAS window, results will be saved in the SAS library defined for each specific project. Select a set of significant genes using permutation p value for future analyses such as cluster analysis, etc.
STORE COMPILED MACRO

\h\LIBNAME MYMACROS "C:\DEQING\PEI2003\ARRAYPERMUTATION\SAS PROGRAM";
OPTIONS MSTORED SASMSTORE=MYMACROS;
SAS MACRO CODE

\hSpecify the FILEPATH where compiled MACRO will reside

TO DISPLAY THE STORED-COMPILED MACROS IN A LIBRARY

PROC CATALOG CAT=MYMACROS.SASMACR;
CONTENTS;
RUN;

SAMPLE MRACRO CALL #1: PROC MIXED MODEL
\%ArrayPerm(dataset=test,
  seed=456,
  nperm=1000,
  varlist=%str(id cmsh2 signal lsignal spot),
  identify=spot,
  response=signal,
  ProcName=mixed,
  stmts=%str(class cmsh2;
    model signal=cmsh2; ),
  ODStable=tests3,
  TestStat=fvalue,
  Pvalue=probf)

SAMPLE MRACRO CALL #2: PROC PHREG PROCEDURE
\%ArrayPerm(dataset=pertest2,
  seed=578,
  nperm=1000,
  varlist=%str(id time censor lsignal spot),
  identify=spot,
  response=lsignal,
  ProcName=phreg,
  stmts=%str(model time*censor(0,2,3)=lsignal; ),
  ODStable=ParameterEstimates,
  TestStat=chisq,
  Pvalue=probchisq)

REFERENCES
Tom O'Gorman, Northern Illinois University, A SAS macro for performing the adaptive weighted least squares test for a subset of a model

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the authors at:

Deqing Pei  
Department of Biostatistics  
St. Jude Children's Research Hospital  
332 North Lauderdale St.  
Memphis, TN 38105-2794  
Tel: 901-495-3076  
Fax: 901-522-8843  

Wei Liu  
Department of Biostatistics  
St. Jude Children's Research Hospital  
332 North Lauderdale St.  
Memphis, TN 38105-2794  
Tel: 901-495-3373  
Fax: 901-522-8843

Cheng Cheng, Ph.D.  
Department of Biostatistics  
St. Jude Children's Research Hospital  
332 North Lauderdale St.  
Memphis, TN 38105-2794  
Tel: 901-495-2935  
Fax: 901-522-8843

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.