October 13, 2002

Permax Computations

The permax package is designed to perform permutation analysis of data from gene expression microarray experiments or other types of high dimensional data. The basic approach has been briefly described in Mutter *et. al.* (2001) and in Ibrahim *et. al.* (2002). The arguments and usage of the functions are described in the help files in the package. This document gives additional details on the computations, and describes features for analyzing clustered and stratified data, available beginning with permax version 2.1. "Clustered" here refers to an experimental situation where there is correlation among arrays within groups. First methods for independent arrays will be described followed by extensions for clustered data.

Data Structure for Independent Arrays

Permax is designed for data, such as that arising in gene expression microarray experiments, where a large number of different attributes are measured on each experimental unit. For independent arrays, Let X_{ij} be the value of attribute i on unit j, $i = 1, \ldots, I$, $j = 1, \ldots, n$. The $I \times n$ array (X_{ij}) corresponds to the `data` argument to the `permax`, `permsep` and `permcor` functions. The different units are assumed to be independent, except as described in the Clustered and Stratified Data Section, below. When two groups of units are being compared, eg from different tissue types or with different treatments, the units $1, \ldots, n_1$ are assumed to be from group 1 and the units $n_1 + 1, \ldots, n$ from group 2.

Test Statistics

There are 3 main types of analyses performed by the permax package. The `permax` function uses t statistics to compare two groups. The `permsep` function analyzes complete separation between two groups, which is defined as all values from one group being larger than all values from the other. The `permcor` function examines correlations between the gene expression levels and a continuous phenotype (covariate). The actual values, log values, or ranks can be used in the statistics.

The basic test used in `permax` for comparing attribute i between two groups is the ordinary pooled variance t statistic

$$T_i = \frac{\overline{X}_{1i} - \overline{X}_{2i}}{\{(1/n_1 + 1/n_2)(S_{1i} + S_{2i})/(n_1 + n_2 - 2)\}^{1/2}},$$

where $n_2 = n - n_1$, $\overline{X}_{1i} = \sum_{j=1}^{n_1} X_{ij}/n_1$, $\overline{X}_{2i} = \sum_{j=n_1+1}^{n} X_{ij}/n_2$, $S_{1i} = \sum_{j=1}^{n_1}(X_{ij} - \overline{X}_{1i})^2$. and $S_{2i} = \sum_{j=n_1+1}^{n}(X_{ij} - \overline{X}_{2i})^2$.

It is a well known result that T_i is a monotone function of the statistic $n_1\overline{X}_{1i}$, where the function also depends on the values of n_1, n_2, $U_i = \sum_{j=1}^{n} X_{ij}$, and $V_i = \sum_{j=1}^{n} X_{ij}^2$. Since these quantities are all fixed for any permutation of the units between the two groups, it is only necessary to evaluate the statistic $n_1\overline{X}_{1i}$ when evaluating the permutation distribution of T_i.

With two groups, the Wilcoxon test for equality of the distributions for attribute i is based on the statistic

$$W_i = \sum_{j=1}^{n_1} \sum_{l=n_1+1}^{n} \left(I\{X_{ij} > X_{il}\} + \frac{1}{2}\, I\{X_{ij} = X_{il}\} \right). \qquad (1)$$

Let c_{ij} be the rank of X_{ij} in $\{X_{i1}, \ldots, X_{in}\}$, using the average of the ranks for tied observations. Then $W_i = \sum_{j=1}^{n_1} c_{ij} - n_1(n_1+1)/2$ (see eg Bickel and Doksum, 1977, p. 348–9). Comparing this with the results for the t statistic above, it follows that the permutation test based on (1) is equivalent to the permutation t test computed from the ranks of the data, and that only $\sum_{j=1}^{n_1} c_{ij}$ needs to be evaluated on each permutation. Thus the same computational algorithm can be used for both t tests and for Wilcoxon tests, with the data values replaced by ranks in the latter case.

For the `permcor` function, with a single group of n units, and a covariate (phenotype) taking values $Z_1, \ldots, Z_n$, the statistic used is the estimated linear correlation between Z_j and X_{ij},

$$R_i = \frac{\sum_j (X_{ij} - \overline{X}_i)(Z_j - \overline{Z})}{\left(\sum_j (X_{ij} - \overline{X}_i)^2 \sum_j (Z_j - \overline{Z})^2\right)^{1/2}},$$

where $\overline{X}_i = U_i/n$ and $\overline{Z} = \sum_j Z_j/n$. The standard t statistic for testing for no association between the ith attribute and Z is

$$R_i(n-2)^{1/2}/(1-R_i^2)^{1/2}.$$

Since this statistic is a monotone function of R_i, it is only necessary to evaluate R_i when evaluating the permutation distribution of this test.

Simultaneous Inference

With a high-dimensional multivariate response on each unit, and with a separate statistic computed for each attribute, there are a large number of comparisons being made. Permax uses the free step-down approach of Westfall and Young (1993, Section 2.6) to compute simultaneous p-values that control the overall (or familywise) error rate. Suppose $T_1, \ldots, T_I$ are the observed values of the individual test statistics, with the attributes ordered so that

$T_1 \leq \cdots \leq T_I$, and let T_i^* be the value of statistic for the ith attribute computed from an independent sample. In this approach, the adjusted upper-tail p-values are defined by

$$\tilde{p}_I = P\left(\max_{l \leq I} T_l^* \geq T_I\right)$$

and

$$\tilde{p}_i = \max\left\{\tilde{p}_I, \ldots, \tilde{p}_{i+1}, P\left(\max_{l \leq i} T_l^* \geq T_i\right)\right\}, \quad i < I,$$

where the probabilities are computed under the null hypothesis, conditional on the observed data. Similarly, the lower tail p-values are defined by

$$\tilde{p}_1 = P\left(\min_{l \leq I} T_l^* \leq T_1\right)$$

and

$$\tilde{p}_i = \max\left\{\tilde{p}_1, \ldots, \tilde{p}_{i-1}, P\left(\min_{l \geq i} T_l^* \geq T_i\right)\right\}, \quad i > 1.$$

The same formulas, with the T_i replaced by the appropriate statistics, can be used to define adjusted p-values for the other analyses considered here. In Section 2.8 of Westfall and Young (1993), it is shown that this approach does control the familywise error rate in the strong sense.

Westfall and Young (1993) focus on defining adjusted p-values from individual p-values, rather than from individual test statistics. For the computations using the individual statistics to be meaningful for all attributes, the statistics need to be defined on a common scale. The standardized t statistic above has a distribution that is invariant to location and scale shifts. However, since the marginal distribution of the different attributes could still be quite different, this does not guarantee complete comparability among the different attributes. This problem would not be corrected by using the t distribution to compute individual p-values from the t statistics and using these in place of the statistics in the formulas, since the underlying data may not be normally distributed. This issue is not addressed further here, but should be considered in any practical applications.

In the permax package, the adjusted p-values above are approximated using the null permutation distribution. The data are first standardized to simplify the computations. For the standard t statistic, T_i is a monotone function of the corresponding $n_1\overline{X}_{1i}$, but the function also depends on the the values U_i and V_i, which in general could be different for different attributes. The values for each attribute are first standardized so that $U_i = 0$ and $V_i/(n-1) = 1$ for all i. Then $n_1\overline{X}_{1i}$ is computed from the standardized values. The permutation test based on these quantities is equivalent to that obtained from the T_i, even with regard to the distribution of the maximum over a set of attributes.

For attribute i, the upper tail simultaneous p-value is computed as

$$\tilde{p}_i = \max\left\{\tilde{p}_I, \ldots, \tilde{p}_{i+1}, \sum_p I(n_1\overline{X}_{1i} \leq \max_{l \leq i} n_1\overline{X}_{1l}^{(p)})/P\right\}, \tag{2}$$

where P is the number of permutations and $n_1\overline{X}_{1i}^{(p)}$ is the value of the statistic for the pth permutation. Again here the attributes have been ordered so that $\overline{X}_{11} \leq \cdots \leq \overline{X}_{1I}$. (Equation (2) is the exact permutation adjusted p-value if all permutations are enumerated, and an estimate if random permutations are generated.) Similarly, the lower tail simultaneous p-value is computed as

$$\tilde{p}_i = \max\left\{\tilde{p}_1, \ldots, \tilde{p}_{i-1}, \sum_p I(n_1\overline{X}_{1i} \geq \min_{l \geq i} n_1\overline{X}_{1l}^{(p)})/P\right\}.$$

Prior to version 2.2, the permax package used the permutation distributions of $\max_i T_i$ and $\min_i T_i$ to determine the simultaneous p-values for all genes. The Westfall-Young method gives the same p-values for the most significant genes, but often gives smaller p-values for other genes.

If the null of no difference is rejected for those genes for which $\tilde{p}_i \leq \alpha$, then the familywise error rate is controlled at the level α. An interpretation of this is that if the experiment is repeated with the same analysis procedure, then the chance of there being any false positives (genes with no true difference declared as significant) is $\leq \alpha$. Benjamini and Hochberg (1995) also note that the false discovery rate (proportion of the genes declared to be significant that are false positives) is $\leq$ the familywise error rate, so only declaring genes to be significant if the p-value is $\leq \alpha$ also guarantees that the false discovery rate is $\leq \alpha$. A quantity that can be used to provide additional information about possible false positives is the average number of positives (genes more extreme than a specified critical value) in the null permutation distribution. This quantity estimates the expected number of positives at a given cutoff level under the global null hypothesis. If it is small, then it is likely that there are few false positives in the actual sample at that cutoff. This quantity is computed by the `permax` function, along with the proportion of permutations with as many or more positives, and the proportion of permutations with any positives.

Bonferroni procedures are also often used to control familywise error rates. Let $\hat{p}_i$ be the individual permutation p-values with no correction for multiple comparisons. The single step Bonferroni procedure rejects the null for those genes for which $\hat{p}_i \leq \alpha/I$. It can be improved by using multiple step approaches. Hochberg (1988) gave a step-up approach where the genes are first ordered so that $\hat{p}_{(1)} \leq \cdots \leq \hat{p}_{(I)}$. If i^* is the largest i for which $\hat{p}_{(i)} \leq \alpha/(I - i + 1)$, then Hochberg's procedure rejects the null for all genes with $\hat{p}_i \leq \hat{p}_{i^*}$. (This is termed a 'step-up' procedure because it starts with the least significant gene and proceeds to check genes one by one until a significant gene is found, and then declares all more significant genes to be significant.) The individual one-tailed p-values in both directions are also returned by the `permax` and `permcor` functions.

For the Wilcoxon test, after applying the rank transformation to each set $\{X_{i1}, \ldots, X_{in}\}$, the ranks are centered to sum to 0, but they are not re-scaled, since the rank transformation has already converted the data to a common scale. (A minor note: with ties, the value of V_i computed from the ranks varies slightly with the number and location of the ties, but basing the comparison on W_i as defined above may still be more appropriate than using a standardized version.)

The correlation statistics R_i are already centered and transformed to a common scale. The upper tail and lower tail adjusted p-values are computed as described above, with R_i in place of T_i. To facilitate the calculations, the data are first standardized so that $U_i = 0 = \sum_j Z_j$ and $V_i = 1 = \sum_j Z_j^2$. Then $R_i = \sum_j X_{ij} Z_j$. (Note: this same standardization is still applied even if ranks are used for the X_{ij} in the correlation statistic, to keep the correlation statistics between –1 and 1.)

In computing the permutation distribution for the two group problem, permutations of units within the two groups have no effect on the values of the statistics, so only the

$$\binom{n}{n_1}$$

distinct ways of choosing the n_1 units in group 1 need to be considered. Each of these occurs the same number of times ($n_1!\, n_2!$) in the full permutation distribution. When enumerating all possible combinations, these combinations are systematically generated. For random permutations, the combinations are randomly generated, and the same combination may occur more than once. Note that each permutation is applied to the experimental units as a whole (that is, all the attributes are permuted together). In this way the correlation structure among the attributes is maintained. If attributes were permuted separately, then the correlation structure among the attributes would be lost.

In the permcor function, since it is assumed that the covariate values could be different for every unit, all $n!$ permutations would be generated when the full permutation distribution is used. This is only feasible if n is fairly small. In generating the permutations, the values of the Z_j are permuted, while the X_{ij} are kept fixed.

The uniform generator of Wichmann and Hill (1982) is used to generate random numbers for the random combination and permutation algorithms. The main deficiency of this algorithm is that it has a relatively short period. However, since a large number of calculations are performed for each permutation or combination, it will not be feasible to perform a very large number of permutations, so the period of almost 7×10^{12} should be more than adequate.

Complete Separation

In the two group problem, the permsep function computes a p-value based on the number of attributes with complete separation. Complete separation occurs in the ith attribute if either

$$\max_{1 \le j \le n_1} X_{ij} < \min_{n_1+1 \le j \le n} X_{ij}$$

or

$$\min_{1\leq j\leq n_1} X_{ij} > \max_{n_1+1\leq j\leq n} X_{ij}.$$

The `permsep` function counts the number of attributes with complete separation. Call this value N_s. Let $N_s^{(p)}$ be the number of attributes with complete separation in the pth permutation. The p-value is given by $\sum_p I(N_s \leq N_s^{(p)})/P$. The average number of attributes with complete separation, $\sum_p N_s^{(p)}/P$, and the proportion of permutations with any attributes with complete separation, $\sum_p I(0 < N_s^{(p)})/P$, are also reported.

Clustered and Stratified Data

Clustered or stratified data refers to the setting where experimental units can be grouped into clusters or blocks. Units from the same cluster may be correlated, but units from different blocks are independent. For example, if a microarray experiment were run on tissue taken from mice, and several sets of tissue were collected from each mouse and different treatments applied to the different tissue sets from each mouse, then the sets of outcomes from each mouse would be a block or cluster. In this setting, let X_{ijk} be the value for attribute i on unit j from cluster k, $i = 1,\ldots,I$, $j = 1,\ldots,m_k$, $k = 1,\ldots,K$.

With clustered data, the group membership in the two group problem (for `permax`) or the covariate values (for `permcor`) can be defined at either the cluster level or at the level of the individual unit. In the former case, each unit in a cluster belongs to the same group, or has the same covariate value, while in the latter case different units within a cluster may belong to to different groups or have different covariate values. In a split-plot experiment, the first would correspond to whole plot effects and the second to sub-plot effects.

In the functions in the permax package, the first case can be analyzed by using the `cluster` argument to specify the cluster membership, and specifying `permute.cluster=TRUE`. In this case, statistics are computed as described above, but only whole clusters are permuted. This maintains the within cluster dependencies in the permuted data sets. In the `permax` function, since the number of clusters in each group remains fixed, the number of units in the groups will vary, if the cluster sizes are not equal. Similarly in `permcor`, since the covariate values are permuted among the clusters, the number of individual units with a given value will vary if the cluster sizes are not equal. This means the phenotype values need to be re-standardized for each permutation. Note: In many situations permuting whole clusters will produce the same results as an analysis where first each cluster is replaced with a single set of summary statistics (eg the cluster means $\sum_j X_{ijk}/m_k$), and then the cluster summary statistics are analyzed as a sets of values from independent units.

For the second case, with unit level effects, permutations are only applied within clusters. Again this maintains the correlation structure in the clusters. There are two options for computing the statistics. In the first (unstratified), the data are standardized and the test statistics are computed exactly as described above. The only difference in this case is that the reference set used in the permutation distribution is restricted to within cluster permutations. Consider, for example, a classic balanced randomized block experiment. Here the permutation distribution can be thought of as permuting the treatment labels among the units. By only permuting treatment labels within blocks, treatment balance will be maintained within blocks for all permutations considered, while the unrestricted permutation distribution would include permutations where treatment assignments are unbalanced within blocks, and hence where the treatment effects in the permuted sample would be confounded with block effects. (Note: the unstratified tests may not be appropriate in general for unbalanced, clustered data.)

The second option is to use stratified tests. In this case, statistics are computed separately within clusters and then combined over clusters. For the stratified tests in `permax`, the data are first standardized within each cluster. That is, defining $U_{ik} = \sum_{j=1}^{m_k} X_{ijk}$ and $V_{ik} = \sum_{j=1}^{m_k} X_{ijk}^2$, in the two group problem the data are standardized so that $U_{ik} = 0$ and $V_{ik}/(m_k - 1) = 1$. When ranks are used, values are ranked separately within each cluster, and the ranks centered separately within each cluster.

In the two group problem, define $\overline{X}_{lik}$ to be the average of the standardized attribute i values in group l within cluster k. The stratified statistic for attribute i is

$$\sum_k \omega_k(\overline{X}_{1ik} - \overline{X}_{2ik}) = \sum_k \omega_k(1 + n_{1k}/n_{2k})\overline{X}_{1ik} = \sum_k \omega_k \frac{m_k}{n_{1k}n_{2k}} \sum_{j=1}^{n_{1k}} X_{ijk},$$

where n_{1k} is the number of subjects in group 1 in cluster k, $n_{2k} = m_k - n_{1k}$, and the ω_k are user supplied cluster weights (argument `weights` in `permax`), which default to $1/K$. For the correlation analysis, the stratified statistic is

$$\sum_k \omega_k R_{ik},$$

where R_{ik} is the estimated correlation for attribute i within cluster k. Here the user supplied weights ω_k (argument `weights` in `permcor`) are renormalized to sum to 1, and again default to $1/K$.

Units are again only permuted within clusters in the stratified case. P-values are computed from the permuted data sets as described earlier.

In the special case of the two group problem with paired data, where each cluster consists of two units, one from each group, the Wilcoxon signed-rank statistic can be used by specifying `signed.rank=TRUE` in the `permax` function. In this case, the differences $X_{i1k} - X_{i2k}$ are first calculated (using log values if `logs=TRUE`). The absolute values of the differences are then ranked, and then the sign of the difference attached to the rank. Pairs with $X_{i1k} - X_{i2k} = 0$ are given a signed rank of 0. The statistic is computed as the sum of the ranks over all pairs. This is a monotone function of the sum of the ranks where the difference is positive, which is also sometimes used as the test statistic. The permutation distribution is obtained by permuting group membership within pairs, which has the effect of flipping the signs on the signed ranks, but leaving the ranks of the absolute differences unchanged. The calculations are performed as above for the case with `cluster` specified and `stratify=FALSE`, with an appropriate set of pseudo-data. In particular, the absolute differences are ranked and the signed ranks created as defined above. The signed rank is put into the group 1 observation and the negative of the signed rank is put into the group 2 observation. Permuting the pair membership and computing the sum of the group 1 values for each permutation then generates the permutation distribution of the signed-rank statistic. The difference between this test and the test obtained with `ranks=TRUE`, `stratify=FALSE` and `signed.rank=FALSE` is that in the latter case the data values are ranked and differences of the ranks taken in each pair, while in the signed-rank test differences of the values are taken and then ranked. Also, if `ranks=TRUE` and `stratify=TRUE` is specified with paired data, then the resulting test is equivalent to the sign test.

The `permsep` function also has the same options described above for clustered data. If the `cluster` argument is given, either whole cluster permutations or within cluster permutations are used, depending on the value of `permute.cluster`. If `stratify=FALSE`, then complete separation is defined using all values, while if `stratify=TRUE`, then complete separation is only examined within clusters. In the latter case, values from different clusters can overlap, as long as the groups are separated within each cluster.

Custom Statistics

In principle, other statistics can be used by replacing the Fortran functions `tsum` (for the two group problem) and `dip` (for correlation tests) with new functions to compute the desired statistics. However, considerable care is needed as the data have been pre-processed prior to calling these functions. Also, note that each is called once for each attribute in the data set on the original data, and once for each attribute on each permuted data set.

The function `tsum` is called from subroutine `ptn`. The specification for `tsum` is

```
function tsum(ng,d,n,ig1,istrt,nclust,mclust,mct1,wght)
integer ng,n,ig1(ng),istrt,nclust,mclust(nclust),mct1(nclust)
real tsum,d(n,*),wght(nclust)
```

where the values passed from `ptn` are

`ng` number of columns (units) in group 1 (n_1 above)
`d` `d(1,j)` $= X_{ij}$ for the current `i` in `ptn` (not a typo: `d(1,j)`, not `d(i,j)`); X_{ij} have been standardized as described above; columns of `d` have been sorted on cluster labels if `cluster` was specified
`n` # attributes (I above) = # rows in `d`
`ig1` `ig1(j)` = column # in `d` of the `j`th unit in group 1
`istrt` =1 if tests should be stratified
`nclust` # clusters (=1 if `cluster=NULL`)
`mclust` `mclust(k)` = # units in cluster k
`mct1` `mct1(k)` = # units in group 1 in cluster k
`wght` the weights for the stratified test; actually the user supplied weights ω_k multiplied by $m_k/(n_{1k}n_{2k})$

The function `dip` is called from the subroutine `ptcor`. The specification of `dip` is

```
function dip(ng,x,icx,y,icy,nclust,mclust,istrt,wght)
integer ng,icx,icy,nclust,mclust(nclust),istrt
real dip,x(ng*icx),y(ng*icy),wght(nclust)
```

where the values passed from `ptcor` are

`ng` number of columns (units) in the data set (n above)
`x` `x(1+(j-1)*icx)` $= X_{ij}$ for the current `i` in `ptcor`; X_{ij} have been standardized as described above; columns of (X_{ij}) have been sorted on cluster labels if `cluster` was specified
`icx` increment in the index of `x` (should = # attributes, since the ith attribute is a row vector in the array)
`y` `y(1+(j-1)*icy)` $= Z_j$, standardized as described above (sorted on cluster and then permuted within cluster)
`icy` increment in the index of `y` (should = 1)
`nclust` # clusters (=1 if `cluster=NULL`)
`mclust` `mclust(k)` = # units in cluster k
`istrt` =1 if tests should be stratified
`wght` the user supplied weights ω_k for the stratified test ($\sum \omega_k = 1$)

References

Benjamini Y and Hochberg Y (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society, Series B*, **57**:289–300.

Bickel PJ and Doksum KA (1977). *Mathematical Statistics.* Holden-Day.

Hochberg Y (1988). A sharper Bonferroni procedure for multiple tests of significance. *Biometrika*, **75**:800–802.

Ibrahim JG, Chen M-H and Gray RJ (2002). Bayesian models for gene expression with DNA microarray data. *J Am Stat Assoc* 97:88–99.

Mutter GL, Baak JP, Fitzgerald JT, Gray R, Neuberg D, Kust GA, Gentleman R, Gullans S, Wei LJ, and Wilcox M. (2001). Global expression changes of constitutive and hormonally regulated genes during endometrial neoplastic transformation. *Gynecol Oncol* 83:177–185.

Westfall PH and Young SS (1993). *Resampling Based Multiple Testing.* Wiley.

Wichmann BA and Hill ID (1982). [Algorithm AS 183] An efficient and portable pseudo-random number generator. *Applied Statistics*, 31:188–190. (Correction, 1984, 33:123.)