BRB-ArrayTools Version 3.2

User's Manual

by

Dr. Richard Simon
Biometrics Research Branch
National Cancer Institute

and

Amy Peng Lam
The EMMES Corporation

April 7, 2004

Table of Contents

Introduction

Purpose of this software

BRB-ArrayTools is an integrated software package for the analysis of DNA microarray data. It was developed by the Biometric Research Branch of the Division of Cancer Treatment & Diagnosis of the National Cancer Institute under the direction of Dr. Richard Simon. BRB-ArrayTools contains utilities for processing expression data from multiple experiments, visualization of data, multidimensional scaling, clustering of genes and samples, and classification and prediction of samples. BRB-ArrayTools features drill-down linkage to NCBI databases using clone, GenBank, or UniGene identifiers, and drill-down linkage to the NetAffx database using Probeset ids. BRB-ArrayTools can be used to analyze both single-channel and dual-channel experiments. The package is very portable and is not restricted to use with any particular array platform, scanners, image analysis software or database. The package is implemented as an Excel add-in so that it has an interface that is familiar to biologists. The computations are performed by sophisticated and powerful analytics external to Excel but invisible to the user. The software was developed by statisticians experienced in the analysis of microarray data and involved in research on improved analysis tools. BRB-ArrayTools serves as a tool for instructing users on effective and valid methods for the analysis of their data. The existing suite of tools will be updated as new methods of analyses are being developed.

Overview of the software's capabilities

BRB-ArrayTools can be used for performing the following analysis tasks:

- **Collating data:** Importing your data to the program and aligning genes from different experiments. The software can load a maximum of 249 experiments consisting of up to 65,000 genes. However, other limitations may apply, which depend on the user's system resources. The entire set of genes may be spotted or printed onto a single array, or the set of genes may be spotted or printed over a "multi-chip" set of up to five arrays. Users may elect whether or not to average over genes which have been multiply spotted or printed onto the same array. Both dual-channel and single-channel (such as Affymetrix) microarrays can be analyzed. A data import wizard prompts the user for specifications of the data, or special interface may be used for Affymetrix or NCI format data. Data should be in tab-delimited text format. Data which is in Excel workbook format can also be used, but will automatically be converted by BRB-ArrayTools into tab-delimited text format.

- **Gene annotations:** Data can be automatically annotated using standard gene identifiers, either using the SOURCE database, or by importing automatic annotations for specific Affymetrix chips. If data has been annotated using the gene annotation tool, then annotations will appear with all output results, and Gene Ontology (GO) classification terms may be analyzed for the class comparison, class prediction, survival, and quantitative traits analyses. Gene Ontology structure files may also be automatically updated from the GO website.

- **Filtering, normalization, and gene subsetting:** Filter individual spots (or probesets) based on channel intensities (either by excluding the spot or thresholding the intensity), and by spot flag and spot size values. Affymetrix data can also be filtered based on the Detection Call. For dual-channel experiments, arrays can be normalized by median-centering the log-ratios in each array, by subtracting out a lowess-smoother based on the average of the red and green log-intensities, or by defining a list of housekeeping genes for which the median log-ratio will be zero. For single-channel experiments, arrays can be normalized to a reference array, so that the difference in log-intensities between the array and reference array has median of zero over all the genes on the array, or only over a set of housekeeping genes. The reference array may be chosen by the user, or automatically chosen as the median array (the array whose median log-intensity value is the median over all median log-intensity values for the complete set of arrays). Each array in a multi-chip set is normalized separately. Outlying expression levels may be truncated. Genes may be filtered based on the percentage of expression values that are at least a specified fold-difference from the median expression over all the arrays, by the variance of log-expression values across arrays, by the percentage of missing values, and by the percentage of "Absent" detection calls over all the arrays (for Affymetrix data only). Genes may be excluded from analyses based on strings contained in gene identifiers (for example, excluding genes with "Empty" contained in the Description field). Genes may also be included or excluded from analyses based on membership within defined genelists.

- **Scatterplot of experiment v. experiment:** For dual-channel data, create clickable scatterplots using the log-red, log-green, average log-intensity of the red and green channels, or log-ratio, for any pair of experiments (or for the same experiment). For "M-A plots" (i.e., the plot of log-ratios versus the average red and green log-intensities), a trendline is also plotted. For single-channel data, create clickable scatterplots using the log-intensity for any pair of experiments. All genes or a defined subset of genes may be plotted. Hyperlinks to NCI feature reports, GenBank, NetAffx, and other genomic databases.

- **Scatterplot of phenotype classes:** Create clickable scatterplots of average log-expression within phenotype classes, for all genes or a defined subset of genes. If more than two class labels are present, then a scatterplot is created for each pair of class labels. Hyperlinks to NCI feature reports, GenBank, NetAffx, and other genomic databases.

- **Hierarchical cluster analysis of genes:** Create cluster dendrogram and color image plot of all genes. For each cluster, provides a hyperlinked list of genes, and a lineplot of median expression levels within the cluster versus experiments. The experiments may be clustered separately with regard to each gene cluster. Each gene cluster can be saved and used in later analyses. A color image plot of median expression levels for each gene cluster versus experiments is also provided. The cluster analysis may be based on all data or on a user-specified subset of genes and experiments.

- **Hierarchical cluster analysis of experiments:** Produces cluster dendrogram, and statistically-based cluster-specific reproducibility measures for a given cut of the cluster dendrogram. The cluster analysis may be based on all data or on a user-specified subset of genes and experiments.

- **Interface for Cluster 3.0 and TreeView:** Clustering and other analyses can now be performed using the Cluster 3.0 and TreeView software, which was originally produced by the Stanford group. This feature is only available for academic, government and other non-profit users.

- **Multidimensional scaling of samples:** Produces clickable 3-D rotating scatterplot where each point represents an experiment, and the distance between points is proportional to the dissimilarity of expression profiles represented by those points. If the user has PowerPoint installed, then a PowerPoint slide is also created which contains the clickable 3-D scatterplot. The PowerPoint slide can be ported to another computer, but must be run on a computer which also has BRB-ArrayTools v3.0 or later installed, in order for the clickable 3-D scatterplot to execute.

- **Global test of clustering:** Statistical significance tests for presence of any clustering among a set of experiments, using either the correlation or Euclidean distance metric. This analysis is given as an option under the multidimensional scaling tool.

- **Class comparison between groups of arrays:** Uses univariate parametric and non-parametric tests to find genes that are differentially expressed between two or more phenotype classes. This tool is designed to analyze either single-channel data or a dual-channel reference design data. The class comparison analysis may also be performed on paired samples. The output contains a listing of genes that were significant and hyperlinks to NCI feature reports, GenBank, NetAffx, and other genomic databases. The parametric tests are either t/F tests, or random variance t/F tests. The latter provide improved estimates of gene-specific variances without assuming that all genes have the same variance. The criteria for inclusion of a gene in the gene list is either a p-value less than a specified threshold value, or specified limits on the number of false discoveries or proportion of false discoveries. The latter are controlled by use of multivariate permutation tests. The tool also includes an option to analyze randomized block design experiments, i.e., take into account influence of one additional covariate (such as gender) while analyzing differences between classes.

- **Class comparison between red and green channels:** Uses univariate parametric and non-parametric tests to find genes that are differentially expressed between two phenotype classes. This tool is designed to analyze data from a non-reference design experiment where the red and green samples represent the two distinct phenotype classes. Only one specific type of non-reference design experiment is allowed. Specifically, only two classes can be compared; red and green samples should be from two different classes; aside from technical replicates and die-swap experiments, each sample should be present on only one array, paired with only one sample of the opposite class. The output and the criteria for inclusion of a gene in the gene list are identical to the one implemented for class comparison of single-channel and reference-design experiments. As a special case, this tool can also analyze reference design with one class compared with the common reference.

- **Class prediction:** Constructs predictors for classifying experiments into phenotype classes based on expression levels. Six methods of prediction are used: compound covariate predictor, diagonal linear discriminant analysis, k-nearest neighbor (using k=1 and 3), nearest centroid, and support vector machines. The compound covariate predictor and support vector machines are only implemented for the case when the phenotype variable contains only two class labels, whereas the diagonal linear discriminant analysis, k-nearest neighbor and nearest centroid may be used even when the phenotype variable contains more than two class labels. Determines cross-validated misclassification rate and performs a permutation test to determine if the cross-validated misclassification rate is lower than would be expected by chance. The class prediction analysis may also be performed on paired samples. The criterion for inclusion of a gene in the predictor is a p-value less than a specified threshold value. For the two-classes prediction problem, a specified limit on the univariate misclassification rate can be used instead of the parametric p-value. In addition, a specified limit on the fold-ratio of geometric means of gene expressions between two classes can be imposed. The output contains the result of the permutation test on the cross-validated misclassification rate, and a listing of genes that comprise the predictor, with parametric p-values for each gene and the CV-support percent (percent of times when the gene was used in the predictor for a leave-one-out cross-validation procedure). The hyperlinks to NCI feature reports, GenBank, NetAffx, or other genomic databases are also included. Permits application of predictive models developed for one set of samples to expression profiles of a separate test set of samples.

- **Binary tree prediction:** The multistage algorithm constructs a binary tree for classifying experiments into phenotype classes based on expression levels. Each node of the tree provides a classifier for distinguishing two groups of classes. The structure of the tree is optimized to minimize the cross-validated misclassification rate. The binary tree prediction method can be based on any of the six prediction methods (compound covariate predictor, diagonal linear discriminant analysis, k-nearest neighbor using k=1 or 3, nearest centroid, and support vector machines). Unlike the class prediction tool, the compound covariate predictor and support vector machines can be used even for the case when the phenotype variable contains more than two class labels. All the other options of this tool are identical to the class prediction tool. The output contains the description of the binary tree and the result of the permutation test on the cross-validated misclassification rate (if requested by the user). For each node of the tree, the result of the permutation test on the cross-validated misclassification rate, and a listing of genes that comprise the predictor are shown. Listings of genes include parametric p-values, CV-support percent, the hyperlinks to NCI feature reports, GenBank, NetAffx, or other genomic databases.

- **Survival analysis:** Uses Cox regression (with Efron handling of ties) to identify genes that are significantly correlated with survival. The output contains a listing of genes that were significant and hyperlinks to NCI feature reports, GenBank, NetAffx, and other genomic databases. The criteria for inclusion of a gene in the gene list is either a p-value less than a specified threshold value, or specified limits on the number of false discoveries or proportion of false discoveries. The latter are controlled by use of multivariate permutation tests.

- **Quantitative traits analysis:** Correlates gene expression with any quantitative trait of the samples. Either Spearman or Pearson correlation tests are used. The output contains a listing of genes that were significant and hyperlinks to NCI feature reports, GenBank, NetAffx, and other genomic databases. The criteria for inclusion of a gene in the gene list is either a p-value less than a specified threshold value, or specified limits on the number of false discoveries or proportion of false discoveries. The latter are controlled by use of multivariate permutation tests.

- **Gene Ontology comparison tool:** Classes are compared by GO category rather than with regard to individual genes. Provides a list of GO categories that have more genes differentially expressed among the classes than expected by chance. P-values of two permutation tests, LS and KS, are used to select these GO categories. A GO category is selected if the corresponding LS or KS permutation p-value is below the threshold specified by the user. The GO categories are ordered by the p-value of the LS test (smallest first).

- **Gene List comparison tool:** Investigates user-defined genelists and selects a set of genelists with more genes differentially expressed among the classes than expected by chance. P-values of two permutation tests, LS and KS, are used to select these gene lists. A genelist is selected if the corresponding LS or KS permutation p-value is below the threshold specified by the user. The gene lists are ordered by the p-value of the LS test (smallest first).

- **Plugins:** Allows users to share their own analysis tools with other users. Advanced users may create their own analysis tools using the R language, which can then be distributed to other users who have no knowledge of R. Details about the Plugin utility are covered in a separate manual.

A note about single-channel experiments

All of the tools within BRB-ArrayTools can be equally run on single-channel and dual-channel experiments. For Affymetrix data, it is suggested that the "signal" field produced in MAS 5.0 should be used as the intensity signal. If the "average difference" field is used as the intensity signal, then genes with negative "average difference" will be automatically thresholded to a value of 1 (log-transformed value of 0), unless the user specifically elects to set those negative “average difference” values to missing during the log-transformation. For sake of convenience of exposition, we will assume dual-channel data throughout this document. We will refer to log-ratios, though a comparable analysis can be run on the log-intensities for single-channel data. We will also refer to "spots" but for Affymetrix arrays the analog of "spot" is the probe set used to detect the expression of a specified gene.

All analyses on log-intensities work exactly the same way as the analyses on log-ratios, but for two exceptions:

1) Data normalization: Dual-channel data is normalized within each array, whereas single-channel data is normalized relative to a designated reference array. See the section on Normalization for more details.

2) Gene filtering: BRB-ArrayTools includes a special filter for single-channel data, to allow users to filter out genes with more than a minimum percentage of “Absent” Affymetrix detection calls.

3) Class prediction: To reduce the dominating effect that genes with overall high intensity might have, the single-channel log-intensities are median-centered on a gene-by-gene basis for all class prediction methods.

Installation

System Requirements

BRB-ArrayTools is Windows-compatible software, Windows 98/2000/NT/XP or later. It is designed to be used as an add-in to Excel 2000 or later. BRB-ArrayTools is no longer supported for Excel 97.

It is recommended that the user have at least 256MB of RAM. It is possible to run BRB-ArrayTools with less than 256MB of RAM; however, some procedures may be very slow or may not be able to run at all, depending on the size of the dataset.

For producing high-resolution graphics, it is recommended that the user set the display color palette to **True Color**. To change the display color palette settings, click on the following sequence of items: **Windows Start Button** → **Settings** → **Control Panel** → **Display** → **Settings**.

Installing the software components

The BRB-ArrayTools software and associated files may be downloaded from the BRB website:

http://linus.nci.nih.gov/BRB-ArrayTools.html

If you have Excel open, please close Excel before installing BRB-ArrayTools.

There are three installation steps:

1. If you do not already have the **Java Runtime Environment, version 1.2.2 or later** on your computer, then you should download and execute the `j2re-1_4_1_01-windows-i586.exe` installation file, which installs the JRE 1.4.1 onto your computer. (If you already have a previous version of BRB-ArrayTools on your computer, then you do not need to complete this step because you should already have the JRE v1.2.2 on your computer.)

2. If you do not already have the **R 1.8.1 or later** software on your computer, then you should download and execute the `rw1081.exe` installation file, which installs the R 1.8.1 software onto your computer.

3. Now you are ready to install **BRB-ArrayTools**. Download and execute the `ArrayTools_v3_2.exe` installation file. If not you do not already have the **R-(D)COM 1.2 or later**, then the BRB-ArrayTools installer will automatically install the R-(D)COM. If you do not already have R 1.8.1 or later installed, then

you should go back to step #2 and install R 1.8.1 first before installing BRB-ArrayTools. If you have a previous version of BRB-ArrayTools installed, then your new version of BRB-ArrayTools will overwrite your previous version.

Loading the add-in into Excel

To begin using BRB-ArrayTools, you must have BRB-ArrayTools loaded into Excel as an add-in. The BRB-ArrayTools installer now pre-loads BRB-ArrayTools into Excel, so users no longer have to load BRB-ArrayTools into Excel as a separate step as they did with previous versions of BRB-ArrayTools. Users may still load and unload BRB-ArrayTools in Excel manually by going to the **Tools** menu within Excel and clicking on **Add-Ins**.

Previous versions of BRB-ArrayTools required users to load the RServer add-in as a separate step. This extra step is no longer required, so users who had a previous version of BRB-ArrayTools should going to the **Tools** menu within Excel, click on **Add-Ins**, and check to make sure that the RServer add-in (if it is still showing in the list of available add-ins) is *deselected*. Leaving the RServer add-in loaded as a separate add-in may cause BRB-ArrayTools to display spurious error messages such as "This workbook is referenced by another workbook and cannot be closed" when loading and unloading the ArrayTools add-in. For this reason, it is best not to load the RServer as a separate add-in, since the ArrayTools add-in will now automatically load the RServer internally.

If you close an Excel session with BRB-ArrayTools still loaded, then your next Excel session will automatically begin by re-loading BRB-ArrayTools. BRB-ArrayTools will not be unloaded from Excel unless you specifically unload it through the **Add-Ins** dialog.

A note about security settings

In order for BRB-ArrayTools to function, you must have your Excel Security Level set to either Medium or Low, and you must select the 'Trust all installed add-ins and templates' option. Office XP users must also select 'Trust access to Visual Basic Project'. To change your security settings within Excel, go to the **Tools → Macro → Security…** menu item from your Excel menubar.

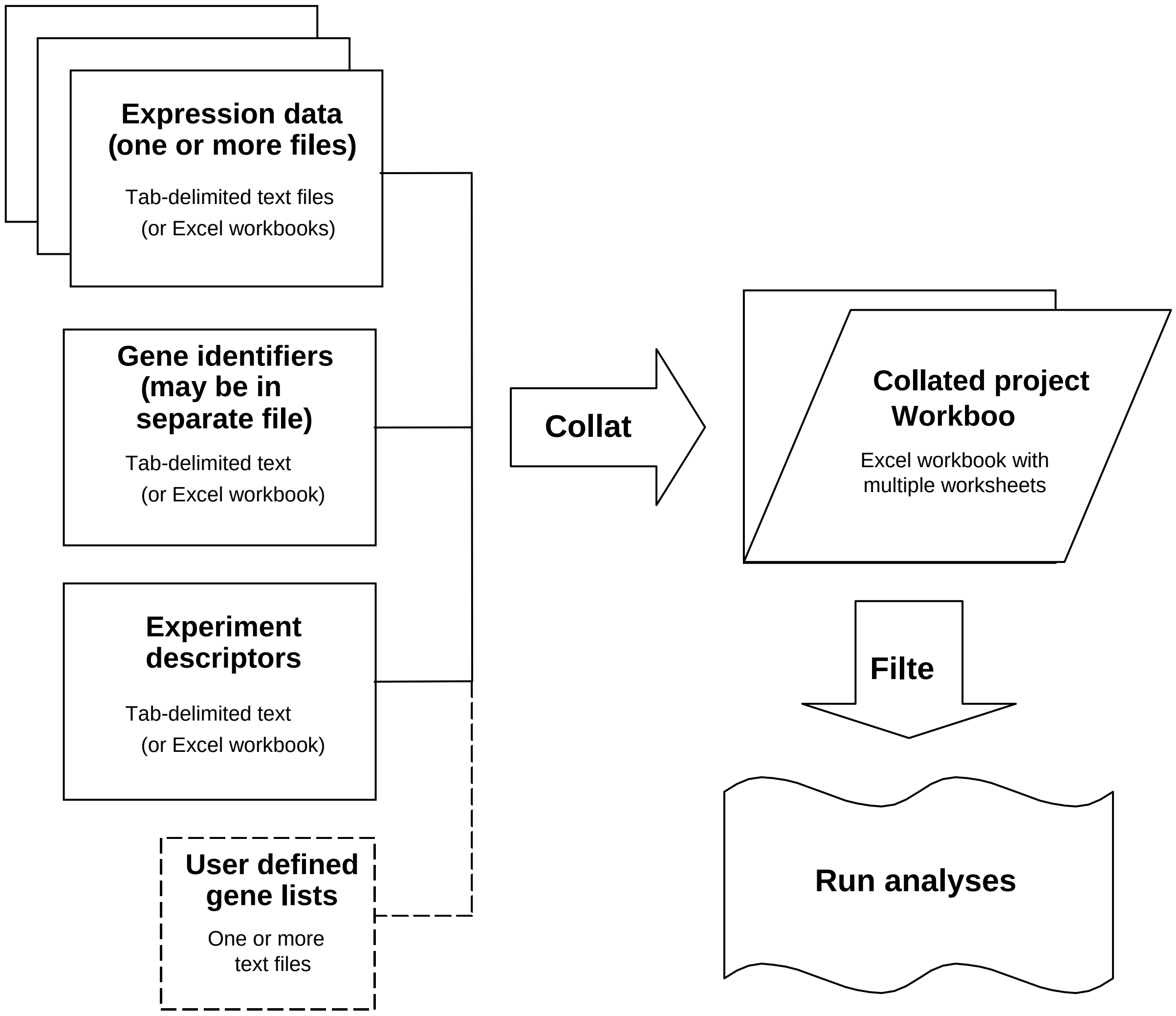
Collating the data

Overview of the collating step

Collating is a process in which your data are read, sorted, and written to a standard Excel workbook format which is "understood" by BRB-ArrayTools. The user inputs three types of data (expression data, gene identifiers, and experiment descriptors) by entering parameters to describe the file format of each data type. BRB-ArrayTools then processes these data files and produces a **collated project workbook** which is a standard data format used by all the analysis tools included in the software package. Curated gene lists may also be used to define gene functions or pathways, and will be read in at the time of collating. The figure below shows a diagram of the collation process.

A note about Excel workbooks

A single Excel file (with an .xls extension) is called a **workbook**, and a workbook may contain one or more spreadsheets called **worksheets**. A tab-delimited ASCII text file with an .xls file extension, though not a true Excel object, is interpreted by Excel to be a workbook containing a single worksheet.


Expression data
(one or more files)
Tab-delimited text files
(or Excel workbooks)
Gene identifiers
(may be in
separate file)
Tab-delimited text
(or Excel workbook)
Experiment
descriptors
Tab-delimited text
(or Excel workbook)
User defined
gene lists
One or more
text files
Collat
Collated project
Workboo
Excel workbook with
multiple worksheets
Filte
Run analyses

Input to the collating step

The input data files should be tab-delimited ASCII text files. Excel workbooks (where the input data is on the *first* worksheet if there are multiple worksheets within the workbook) may be used instead of tab-delimited text files. However, BRB-ArrayTools will automatically convert those Excel files into tab-delimited text format. If the user wishes to save a copy of the original Excel files, then the user should first save a copy of those original data files into another folder. The user must enter enough information describing the format of these input data files (such as the column designation of each data element) so that BRB-ArrayTools can find and process these data elements from the input files.

Input data elements

Expression data

BRB-ArrayTools accepts two input file formats for the expression data: (1) arrays **horizontally aligned** in one file, or (2) arrays in **separate files**. Both the horizontally aligned and separate files formats can be used with multi-chip sets. If the genes are listed in the same order and there is a one-to-one correspondence between the rows of each of the expression data columns, then the data is said to be **aligned**. The horizontally aligned data format can only be used when the genes are already aligned, while the separate files format may be used whether or not the genes are already aligned. The user has the option whether or not to geometrically average over replicated spots within an array. For multi-chip sets, genes that are multiply-spotted over several arrays can have replicated spots within the same array geometrically averaged, but replicates across different arrays will not be averaged. For multi-chip sets, the individual arrays are normalized separately, but the data for all arrays within a set will be concatenated to form a “virtual array”.

The required expression data elements are either the red and green signal intensity columns or the log-ratio column (for dual-channel data), or a single signal intensity column (for single-channel data). If a background subtraction is desired for the dual-channel data, then the red and green background columns are also necessary. In addition, a spot flag or spot size column may optionally be entered, to allow spot filtering using these data elements. For Affymetrix data, the Detection Call (Absent, Marginal, or Present) may be used as the “spot flag” variable for the purpose of flag filtering.

For single channel data, all signal intensity values less than 1 will automatically be thresholded to 1 by default, before the log transformation is applied. Users who do not wish to automatically threshold their data can turn off this default action by selecting the “Do not threshold to 1 (e.g., CodeLink)” checkbox at the time of collating. For CodeLink data, signal intensity values have already been normalized so that half of all signal intensities on an array are between 0 and 1, so that it does not make sense to threshold these values to 1. When the “Do not threshold to 1” option is selected, then signal intensity values which are negative or 0 will be set to missing, since a log transformation is not valid on such data values. Please note, however, that the "Do not threshold to 1" option is IRREVERSIBLE! Once the negative or 0 values have been set to missing, they can never be thresholded again to 1 by re-filtering the data. Likewise, once the values less than 1 have been thresholded to 1, the negative or 0 values can never be separated from the values in the interval between 0 and 1 and be subsequently set to missing. In order to change the "Do not threshold to 1" option, the data must be re-collated.

Gene identifiers

Various identifiers can be associated with each spot, such as spot numbers, well numbers, clone names, clone identifiers, probe set identifiers, UniGene identifiers, GenBank accession numbers, etc. The gene identifiers may be located alongside the expression data in the same files, or may be contained in a separate file which is used as a look-up table for the genes in all the arrays. For multi-chip sets using a separate gene identifier file, only one gene identifier file should be used for the entire set of arrays. If the gene identifiers are contained in a separate file, then there must be corresponding columns within the expression data file(s) and the gene identifier file, containing gene ids which can be used for matching the gene identifiers with the expression data.

The column which is designated within BRB-ArrayTools as **clone id** should contain an organization-prefixed clone id (e.g., a prefix such as "IMAGE:", "ATCC:", "TIGR:" etc.). These clone ids can be used to link to clone reports in the NCBI database. Note that clone reports in the NCI mAdb database are only available for clones in the NCI Advanced Technology Center inventory or for other expression array sets that are tracked by BIMAS/CIT/NIH. All clone identifiers found within a **clone id** column that are numeric and have no prefix will be assumed to have a prefix of IMAGE by default.

Probe set ids are used to link to feature reports in the NCI mAdb database. Currently feature reports are available only for the Human Genome U133 A and B chips, and for the Mouse Genome U74 A-C chips. For Class Comparison, Class Prediction, and Survival Analysis output, probe set ids are also used for batch query links to NetAffx.

UniGene cluster ids and **gene symbols** are used to search for the UniGene annotation mirrored in the NCBI database. **GenBank accession numbers** are used to search for the GenBank annotation which is also mirrored in the NCBI database.

A minimum of one gene identifier is required for use in collating the dataset. However, the user may wish to enter any or all of the above gene identifiers, if they are available, to enhance the usability of the output from the analyses.

Experiment descriptors

A file containing experiment descriptors must be prepared by the user and input during collation. Each row in the experiment descriptor file represents an experiment in the dataset (except for the first row which is a header row), and each column represents a descriptor variable whose entries are used to describe each of the experiments. For multi-chip sets, each row in the experiment descriptor file should represent an entire set of arrays performed with the same sample, not a separate row for each individual array. The experiment descriptor file should contain *exactly* the same experiments as those to be collated (i.e., it should not contain any extra rows representing experiments which are not represented in the dataset to be collated).

The first column of the experiment descriptor file should contain the names of the experiments. If the expression data is in a separate file for each array, then these names should be the filenames without the ".xls" or ".txt" file extensions. For multi-chip sets with separate data files for each array, these experiment names should be the filenames without the optional "_A", "_B", "_C", "_D", and "_E" suffixes and without the ".xls" or ".txt" file extensions. For horizontally aligned data, the order of the arrays in the experiment descriptor file is assumed to correspond to the order of the arrays in the gene expression data file.

Each succeeding column in the experiment descriptor file contains a descriptor variable which may be used for labeling purposes, for identifying reverse fluor arrays, for classification analyses, for identifying replicate experiments, for matching between paired experiments, or for specifying the plotting order of the experiments when clustering genes. The user can create as many columns of the experiment descriptor file as he/she finds useful for classifying the arrayed samples. There should be no empty columns between the experiment descriptor columns.

User-defined genelists

BRB-ArrayTools allows the user to associate genes with defined functions or pathways through the use of user-defined genelists. Unlike the other input data files mentioned above which collectively describe the expression, genes, and samples of a specific project, the user-defined genelists are stored in the `ArrayTools` installation directory, and are used as a general database for all projects that are analyzed within BRB-ArrayTools.

In the `ArrayTools` installation directory, there is a `Genelists` subdirectory, which contains the subdirectories `Cgap` and `User`. The `Cgap` directory contains curated gene lists which have been downloaded from the Cancer Genome Anatomy Project website (http://cgap.nci.nih.gov/Genes/CuratedGeneLists). These HTML files will be updated with each new release of BRB-ArrayTools, and may be manually updated in the interim by the user by replacing or adding to the existing files with new files downloaded from the CGAP website.

The `User` directory contains genelists that may be created by the user. Unlike the HTML formatted files contained in the `Cgap` directory, the files contained in the `User` directory are simply ASCII text files containing lists of genes. The name of the file denotes the function or pathway that is defined by the genes listed in the file. The first row of the file specifies the type of gene identifier used within the file, and must *exactly* match one of the following pre-defined labels

```
Unique id
Description
Clone
Probe set
GB acc
UG cluster
Gene symbol
```

or must exactly match one of the write-in gene identifier names specified in the **column format** section of the **gene identifiers** page of the **collating data** dialog box. Each succeeding row of the file should contain the gene identifier of a specific gene or clone to be included in the list.

The `User` directory currently contains some examples of genelists which will be used with the `BreastSamples` dataset that is provided with this package. These example genelists are used with the example dataset, and may be deleted at any time by the user.

Some procedures will automatically create new user genelists. The following procedures can be used to create new genelists: the hierarchical clustering of genes, the class comparison tools, and the class prediction tools. The name of the genelist that is produced by the class comparison or class prediction tool is specified by the user before running the analysis. If there is already an existing genelist with the same name, then that genelist will be overwritten.

During the collating step, genes in the user's dataset can be automatically matched against CGAP and user-defined genelists which are currently stored within the `Genelists` folder. If any of the genes in the user's dataset match any of those in the genelists, then the name of the genelist will appear in a column of the **gene identifiers** page in the collated project workbook. For example, if the user has the “Activating transcription factor 3” gene in his dataset with the identifiers Clone 428348 and Gene symbol ATF3, then the Clone identifier would be matched against the user genelists and found to be in the “Example –Proliferation” genelist file, while the Gene symbol ATF3 would be matched against the CGAP genelists, and found to be in the “CGL:gene_regulation” and “CFL:transcription” genelist files. Thus, an additional column named **defined genelists** would appear in the **gene identifiers** page of the collated project workbook, and the entry under the defined genelists column would be `Example- Proliferation, CGL:gene_regulation, CGL:transcription` for this particular gene.

If the user adds or deletes a genelist after the data has already been collated, then the **defined genelists** column in the **gene identifiers** page can be updated by clicking on the **ArrayTools → Utilities → Annotate data using genelists** menu item**.**

Minimal required data elements

Although there are many optional data elements that may be input into the collating step, the minimal required data elements for all input formats are as follows:

(1) the expression data (either the red and green signals or the log-ratio for dual-channel data, or the single signal intensity for single-channel data),
(2) at least one gene identifier (may be located alongside the expression data or in a separate gene identifier file), and
(3) at least one experiment descriptor column (the experiment id) in the experiment descriptor file.

Required file formats and folder structures

Horizontally aligned format

For data in the horizontally aligned format, expression data for all experiments are found in one file. Usually the first few columns will be expected to contain gene identifiers. After the gene identifier columns, there should be a set of columns for each experiment in the dataset, and the *order* of the columns *should be the same* for each experiment. Also, there should be no extraneous columns at the *end* of the file. For example, the file may have columns:

Probeset	Signal1	Detection1	Signal2	Detection2	Signal3	Detection3

However, the following set of columns are illegal for the horizontally aligned data format:

Clone Id	GBAcc	Red_ Exp1	Green_ Exp1	Ratio_ Exp1	Green_ Exp2	Red_ Exp2	Ratio_ Exp2	Filter

There are two reasons why the above set of columns are illegal. First, the data elements do not appear in the *same order* for experiment 1 (with “Red”, “Green” and “Ratio”) and experiment 2 (with “Green”, “Red” and “Ratio”). Also, there is an extraneous column (“Filter”) at the end of the file which does not belong to any of the experiments. The above file can be corrected by re-ordering the columns as follows:

Clone Id	GBAcc	Filter	Red_ Exp1	Green_ Exp1	Ratio_ Exp1	Red_ Exp2	Green_ Exp2	Ratio_ Exp2

The “Filter” column will be ignored, since it is neither a gene identifier nor an experiment data element. Also, the “Ratio” columns will be ignored, because the ratio will be automatically computed from the “Red” and “Green” columns.

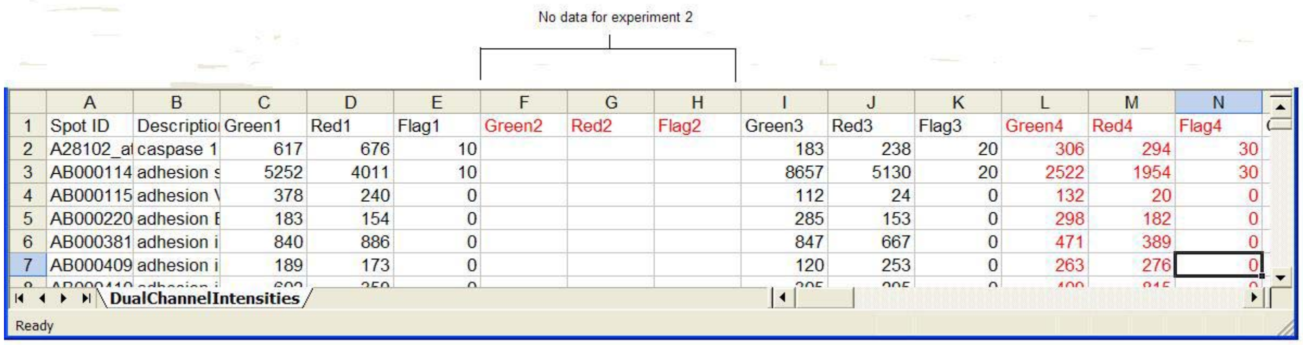
Separate files format

For the separate files format, the user should create a data folder consisting *exclusively* of expression data files (one for each array). The column format (i.e., the order of the data elements) should be the *same* in all the expression data files.

Multi-chip sets

The user must also specify whether the data comes from a multi-chip set. Multi-chip sets are sets of arrays in which the total number of clones or probesets are divided over several arrays. BRB-ArrayTools accepts multi-chip sets with up to 5 array types in the set, with each array in the set labeled as ‘A’, ‘B’, ‘C’, ‘D’, and ‘E’ for convenience. When speaking of multi-chip sets, the term “experiment” is distinguished from the term “array”, since an “experiment” consists of the same sample being hybridized to a set of arrays forming the multi-chip set. However, when speaking of single-chip type experiments (for example when only chip ‘A’ is used from the Affymetrix U133 two-chip set), the terms “experiment” and “array” may then be used interchangeably. BRB-ArrayTools processes each chip type separately, and then concatenates the chip types to form a “virtual array”. The total geneset on the virtual array is limited to approximately 65000. The user does not need to have performed experiments using all of the array types in a multi-chip set, but it is expected that for the array types that were actually used from the set, the same samples should have been used for each array type. For example, if the user chose to use only two chip types from a five-chip set, then for each sample used on array type ‘A’, the same sample should also have been used on array type ‘B’, though allowance can be made for the case when a few chips of either type have “failed”.

When multi-chip data is in the horizontally aligned format, then a *file* is expected for *each* chip type in the set, and the experiments should appear in the same order within each file. For example, a user may have a file named “ChipA.xls” containing data columns for “Sample1”, “Sample2” and “Sample3”, and another file named “ChipB.xls” containing data columns for “Sample1”, “Sample2” and “Sample3” in the same order as the columns in the “ChipA.xls” file. If data is not available or an experiment is not performed for a sample using the given array type, then the data columns for that experiment in the horizontally aligned file for that array type should be empty but not missing. For example, experiment 2 is not performed for this array type, but the data columns are still listed.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Spot ID	Descriptio	Green1	Red1	Flag1	Green2	Red2	Flag2	Green3	Red3	Flag3	Green4	Red4	Flag4
2	A28102_at	caspase 1	617	676	10				183	238	20	306	294	30
3	AB000114	adhesion s	5252	4011	10				8657	5130	20	2522	1954	30
4	AB000115	adhesion \	378	240	0				112	24	0	132	20	0
5	AB000220	adhesion E	183	154	0				285	153	0	298	182	0
6	AB000381	adhesion i	840	886	0				847	667	0	471	389	0
7	AB000409	adhesion i	189	173	0				120	253	0	263	276	0

When multi-chip data is in the separate files format, then a *data folder* is expected for *each* chip type in the set. For example, if the dataset described above were formatted as separate files, then the user might have a folder called "ChipA" containing identically formatted files "Sample1.xls", "Sample2.xls", and "Sample3.xls", and another folder called "ChipB" containing identically formatted files "Sample1.xls", "Sample2.xls", and "Sample3.xls".

The data for each of the array types must be in a separate files format similar to that described in Example 2, with the expression data files for each chip type must be contained in a separate folder. For example, suppose the user performed three experiments, `Exp1`, `Exp2`, and `Exp3`, on chips 'A' and 'B' in a multi-chip set. Then the user would have two folders, with each folder containing a data file for each of the three experiments as follows:

<u>In data folder for 'A':</u>

`Exp1_A.txt`

GeneId	**Signal**	**Flag**
Gene1	10000	P
Gene2	10300	P
Gene3	1031	A

`Exp2_A.txt`

GeneId	**Signal**	**Flag**
Gene1	20420	P
Gene2	324	A
Gene3	34578	P

`Exp3_A.txt`

GeneId	**Signal**	**Flag**
Gene1	67835	P
Gene2	4568	P
Gene3	345	A

In data folder for 'B':

`Exp1_B.txt`

GeneId	Signal	Flag
Gene4	34679	P
Gene5	5321	P
Gene6	23568	P

`Exp2_B.txt`

GeneId	Signal	Flag
Gene4	2345	P
Gene5	7890	P
Gene6	34676	P

`Exp3_B.txt`

GeneId	Signal	Flag
Gene4	34576	P
Gene5	3378	P
Gene6	45879	P

The filenames in the data folders *must be the same* for each experiment, except that they are allowed to differ by the suffixes '_A', '_B', '_C', '_D', and '_E'. However, the filenames need not differ at all.

For example, the following two sets of filenames would have also been acceptable:

Data Folder for 'A': `Exp1.txt`, `Exp2.txt`, `Exp3`.txt
Data Folder for 'B': `Exp1_B.txt`, `Exp2_B.txt`, `Exp3_B`.txt

or

Data Folder for 'A': `Exp1.txt`, `Exp2.txt`, `Exp3`.txt
Data Folder for 'B': `Exp1.txt`, `Exp2.txt`, `Exp3`.txt

The experiment labels in the first column of the Experiment Descriptor file should match the filenames (without the '_A', '_B', '_C', '_D', and '_E' file extensions). So, for all of the above examples, the Experiment Descriptor file should contain the following experiment labels in the first column:

ExpLabel	SampleId	Class
Exp1	001	Tumor
Exp2	002	Normal
Exp3	003	Tum

		or

In the **filtered log intensity worksheet** of the collated project workbook, the data from the ‘A’and ‘B’ chips will be concatenated as follows:

GeneId	**Exp1**	**Exp2**	**Exp3**
Gene1	13.288	14.318	16.05
Gene2	13.33	8.34	12.157
Gene3	10.01	15.078	8.43
Gene4	15.082	11.195	15.077
Gene5	12.377	12.946	11.722
Gene6	14.525	15.082	15.486

If one of the chips fail due to experimental reasons, then the corresponding expression data file for that chip may be missing from one of the folders. For example, if `Exp2` had not been performed successfully for the ‘A’ chip, then the `Exp2_A.txt` file would be missing from the data folder ‘A’. However, the Experiment Descriptor file should still contain an entry for `Exp2`, since the experiment still exists in the ‘B’ folder.

The following set of filenames are not acceptable, because BRB-ArrayTools would not know how to match up the files in data folder ‘A’ with the files in data folder ‘B’:

Data Folder ‘A’: `Sample1.txt`, `Sample2.txt`, `Sample3.txt`
Data Folder ‘B’: `Exp1.txt`, `Exp2.txt`, `Exp3.txt`

In fact, for the above case, BRB-ArrayTools will think that there are actually six experiments, and that the experiments `Sample1`, `Sample2`, and `Sample3` had failed for the ‘B’ chip, while the experiments `Exp1`, `Exp2`, and `Exp3` had failed for the ‘A’ chip.

Using the collation dialogs

Collating data using the data import wizarad

To import data into BRB-ArrayTools, the user should go to the

ArrayTools -> Collate data -> Data import wizard

menu item. A wizard will prompt the user for the data type and location of the data elements described above. The user must specify the following information within the wizard screens:

Data types (e.g., single or dual channel, single array type or multiple array types, etc.)
File type (e.g., horizontally aligned file format or expression data in a separate file for each array)
Expression data
Gene identifiers
Experiement descriptors

Wizard screen: data types

BRB-ArrayTools can import the following data types:

1. Dual-channel intensities	Data consists of red and green channel intensities. Background columns may also be included if intensities have not yet been background-subtracted. Data has not yet been log-transformed.
2. Dual-channel log-ratios	Data consists of log-ratios (= log2(red/green)).
3. Single-channel intensity	Data from single channel experiments. Data has not yet been log-transformed.
4. Affymetrix data	Probeset-level data from Affymetrix GeneChips. Data consists of the Signal (MAS 5.0), Avg Diff (MAS 4.0), or other signal computed from the probe-level data. Data has not yet been log-transformed.

Important: Each data column must be clearly labeled with a variable name, and data columns should be tab-delimited. The data file should not contain consecutive tabs, except as necessary to denote missing values in the data columns.

The dual-channel and single-channel data types (data types #1, #2, and #3 above) may also include optional spot flag or spot size columns, which are used to indicate spot quality. Affymetrix data (data type #4 above) may also include detection call or absolute call columns.

Some datasets may fall into more than one category. For instance, the user may have dual-channel data in GenePix files which contain the individual channel intensities as well as a log-ratio column. In that case, the user may elect to import either the individual channel intensities (data type #1 above) or the log-ratios (data type #2 above), but not both. If the individual channel intensities are imported instead of the log-ratios, then log-ratios will be automatically computed from the channel intensities. However, any normalization which had been applied to the log-ratios in the original expression data file will be lost, and the imported data must be normalized.

Affymetrix data may be imported as single-channel intensity (data type #3 above) or Affymetrix data (data type #4 above). However, the default filtering criteria are different for these two data types, and the filtered data in the collated project workbook will differ slightly unless the user is careful to use the same filtering criteria in both cases. For Affymetrix data (data type #4 above), the user must specify the chip type (e.g., U133, U74, etc.).

Note that, with the exception of dual-channel log-ratio data, all expression data types are expected in the original scale WITHOUT log transformation. BRB-ArrayTools will automatically apply a log-transformation to the data. If the intensity data had already been log-transformed prior to collation, then the user should exponentiate the intensity data (undo the log-transformation) before importing the data into BRB-ArrayTools. In that case, the utility to **Undo log base 2** (found under the **ArrayTools → Utilities → Log-transformation** menu) may be useful in pre-processing the files before collation.

Dual-channel log-ratios, on the other hand, are already expected to be in the log base 2 scale. If the user has dual-channel ratios which have not yet been log-transformed, then the user should log-transform the data before importing it into BRB-ArrayTools. In that case, the utility to **Apply log base 2** (found under the **ArrayTools → Utilities → Log-transformation** menu) may be useful in pre-processing the files before collation.

Wizard screen: file type

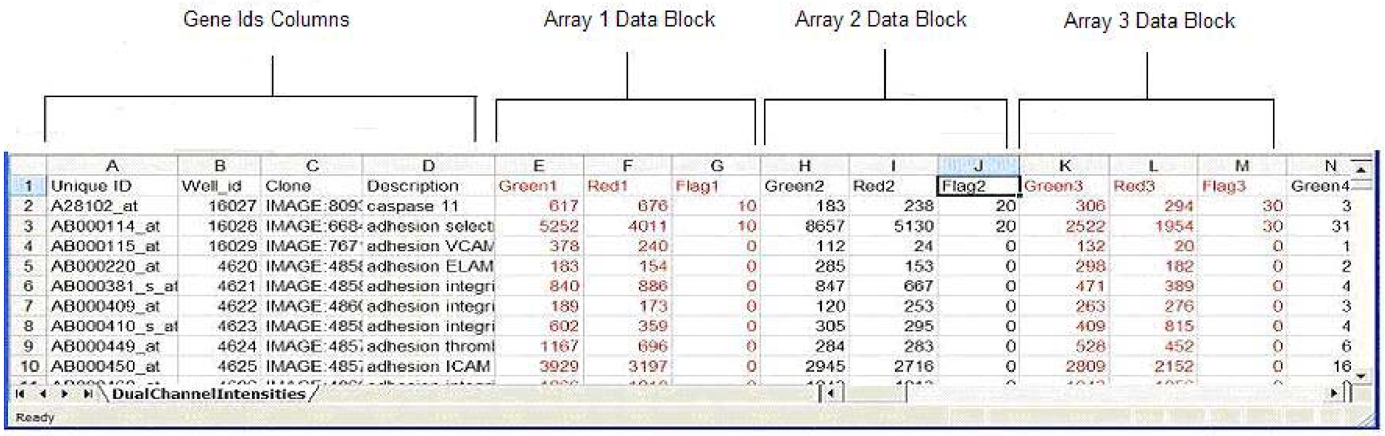
BRB-ArrayTools can accept either tab-delimited ASCII text files, or Excel spreadsheets (a single worksheet within an Excel workbook). If the file is an Excel spreadsheet, then BRB-ArrayTools will automatically convert it to tab-delimited text file with the same name.

The expression data can be saved either in a horizontally aligned file or saved in separate files stored in one folder.

Horizontally aligned file

In this file format, the data columns are organized into array data blocks, where an array data block is a set of consecutive columns containing data from the same array. The data elements must appear in the same order within each array data block. All gene identifier columns must be placed before the first array. If there are any miscellaneous data columns following the last array data block, then the user must either delete those last columns or place them before the first array.

The following figure illustrates an example of a horizontally aligned file with three columns (Green, Red and Flag) in each data block:

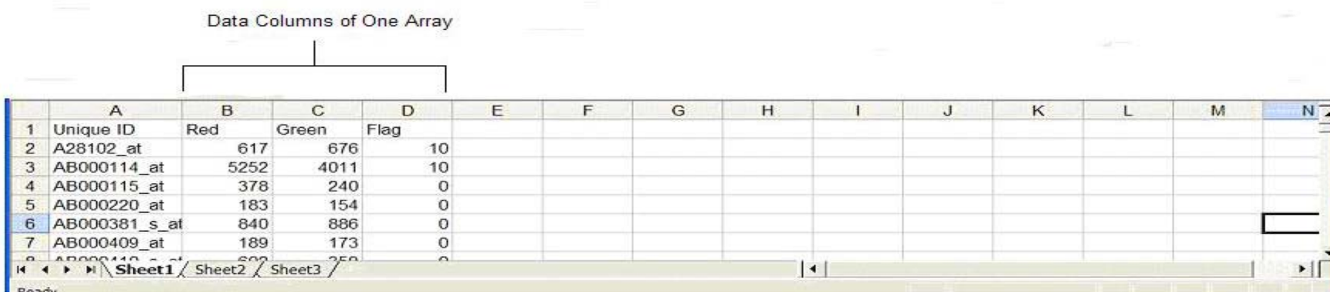


	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Unique ID	Well_id	Clone	Description	Green1	Red1	Flag1	Green2	Red2	Flag2	Green3	Red3	Flag3	Green4
2	A28102_at	16027	IMAGE:809	caspase 11	617	676	10	183	238	20	306	294	30	3
3	AB000114_at	16028	IMAGE:668	adhesion select	5252	4011	10	8657	5130	20	2522	1954	30	31
4	AB000115_at	16029	IMAGE:767	adhesion VCAM	378	240	0	112	24	0	132	20	0	1
5	AB000220_at	4620	IMAGE:485	adhesion ELAM	183	154	0	285	153	0	298	182	0	2
6	AB000381_s_at	4621	IMAGE:485	adhesion integri	840	886	0	847	667	0	471	389	0	4
7	AB000409_at	4622	IMAGE:486	adhesion integri	189	173	0	120	253	0	263	276	0	3
8	AB000410_s_at	4623	IMAGE:485	adhesion integri	602	359	0	305	295	0	409	815	0	4
9	AB000449_at	4624	IMAGE:485	adhesion thromb	1167	696	0	284	283	0	528	452	0	6
10	AB000450_at	4625	IMAGE:485	adhesion ICAM	3929	3197	0	2945	2716	0	2809	2152	0	16

Important: If your experiments consist of multiple array types, then each horizontally aligned file must contain arrays of the same type, and you should have one horizontally aligned file for each array type used. The experiments should appear in the *same order* in each horizontally aligned file. Up to five array types (denoted here as ‘A’, ‘B’, ‘C’, ‘D’ and ‘E’ for convenience) may be used, and data from these array types will be concatenated to form a “virtual array”.

<u>Separate files</u>

In this file format, each array is stored in a separate file. Each file must have identical format, including the same number of header lines before the data lines. The following example illustrates a file with three columns of data:



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Unique ID	Red	Green	Flag										
2	A28102_at	617	676	10										
3	AB000114_at	5252	4011	10										
4	AB000115_at	378	240	0										
5	AB000220_at	183	154	0										
6	AB000381_s_at	840	886	0										
7	AB000409_at	189	173	0										

Important: The data folder must contain only expression data files. Extraneous files should be removed before collation. If your experiments consist of multiple array types, do not mix arrays of different types in the same data folder. Data for each array type should be stored in a separate folder, and corresponding samples performed on each array type should have the *same filename* within their respective folders, except that the names may differ by the extension '_A', '_B', etc. (For example, the user may have a folder called 'ChipTypeA' containing files 'PatientID001_A.txt', 'PatientID0234_A.txt' and 'PatID32_A.txt', and another folder called 'ChipB' containing files 'PatientID001_B.txt', 'PatientID0234_B.txt' and 'PatID32_B.txt'.) Up to five array types (denoted here as 'A', 'B', 'C', 'D' and 'E' for convenience) may be used, and data from these array types will be concatenated to form a "virtual array".

Wizard screen: expression data

To specify the data columns in the expression data file(s), first select the header line that identifies the data columns as well as the first line of data, and then specify the individual column containing the data elements. For the horizontally aligned file format, the user must also select the columns where data for the first array and second array begin. This helps the Wizard to determine how many columns of data belong to each array, and allows the Wizard to calculate the beginning column for each subsequent array. The columns for each array must be placed together, and there should be no miscellanous columns inserted between the data columns for each array or after the last column of the last array.

Here is a brief description of the data columns:

Gene identifier column:

- The Unique ID or Well ID or Spot ID column uniquely identifies each spot (or feature) on the array. If this ID appears more than once within the same file, then it is assumed that the clone has been multiply-spotted onto the array. If the data is Affymetrix, then this column is called "Probe Set Name" or 'Probe Set ID".

For single channel data:

- The Signal Intensity column contains the signal values of the spots (or features). The signal values should NOT have already been log-transformed, since BRB-ArrayTools will automatically perform a log base 2 transformation on the Signal Intensity. If the user has Affymetrix data from MAS 4.0, then the "Avg Diff" column may be used as the Signal Intensity column.

For dual channel data:

- The Red and Green Intensity columns indicate the intensity of the red and green signal. Log-ratios of red-to-green will be computed from the red and green background-adjusted values.

- The Red and Green Background columns contains background values to be subtracted from the Red and Green Intensity columns if they are not already background adjusted. These two columns are optional.

Optional data elements common to single and dual channel data:

- The Spot Size column indicates the size of the spot. Some imaging software use the size data for quality control. This column is optional.

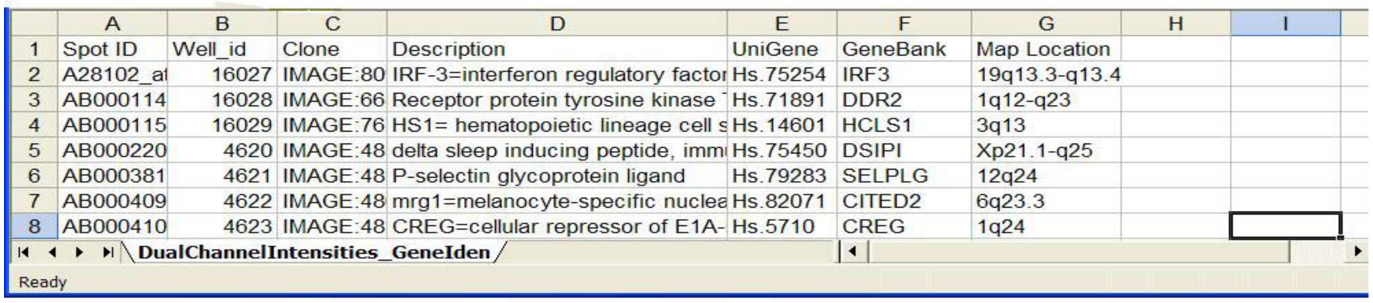
- The Spot Flag column indicates the quality of the spot. Some imaging software set the spot flag values such as 0 or 1 to indicates spot quality. A string such as "Failed" or "Pass" may also be used to indicate spot quality. For Affymetrix data, the Detection Call will be used in lieu of the Spot Flag. This column is optional.

Note: For single channel data, all signal intensity values less than 1 will automatically be thresholded to 1 by default, before the log transformation is applied. Users who do not wish to automatically threshold their data can turn off this default action by selecting the "Do not threshold to 1 (e.g., CodeLink)" checkbox at the time of collating. For CodeLink data, signal intensity values have already been normalized so that half of all signal intensities on an array are between 0 and 1, so that it does not make sense to threshold these values to 1. When the "Do not threshold to 1" option is selected, then signal intensity values which are negative or 0 will be set to missing, since a log transformation is not valid on such data values. Please note, however, that the "Do not threshold to 1" option is IRREVERSIBLE! Once the negative or 0 values have been set to missing, they can never be thresholded again to 1 by re-filtering the data. Likewise, once the values less than 1 have been thresholded to 1, the negative or 0 values can never be separated from the values in the interval between 0 and 1 and be subsequently set to missing. In order to change the "Do not threshold to 1" option, the data must be re-collated.

Wizard screen: gene identifiers

Various identifiers such as spot number, well number, clone number, UniGene cluster identifiers, GenBank accession number or gene title can be associated with each spot. They can be either placed alongside the expression data or stored in a separate gene identifiers file. These identifiers will be hyperlinked in the analysis output. For Affymetrix data, the user has the option of downloading the probeset annotation file directly from BRB's server. These files were originally downloaded from the NetAffx website and have been especially formatted for use with BRB-ArrayTools.

Here is an example of a gene identifiers file:



	A	B	C	D	E	F	G	H	I
1	Spot ID	Well_id	Clone	Description	UniGene	GeneBank	Map Location		
2	A28102_at	16027	IMAGE:80	IRF-3=interferon regulatory factor	Hs.75254	IRF3	19q13.3-q13.4		
3	AB000114	16028	IMAGE:66	Receptor protein tyrosine kinase	Hs.71891	DDR2	1q12-q23		
4	AB000115	16029	IMAGE:76	HS1= hematopoietic lineage cell s	Hs.14601	HCLS1	3q13		
5	AB000220	4620	IMAGE:48	delta sleep inducing peptide, imm	Hs.75450	DSIPI	Xp21.1-q25		
6	AB000381	4621	IMAGE:48	P-selectin glycoprotein ligand	Hs.79283	SELPLG	12q24		
7	AB000409	4622	IMAGE:48	mrg1=melanocyte-specific nuclea	Hs.82071	CITED2	6q23.3		
8	AB000410	4623	IMAGE:48	CREG=cellular repressor of E1A-	Hs.5710	CREG	1q24		

DualChannelIntensities_GeneIden

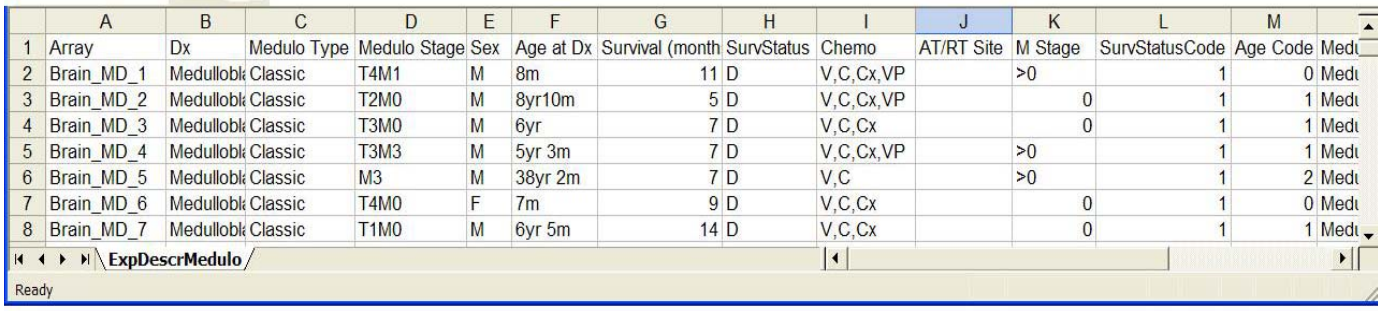
Ready

Important: If the gene identifiers are in a separate file rather than in the expression data file, then you must specify which gene identifier in the separate gene identifiers file should be used to match against the Spot ID (Well ID, Unique ID or Probe Set ID) of the expression data file(s). For multi-array sets using a separate gene identifiers file, all gene identifiers should be contained in one file rather than a separate file for each array type.

Wizard screen: experiment descriptors

In order to facilitate the analysis of your experiments, an experiment descriptors file may be prepared by the user before the collation. If the user does not have an experiment descriptors file prepared in advance, the user may elect to have BRB-ArrayTools create a template.

Here is an example of an experiment descriptors file:



	A	B	C	D	E	F	G	H	I	J	K	L	M	
1	Array	Dx	Medulo Type	Medulo Stage	Sex	Age at Dx	Survival (month	SurvStatus	Chemo	AT/RT Site	M Stage	SurvStatusCode	Age Code	Medu
2	Brain_MD_1	Medullobl	Classic	T4M1	M	8m	11	D	V,C,Cx,VP		>0	1	0	Medu
3	Brain_MD_2	Medullobl	Classic	T2M0	M	8yr10m	5	D	V,C,Cx,VP		0	1	1	Medu
4	Brain_MD_3	Medullobl	Classic	T3M0	M	6yr	7	D	V,C,Cx		0	1	1	Medu
5	Brain_MD_4	Medullobl	Classic	T3M3	M	5yr 3m	7	D	V,C,Cx,VP		>0	1	1	Medu
6	Brain_MD_5	Medullobl	Classic	M3	M	38yr 2m	7	D	V,C		>0	1	2	Medu
7	Brain_MD_6	Medullobl	Classic	T4M0	F	7m	9	D	V,C,Cx		0	1	0	Medu
8	Brain_MD_7	Medullobl	Classic	T1M0	M	6yr 5m	14	D	V,C,Cx		0	1	1	Medu

ExpDescrMedulo

Ready

Except for the first row which is a header row, each row represents an experiment in the dataset. The first column should contain the names of the experiments.

Important: For data in separate files format, the experiment names should be the name of the file minus the ".xls" or ".txt" file extensions. For data stored in a single file, the experiment names should correspond to the order of the arrays listed in the expression data file.

Important: The experiment descriptors file should contain *exactly* the same experiments as those to be collated (i.e., the experiment descriptors file should not contain any extra rows representing experiments which are not represented in the expression data, nor should any experiment which are present in the expression data be missing from the experiment descriptors file). For multi-chip sets, each row should represent an entire set of arrays performed with the same sample, not a separate row for each individual array.

Each succeeding column contains a descriptor variable which may be used for labeling purposes, for identifying reverse fluor arrays, for classification analyses, for identifying replicate experiments, for matching between paired experiments, or for specifying the plotting order of the experiments when clustering genes. The user can create as many columns of the experiment descriptors file as he/she finds useful for classifying the arrayed samples. There should be no empty columns between the experiment descriptor columns.

If the dataset contains reverse fluor experiments, then select this checkbox, and specify the column in the experiment descriptor sheet and the labels in this column which indicate the reverse fluor arrays. The log-ratio will be computed as log(green/red) instead of log(red/green) for the reverse fluor arrays.

Collating Affymetrix data from MAS output files

BRB-ArrayTools has a dialog especially designed for easy importing of Affymetrix data. To access this tool, go to the **ArrayTools → Collate data → Affymetrix GeneChips** menu item.

If the user has Affymetrix data, then the CHP files should be exported from MAS 4.0 or 5.0 as tab-delimited text files in either the “horizontally aligned” or “separate files” input data formats as previously described.

Single chip type

The single chip type refers to the situation when all the chips which were hybridized were of the same type and format, containing the same probesets. Experiments which used only chip ‘A’ of a multi-chip set can also be considered as a single chip type experiment.

If data is exported in the horizontally aligned input data format, then a Pivot Table should be created containing the absolute analysis data from all experiments that were performed within that chip type, and the pivot table should be exported as a tab-delimited text file.

If the data is exported in the “separate files” format, then the Metrics Table data from each CHP file should be exported as a separate tab-delimited text file in a data folder devoted exclusively to expression data files. The files may contain miscellaneous rows before the column header row, but all files must have the exact same format (and same number of miscellaneous rows before the column header row). Each file should contain data from one chip or array, and the names of the separate files (without the “.xls” or “.txt” file extensions) will be used as experiment names in the collated project workbook.

In all of the following formats, the “Probe Set Name”, “Signal” and “Detection” are the only required columns from MAS 5.0 data, and the “Probe Set Name”, “Avg Diff” and “Abs_Call” are the only required columns from MAS 4.0 data. All other columns are optional.

BRB-Arraytools can automatically recognize the data format of files that have been exported directly from MAS 4.0 and MAS 5.0. However, users who choose to edit the data columns and column headers of their files should follow the format described below, in order for BRB-ArrayTools to automatically recognize the data format. If the column header row is not in any of the following formats, then BRB-ArrayTools will not be able to parse the files automatically. In that case, the user will need to use either the ‘horizontally aligned’ or ‘separate files’ collating dialogs (for a single array type) or the “multi-chip sets” collating dialog (for multi-array sets), in order to collate the data.

<u>Expression data in a “horizontally aligned” file:</u>

1. <u>For expression data output from a Pivot Table in MAS 5.0:</u>

Probe Set Name	Description	Exp1_ Signal	Exp1_Detection	Exp2_Signal	Exp2_Detection
1053_at	Human replication	234	P	456	P
1773_at	Human farnesyl-	123	P	39	P

2. <u>For expression data output from a Pivot Table in MAS 4.0:</u>

Probe Set Name	Description	Exp1_Avg Diff	Exp1_Abs_Call	Exp2_Avg Diff	Exp2_Abs_Call
1053_at	Human replication	234	P	456	P
1773_at	Human farnesyl-	123	P	39	P

For the above two formats, the prefixes Exp1 and Exp2 will be used as experiment names in the collated project workbook. You may use any other experiment names, but do not use the following special characters “\ / : * ? “ < > | .” which have special meanings in Windows. The “Probe Set Name” column label may also be given as simply “Probe Set”. The detection (Abs_call) column should be right next to the signal (Avg Diff) column for each experiment.

<u>Expression data in a “Separate Files” format:</u>

1. <u>For expression data output from a Metric Table in MAS 5.0:</u>

Probe Set Name	Description	Signal	Detection
1053_at	Human replication	234	P
1773_at	Human farnesyl-	123	P

2. <u>For expression data output from a Metric Table in MAS 4.0:</u>

Probe Set Name	Description	Avg Diff	Abs_Call
1053_at	Human replication	234	P
1773_at	Human farnesyl-	123	P

3. <u>In addition to the above mentioned file formats, the following file format from the NCBI portal will also be recognized:</u>

ID_REF	VALUE	ABS_CALL
AFFX-MurIL2_at	-896	A
AFFX-MurIL10_at	682	A

4. <u>In addition to the above mentioned file formats, the following file format from mAdb will also be recognized:</u>

AffyId	Signal	Detection Call
A28102_at	7.002252579	A
AB000114_at	4.121015549	A

<u>Gene annotations file format:</u>

The gene annotations file is optional. If you have probe set annotation columns other than the Description column, please place them in a separate tab-delimited gene annotation file and use the following column header labels. For multi-chip sets, please place all your probe set annotation information for all chip types in just one file.

Probe Set ID or Probe Set Name	- the name of the probe set.
Title or Description	- the description or title of the probe set.
Unigene	- the Unigene Id.
Gene Symbol	- the gene symbol.
GenBank or SeqDerivedFrom	- the GeneBank Id or source of the sequence.
Map Location or Chromosomal Location	- the location of the gene in the chromosomal map.

<u>Experiment descriptors file:</u>

The experiment descriptors file should follow the same format as described in the previous section Experiment descriptors. The first row of the experiment descriptors file should be a column header row, and each subsequent row should represent an individual array. The first column of the experiment descriptors file should contain the experiment labels. When the expression data is horizontally aligned in one file, then the order of the rows in the experiment descriptors file will be assumed to correspond to the order of the array data blocks in the horizontally aligned expression data file. When the expression data is in a separate file for each array, then the experiment labels in the first column of the experiment descriptor sheet must correspond to the expression data filenames (without the ".xls" or ".txt" file extensions).

Multi-chip sets

<u>Expression data files:</u>

If data is exported in the horizontally aligned input data format, then a *separate* Pivot Table should be created *for each chip type*, containing the absolute analysis data from all experiments which were performed of that chip type, and each pivot table should be exported as a tab-delimited text file.

If the data is exported in the "separate files" format, then the Metrics Table data from each CHP file should be exported as a separate tab-delimited text file, and the files should be named and organized into folders as described in "Example 3 – Multi-chip sets" above (i.e., the files should be organized into separate expression data folders for each chip type, and the same experiment should have identical filenames within each data folder except for a possible suffix of "_A", "_B", "_C", "_D", or "_E").

Gene annotations file format:

The gene annotations file is optional, and the file format is the same as described above for the single chip type. Probesets for all chip types in a multi-chip set should be listed in a single gene annotations file, when the gene annotations file is provided.

Experiment descriptors file:

The experiment descriptors file should follow the same format as described in the previous sections: Experiment descriptors and Example 3 – Multi-chip sets. The first row of the experiment descriptors file should be a column header row, and each subsequent row should represent an individual array. The first column of the experiment descriptors file should contain the experiment labels. When the expression data is horizontally aligned in one file, then the order of the rows in the experiment descriptors file will be assumed to correspond to the order of the array data blocks in the horizontally aligned expression data file. When the expression data is in a separate file for each array, then the experiment labels in the first column of the experiment descriptor sheet must correspond to the expression data filenames without the optional "_A", "_B", "_C", "_D", and "_E" suffixes and without the ".xls" or ".txt" file extensions.

Collating data from an NCI mAdb archive

BRB-ArrayTools has a collating interface that allows the National Cancer Institute Advanced Technology Center users to easily collate their data that has been downloaded from the "mAdb" website as a zipped archive. To access the "mAdb" collating interface tool, go to the **ArrayTools → Collate data → NCI Microarray Database (mAdb) archive** menu item. Currently, this collating interface is only implemented for dual-channel cDNA data, and Affymetrix data from a single chip type. The "mAdb" collating interface will be modified in the future to handle Affymetrix multi-chip data as well. For now, however, if the user has an "mAdb" archive that contains Affymetrix multi-chip data, then the user will need to use either the **Multi-chip sets** or **Affymetrix GeneChips** collating dialog.

With the "mAdb" collating interface, the user only needs to browse for the folder that contains the unpacked "mAdb" archive, and for dual-channel data, specify whether or not the genes are aligned and whether or not the data contains reverse fluor experiments. If the data contains reverse fluor experiments, then the user should first edit the `array_descriptions` file by entering a reverse fluor indicator in column "G", in which reversed arrays are denoted by the label "Yes".

The zipped archive should be unpacked before collating. When unpacking the archive, please check to make sure the directory structure of the archive has been preserved.

When the data for each array is in a separate file, there should be two files

```
array_descriptions_xxx_xxxxxx.xls
gene_identifiers_xxx_xxxxxx.xls
```

and a folder

```
array_data_xxx_xxxxxx
```

which are unpacked into the same directory. The `array_data_xxx_xxxxxx` folder contains all the expression data files, where each file contains the expression data for a single array.

Output of the collating step

Organization of the project folder

During the collating step, BRB-ArrayTools will create a **project folder** either within or alongside the folder that contained the user's original raw data files. This project folder will contain the **collated project workbook**, as well as supporting files needed by the collated project workbook. The new project folder will contain a `BinaryData` folder, and possibly also an `Annotations` or `Output` folder. The `Annotations` folder is created if the user chooses to lookup gene annotations from the Stanford SOURCE database. Some BRB-ArrayTools analyses may also produce various output files that will automatically be written to the `Output` folder. For example, all the plots that appear onscreen in the **Cluster viewer** page during cluster analysis are also automatically saved into the `Output` folder, and may be subsequently edited by the user for publication.

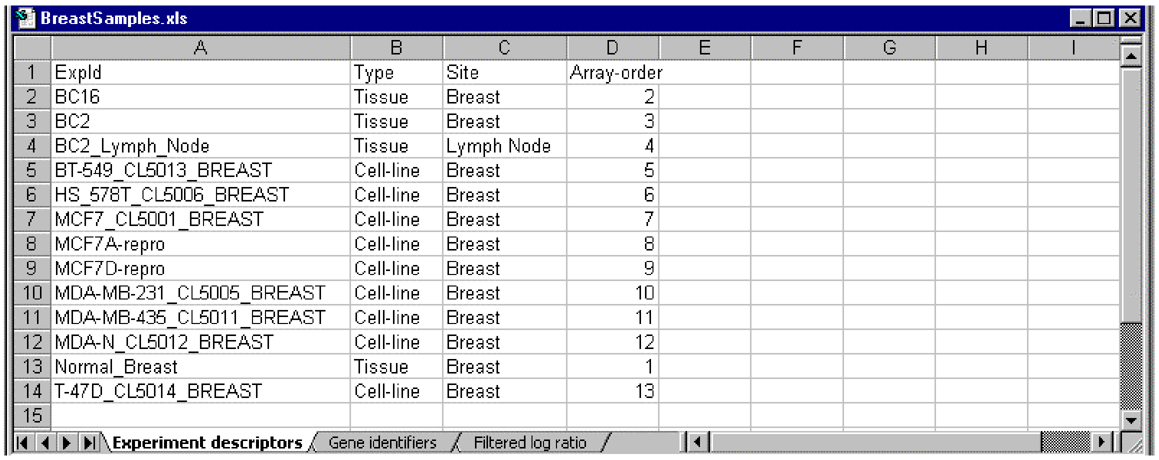
Please note that *only one data project should be collated into a single project folder*. Please use a separate project folder for each collated project, and only re-use an existing project folder if you intend to overwrite the existing collated project within that folder. If you are collating into an existing project folder, and the project folder already contains an existing `Annotations` folder, then the gene identifiers in the existing `Annotations` folder will be compared against the gene identifiers in your new collated project. If the set of gene identifiers in the existing `Annotations` folder match up exactly with the set of gene identifiers in your new collated project, then you will be given the option of importing the existing gene annotations into your new collated project workbook, so that you will not need to lookup the Stanford SOURCE database to get those annotations.

The collated project workbook

The collating procedure produces a collated project workbook. This is an Excel workbook containing three worksheets labeled **Experiment descriptors**, **Gene identifiers**, and either **filtered log-ratio** (for dual-channel data) or **filtered log intensity** (for single-channel data). All other data variables, such as the red and green background-adjusted signals and raw log-ratios (for dual-channel data), the raw log-intensities (for single-channel data), the spot flag (or Detection Call, for Affymetrix data), and the spot size, are considered auxiliary variables and will be written to separate text or binary files within the project folder.

The rows in the **gene identifiers** worksheet correspond to the rows in the filtered log-ratio (or filtered log intensity) worksheet. The last column of the gene identifiers and filtered log-ratio (or filtered log intensity) worksheets contains an internal filtering variable called “Filter” containing the labels TRUE or FALSE. To change the filtering criteria for any given analysis, the user should *not* manually edit the “Filter” column in these worksheets, but should set the filtering criteria by clicking on the **Re-filter the data** menu item. Re-filtering the data will cause BRB-ArrayTools to automatically adjust the TRUE/FALSE labels in the “Filter” column according to the filtering criteria chosen.

The **experiment descriptors** worksheet of the collated project workbook contains an *editable copy* of the experiment descriptors file which was collated with the data. This copy can be edited by the user to add descriptors to be used in later analyses, as in the example below:

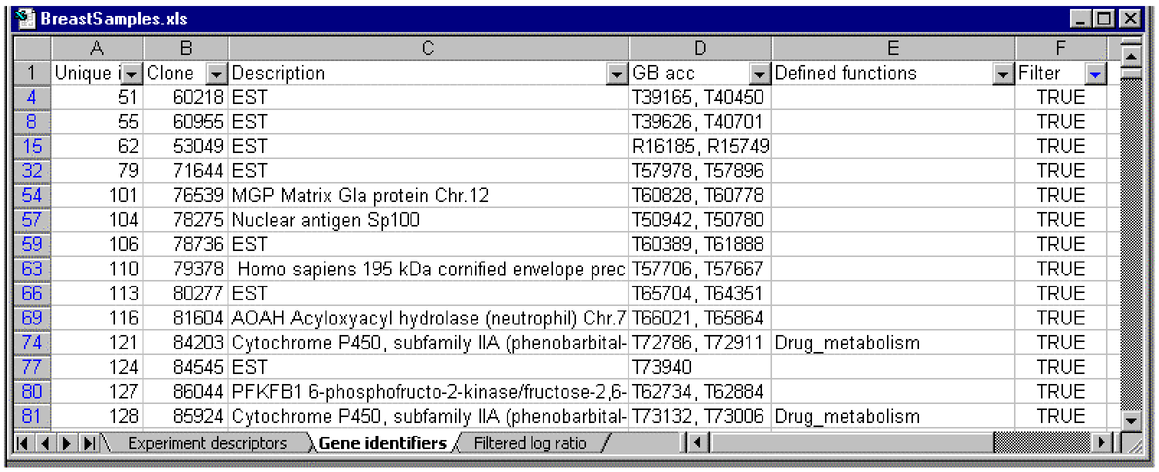

BreastSamples.xls

	A	B	C	D	E	F	G	H	I
1	ExpId	Type	Site	Array-order					
2	BC16	Tissue	Breast	2					
3	BC2	Tissue	Breast	3					
4	BC2_Lymph_Node	Tissue	Lymph Node	4					
5	BT-549_CL5013_BREAST	Cell-line	Breast	5					
6	HS_578T_CL5006_BREAST	Cell-line	Breast	6					
7	MCF7_CL5001_BREAST	Cell-line	Breast	7					
8	MCF7A-repro	Cell-line	Breast	8					
9	MCF7D-repro	Cell-line	Breast	9					
10	MDA-MB-231_CL5005_BREAST	Cell-line	Breast	10					
11	MDA-MB-435_CL5011_BREAST	Cell-line	Breast	11					
12	MDA-N_CL5012_BREAST	Cell-line	Breast	12					
13	Normal_Breast	Tissue	Breast	1					
14	T-47D_CL5014_BREAST	Cell-line	Breast	13					
15									

Experiment descriptors / Gene identifiers / Filtered log ratio

The **Array-order** descriptor variable, for example, might be used to specify the order of experiments to use when plotting lineplots of gene cluster expression.

The **gene identifiers** worksheet of the collated project workbook contains the gene identifiers whose columns had been specified in the **column format** section under the **gene identifiers** page of the collating dialog box.


BreastSamples.xls

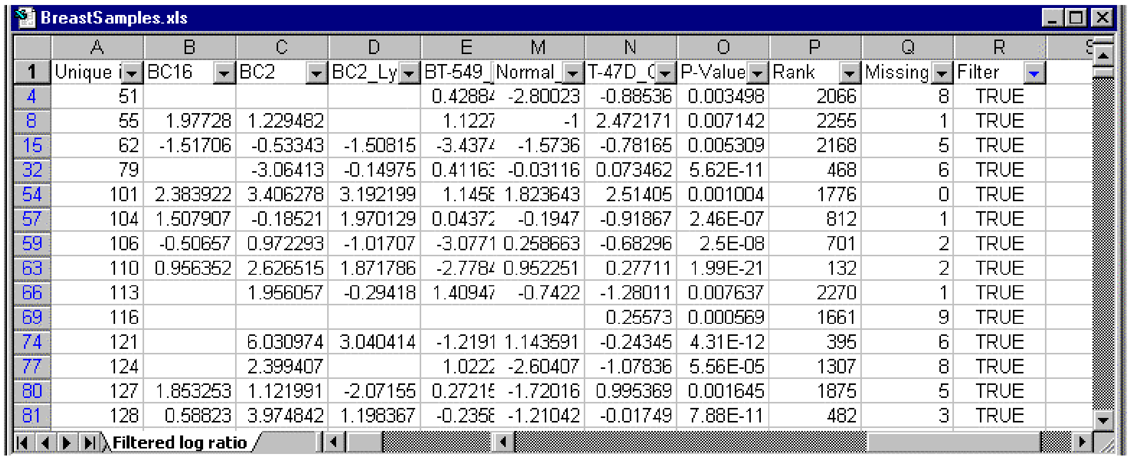
	A	B	C	D	E	F
1	Unique i	Clone	Description	GB acc	Defined functions	Filter
4	51	60218	EST	T39165, T40450		TRUE
8	55	60955	EST	T39626, T40701		TRUE
15	62	53049	EST	R16185, R15749		TRUE
32	79	71644	EST	T57978, T57896		TRUE
54	101	76539	MGP Matrix Gla protein Chr.12	T60828, T60778		TRUE
57	104	78275	Nuclear antigen Sp100	T50942, T50780		TRUE
59	106	78736	EST	T60389, T61888		TRUE
63	110	79378	Homo sapiens 195 kDa cornified envelope prec	T57706, T57667		TRUE
66	113	80277	EST	T65704, T64351		TRUE
69	116	81604	AOAH Acyloxyacyl hydrolase (neutrophil) Chr.7	T66021, T65864		TRUE
74	121	84203	Cytochrome P450, subfamily IIA (phenobarbital-	T72786, T72911	Drug_metabolism	TRUE
77	124	84545	EST	T73940		TRUE
80	127	86044	PFKFB1 6-phosphofructo-2-kinase/fructose-2,6-	T62734, T62884		TRUE
81	128	85924	Cytochrome P450, subfamily IIA (phenobarbital-	T73132, T73006	Drug_metabolism	TRUE

Experiment descriptors / Gene identifiers / Filtered log ratio

If the user has chosen to annotate his or data using the defined genelists, then the gene identifiers worksheet will also contain a column labeled **Defined genelists**, indicating if the genes matched any of those listed in the **user defined gene lists**.

The click-arrows in each cell of the header row may be used to search for specific values of each variable. Column-widths may be adjusted by hovering the cursor between the columns in the lettered column header bar and dragging.

The **filtered log ratio** (or **filtered log intensity**) worksheet contains the log-ratios (or log intensities) *after* the specified filtering levels have been applied. These are the data values that will be used for all analyses except the scatterplot of experiment v. experiment, where the log of the channel intensities can also be plotted for dual-channel data. The first column of the worksheet contains the primary gene id, which matches the gene id listed in the first column of the gene identifiers worksheet. Each succeeding column contains the log-ratios obtained from a single experiment. The highlighted yellow columns at the end of the dataset are used internally by BRB-ArrayTools for gene-filtering purposes. The user should not edit this worksheet in any way. BRB-ArrayTools will automatically re-filter this worksheet, if necessary, before running each analysis.


BreastSamples.xls

	A	B	C	D	E	M	N	O	P	Q	R
1	Unique i	BC16	BC2	BC2_Ly	BT-549_	Normal_	T-47D_(	P-Value	Rank	Missing	Filter
4	51				0.4288	-2.80023	-0.88536	0.003498	2066	8	TRUE
8	55	1.97728	1.229482		1.1227	-1	2.472171	0.007142	2255	1	TRUE
15	62	-1.51706	-0.53343	-1.50815	-3.4374	-1.5736	-0.78165	0.005309	2168	5	TRUE
32	79		-3.06413	-0.14975	0.41163	-0.03116	0.073462	5.62E-11	468	6	TRUE
54	101	2.383922	3.406278	3.192199	1.1458	1.823643	2.51405	0.001004	1776	0	TRUE
57	104	1.507907	-0.18521	1.970129	0.04372	-0.1947	-0.91867	2.46E-07	812	1	TRUE
59	106	-0.50657	0.972293	-1.01707	-3.0771	0.258663	-0.68296	2.5E-08	701	2	TRUE
63	110	0.956352	2.626515	1.871786	-2.7784	0.952251	0.27711	1.99E-21	132	2	TRUE
66	113		1.956057	-0.29418	1.40947	-0.7422	-1.28011	0.007637	2270	1	TRUE
69	116						0.25573	0.000569	1661	9	TRUE
74	121		6.030974	3.040414	-1.2191	1.143591	-0.24345	4.31E-12	395	6	TRUE
77	124		2.399407		1.0222	-2.60407	-1.07836	5.56E-05	1307	8	TRUE
80	127	1.853253	1.121991	-2.07155	0.27215	-1.72016	0.995369	0.001645	1875	5	TRUE
81	128	0.58823	3.974842	1.198367	-0.2358	-1.21042	-0.01749	7.88E-11	482	3	TRUE

Filtering the data

The filtering criteria are specified during the data collating step, and may be changed prior to each analysis. The dialog box for each analysis has a **Filter** button on the bottom, which can be used to specify or change the filtering criteria for all subsequent analyses.

Although the filtering criteria may be changed for each analysis, it may be important to check the filtering criteria before collating the data *if* the data contains replicate spots for some genes in the array. This is because the **spot filtering** is applied *before* averaging the replicate spots, and replicate spots are *not* re-averaged after the collating step has been completed, even if the user changes the filtering criteria.

The **Filtered log ratio** (or **Filtered log intensity**) worksheet of the collated project workbook will reflect the filtering actions which have been selected. With the exception of replicated spots, which are filtered before averaging, the supporting files that are located in the project folder will *not* reflect the filtering actions that have been selected.

The order of operations for filtering the data is that spot filters are applied first, then data normalization, then truncation of extreme values, then gene screening.

Spot filters

Spot filtering refers to filters on spots in individual arrays. Spot filtering is used for quality-control purposes, to filter out "bad" spots. Unlike gene screening, spot filtering does not filter out the entire gene (row), but replaces the existing value of a spot within any given array with a missing value.

Intensity filter

For dual-channel data, a spot may either be filtered out (excluded) from the analysis, or thresholded (with one signal set equal to the specified minimum) in the analysis. Using the appropriate filtering option, a spot on any array that has a red signal less than the specified minimum and/or a green signal less than the specified minimum will have its log-ratio value filtered out on the **filtered log ratio** worksheet. Using the thresholding option, a spot on any array which has BOTH signals less than the specified minimum will be filtered out, but a spot that has only ONE signal less than the specified minimum will have the below-minimum value set equal to the specified minimum, in computing the log-ratio value that is shown in the **filtered log ratio** worksheet. The intensity filter is applied to the *background-adjusted* red and green signals.

For single-channel data, a spot on any array, which has a signal intensity less than the specified minimum, may either be filtered out or thresholded on the **filtered log intensity** worksheet. Please note that the signal intensity values have already been automatically thresholded to 1 by default, unless the “Do not threshold to 1 (e.g., CodeLink)” checkbox was selected at the time of collating. If the “Do not threshold to 1” option was selected at the time of collating, then negative and 0 signal intensities have already been set to missing, and signal intensities in the interval between 0 and 1 have been left as-is. Please note that the “Do not threshold to 1” option is IRREVERSIBLE once the data has been collated! Once the negative or 0 values have been set to missing, they can never be thresholded again to 1 by re-filtering the data. Likewise, once the values less than 1 have been thresholded to 1, the negative or 0 values can never be separated from the values in the interval between 0 and 1 and be subsequently set to missing. In order to change the “Do not threshold to 1” option, the data must be re-collated.

Spot flag filter

The spot flag filter can contain both numeric and character values. The user may specify a numeric range outside of which a numeric flag is considered to be “excluded”, and/or specify a list of flag values denoting the “excluded” values. When the flag filter is on, a spot on any array which has an “excluded” flag value will be filtered out on the **filtered log ratio** (or **filtered log intensity**) worksheet. The flag field is optional.

For Affymetrix users, a Detection Call column can be designated as the spot flag column at the time of collating, allowing users to filter out expression values that have an “A” (Absent) call. Additionally, the spot flag column (or Detection Call column, for Affymetrix users) is also used by the Percent Absent Filter, to filter out spots (or probesets) with a large percentage of expression values that have a spot flag (or Detection Call) value of “A”.

Spot size filter

The spot size field is optional, but when the spot size filter is on and the spot field is present, then a spot on any array which has a spot size less than the minimum value will be filtered out on the **filtered log ratio** (or **filtered log intensity**) worksheet. For Affymetrix data, the number of pairs used to compute the signal can be used as a surrogate for the spot size measurement.

Detection call filter

For data collated using the **Affymetrix GeneChips** collating interface, the Detection call filter is a special case of the spot flag filter mentioned above, allowing users to filter out expression values based on the Detection call. The Detection data column is also used by the Percent Absent Filter to filter out probesets with a large percentage of Detection Calls that have a value of “A”.

Transformations

A logarithmic (base 2) transformation is applied to the signal intensities (for single-channel data) or intensity-ratios (for dual-channel data) before they are normalized and truncated. The effect of the data normalization and truncation will be reflected in the **Filtered log ratio** (or **Filtered log intensity**) worksheet of the collated project workbook.

Normalization

There are currently three normalization options: median normalization, housekeeping gene normalization, and lowess normalization. The median normalization and housekeeping gene normalization options are available for both single-channel and dual-channel data, but the lowess normalization option is available only for dual-channel data. For single-channel data, the user must choose a reference array against which all other arrays will be normalized. Each array in a multi-chip set is normalized separately. Data exported from MAS5 is usually already normalized, and hence the BRB-ArrayTools normalization can be de-selected. Note the method of normalization for single-channel data has been changed in v3.1 from previous versions of BRB-ArrayTools.

Selecting a reference array

For single-channel data, the user has the option to explicitly select one of the arrays to be the reference array, or ask BRB-ArrayTools to automatically select the "median" array as the reference array. The algorithm which BRB-ArrayTools uses to select the "median" array is as follows:

1) Let N be the number of experiments, and let *i* be an index of experiments running from 1 to N.
2) For each array *i*, the median log-intensity of the array (denoted M_i) will be computed.
3) A median M will be selected from the $\{M_1, \ldots, M_N\}$ values. If N is even, then the median M will be the lower of the two middle values.
4) The array whose median log-intensity M_i equals the overall median M will be chosen as the median array.

Median normalization

The median-normalization of dual-channel data is performed by subtracting out the *median log-ratio* for each array, so that each normalized array has a median log-ratio of 0. The median-normalization of single-channel (e.g., Affymetrix) data is performed by computing a gene-by-gene *difference* between each array and the reference array, and subtracing the *median difference* from the log-intensities on that array, so that the gene-by-gene difference between the normalized array and the reference array is 0.

Housekeeping gene normalization

The user may specify a housekeeping genelist that will be used for normalization. The housekeeping genes normalization of dual-channel data is performed by subtracting out the *median log-ratio over housekeeping genes* from all the log-ratios on the array. The housekeeping genes normalization of single-channel data is performed by computing a gene-by-gene *difference* between each array and the reference array, and subtracing the *median difference over housekeeping genes* from the log-intensities on that array.

Lowess normalization

For dual-channel data, a lowess (or intensity-dependent) normalization option is also available. For dual-channel data, median normalization is equivalent to multiplying all of the intensities in one channel of an array by a normalization factor. In some cases it can be advantageous to have a different normalization factor for different intensity levels; i.e., the dye bias may be different for low intensity spots relative to high intensity spots. In the lowess normalization, a non-linear lowess smoother function is fit to the graph of un-normalized log-ratio on the y-axis versus average log intensity (i.e., [log(Red)+log(Green)]/2) on the x-axis. This is the so-called M-A plot. The lowess smoother is based on a concatenation of linear regressions for points in overlapping intervals on the x-axis of the M-A plot. This lowess smoother is subtracted from the un-normalized log-ratios for the array in order to obtain the normalized log-ratios. The lowess normalization is much more computationally intensive than the median centering normalization and may require up to 10 seconds per array on some computers using data with 35,000 genes. You may wish to first normalize your data using the default method and then examine M-A scatterplots to determine whether intensity dependent normalization is needed. If most of the points of the M-A plot are distributed equally above and below the 0 value of the y-axis, without a trend over x-values, then intensity based normalization is not needed.

The default span parameter for the lowess smoother is now set to 2/5, beginning in version 3.0.1. This means that 2/5 of the total set of data points will influence the smooth at each value. Larger span values give more smoothness. In previous versions, the default span value had been set to 2/3, but this was found to create overly-smoothed curves which did not sufficiently capture the trends which were often found in the tails of the M-A plot. The user is now able to change the default span value by editing the “LowessSpan” parameter value in the “Preferences.txt” file in the “Prefs” folder of the ArrayTools installation folder. The “Preferences.txt” file is a tab-delimited file in which parameter names are stored in the first column and parameter values are stored in the second column.

Truncation

This option allows the user to specify a maximum intensity (for single-channel data) or intensity ratio (for dual-channel data) to be used for analysis. Any values greater than the specified threshold will be truncated to the threshold. For truncation of intensity ratios, both the intensity ratios and inverse intensity ratios will be truncated (e.g., an intensity ratio threshold of 64 means that all intensity ratios will be truncated to lie between 1/64 and 64). Truncation is primarily used for dual-channel data, where small denominators can cause intensity ratios to become enormous.

Gene filters

Gene filtering, unlike spot filtering, is not applied on an array-by-array basis for each gene. Instead, it uses a criterion based on *all* arrays for a given gene, to determine if that gene should be screened out or not. Its purpose is not to filter out "bad" spots, but rather to screen out genes that are not likely to be informative. The last column (labeled **Filter**) on the **Gene identifiers** and **Filtered log ratio** (or **Filtered log intensity**) worksheets are internally used by BRB-ArrayTools to select the genes that pass the screening.

Here the criterion for filtering out a gene is based upon the variability of its log expression values across all arrays after normalization. Filtering low variance genes is not really necessary except for clustering genes, where the computer memory requirements increase rapidly with the number of genes clustered. Several filtering options are available.

Minimum fold-change filter

Genes that have low variability may be filtered out using the minimum fold-change filter. Here the criterion for filtering out a gene is based upon the percentage of expression values for that gene which have at least a minimum fold-change from the median expression value for that gene. The user may specify the minimum fold-change that is required. If less than a specified percentage of expression values meet the minimum fold-change requirement, then the gene is filtered out.

Log expression variation filter

Alternatively, the filtering can be based on the variance for the gene across the arrays. One can exclude the x% of the genes with the smallest variances, where the percentile x is specified by the user. Or a statistical significance criterion based on the variance can be used. If the significance criterion is chosen, then the variance of the log-ratios for each gene is compared to the *median* of all the variances. Those genes not significantly more variable than the median gene are filtered out. The significance level threshold may be specified by the user. Specifically, the quantity (n-1) Var_i / Var_{med} is computed for each gene i. Var_i is the variance of the log intensity for gene i across the entire set of n arrays and Var_{med} is the median of these gene-specific variances. This quantity is compared to a percentile of the chi-square distribution with n-1 degrees of freedom. This is an approximate test of the hypothesis that gene i has the same variance as the median variance.

Percent missing filter

Here the criterion for filtering out a gene is based upon the percentage of expression values that are not missing and not filtered out by any of the previous spot filters.

Percent absent filter

Here the criterion for filtering out a gene is based upon the percentage of Absent calls in the Spot Flag or Detection variable. This gene filter may be applied *independently* of the Spot Flag Filter or Detection Call Filter described in the "Spot Filters" section above, though it uses the same Spot Flag or Detection Call variable. For instance, a user may choose to turn *off* the Detection Call Filter in order to preserve all "Absent" expression values as they are, but turn *on* the Percent Absent Filter in order to exclude probesets that are considered unreliable or uninteresting because too many of the expression values were "Absent". In this case, probesets that did not get excluded by the Percent Absent Filter still maintain their expression values even for those values that were "Absent".

Gene subsets

Gene subsetting, unlike gene filtering, is not based on the expression data values for the genes, but rather on the identites of the genes. The purpose of gene subsetting is to select or exclude genes which are known to be interesting or non-informative based on one of the gene labels.

Selecting a genelist to use or to exclude

The user may create user-defined genelists to define sets of genes. The user may select one or more genelists which define a gene subset, and filter the data to include only those genes in the selected subset (or exclude the genes from that subset).

Specifying gene labels to exclude

The user may choose to exclude genes by specifying a string within one of the gene identifiers as an exclusion criterion. For instance, a user may choose to exclude all empty wells, so the user may choose to exclude all genes with "Empty well" in the Description column of the Gene identifiers worksheet.

Annotating the data

Gene annotations

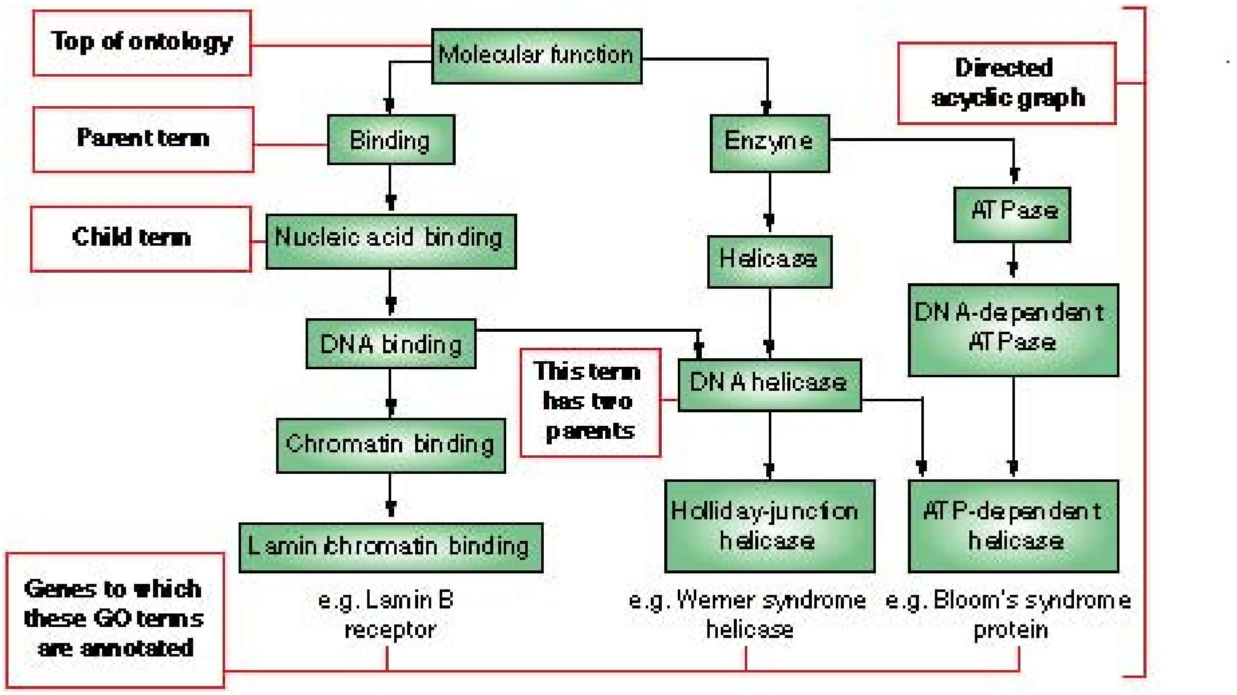
Data which is imported as dual-channel or single-channel non-Affymetrix data can be annotated by using the SOURCE annotation tool. This tool is found under the following menu item: **ArrayTools → Utilities → Annotate data → Import SOURCE annotations.**

Data which is imported as Affymetrix can be annotated by downloading pre-formatted Affymetrix annotations files from the BRB server. This automatic annotation appears as an option in the **Data import wizard** as well as the **Special format: Affymetrix GeneChips** collation dialogs. If the user did not annotate the data at the time of collating or wishes to update the annotations which have already been imported into the project workbook, the user may run the Affymetrix annotations tool at any later time. This tool is found under the following menu item: **ArrayTools → Utilities → Annotate data → Import Affymetrix annotations**. The Affymetrix annotations files on the BRB server will be updated periodically (approximately once a month), so the user may wish to update the local Affymetrix annotations files by running this tool periodically.

The annotations downloaded by either the SOURCE annotations tool or the Affymetrix annotations tool will be stored in a **Gene annotations** worksheet inside the project workbook.

Gene ontology

Gene Ontology is a structure, controlled vocabulary used to describe gene products in terms of their associated biological processes, cellular components and molecular functions. For more information about Gene Ontology, please browse to http://www.geneontology.org. The vocabularies exhibit a complex structure, represented as directed acyclic graphs (DAGs) or networks. The following is an example from the http://www.geneontology.org/doc/GO.doc.html#process .



BRBArray Tools maintains the Gene Ontology structure files in the user's computer by providing a tool for downloading these structure files from the GO database. The tool can be found in the menu item: **ArrayTools → Utilities → Gene Ontology → Download ontology structure**. Since the ontology structure files are released monthly by the GO database, it is recommended that the user download these files regularly to keep the ontology files on the local computer updated. BRB-ArrayTools will automatically check the file dates and prompt the user to re-download these files when the files are 30 days old.

If the SOURCE or Affymetrix annotation tools have already been run on the dataset, then BRB-ArrayTools can automatically generate the complete Gene Ontology information for all genes in the dataset using the structure files downloaded from the Gene Ontology database. The gene ontology structure files are used to generate the observed v. expected frequencies of selected genelists in the output of various analysis tools such as the class comparison, class prediction, survival and quantitative traits analysis tools.

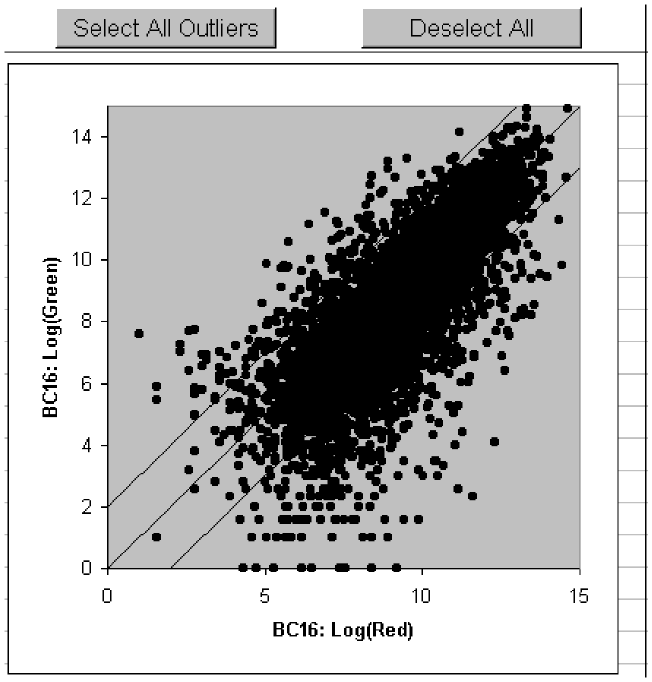
Analyzing the data

Scatterplot tools

Scatterplot of single experiment versus experiment

It is frequently of interest to plot the log-ratios of one experiment versus the log-ratios of another experiment. Generally, most of the plotted points, which represent genes with nonmissing values in both experiments, will line up along a 45-degree diagonal line. All genes that pass the filtering criteria will be used in the scatterplot, unless a genelist has been selected. Genes that are differentially expressed between the two experiments will fall outside of a pair of outlier lines. These outlier lines can be specified by the user to indicate genes whose expression ratios in the two experiments are larger than a given fold-difference.

The user may click on individual points in order to identify its associated gene and to hyperlink to annotated clone reports within NCBI database. Or, all the genes outside of the outlier lines can be simultaneously selected by clicking on a button.



For dual-channel experiments, it may also be of interest to plot the log-intensity of one channel versus the log-intensity of the other channel, or to plot the reference sample in one array versus the reference sample in another array (to check reproducibility). Such plots might reveal any of the following descriptive features: whether or not one channel is uniformly higher than the other, whether the two channels are linearly related (they should be if most genes are not differentially expressed between the two samples), and whether or not there are points that are "clumped" away from the mass of points. A plot of log(Red) versus log(Green) within the same array will often show a "fanning" out effect at the lower end, showing that low-intensity spots are generally less reproducible than the higher-intensity spots. These plots can sometimes be used to help determine appropriate thresholds for the intensity filter.

For dual-channel experiments it may be easier to detect non-linearities that represent inadequate normalization by examining the so-called M-A plot, rather than the plot of log(Red) versus log(Green). The M-A plot is a graph of the normalized log-ratio on the y-axis versus average log intensity (i.e., [log(Red)+log(Green)]/2) on the x-axis. Most of the points of the M-A plot should be distributed equally above and below the 0 value on the y-axis, without a trend over x-values. If there is a trend, then intensity-based normalization may be needed.

Scatterplot of phenotype averages

To compare the experiments in one phenotype class versus the experiments in another phenotype class, BRB-ArrayTools provides a tool that plots the average log-ratio within one class on the x-axis versus the average log-ratio within the other class on the y-axis. These averages are taken on a gene-by-gene basis, and each gene is represented by a single point in the resulting scatterplot. The experiment descriptor sheet must contain a descriptor variable containing the class labels to be compared. A blank label in this column means that the corresponding experiment will be omitted from the analysis. For this analysis, the column should contain only two unique class labels, and all labels belonging to other classes should be "blanked" out by the user.

Genes that are differentially expressed between the two phenotypes will fall outside of a pair of outlier lines. These outlier lines can be specified by the user to indicate genes for that the fold-difference between the geometric mean of the expression ratios within each of the two classes is greater than a specified amount.

As with the scatterplot tool for a single experiment versus experiment, the user may click on individual points in order to identify its associated gene and to hyperlink to annotated clone reports within NCBI database. Or, all the genes outside of the outlier lines can be simultaneously selected by clicking on a button.

Hierarchical cluster analysis tools

For a given dataset, hierarchical clustering can occur in two directions: either the genes can be clustered by comparing their expression profiles across the set of samples, or the samples can be clustered by comparing their expression profiles across the set of genes. In BRB-ArrayTools, the data can be clustered in either one or both directions independently. The objects to be clustered may be either the genes or the samples.

Hierarchical clustering produces a sequentially nested merging of the genes (or samples) determined by a defined measure of pair-wise similarity or distance between expression profiles. The nested merging is represented by a “dendrogram”. At the lowest level of the dendrogram, each gene (or sample) is a member of an individual singleton cluster. At the first step, the genes (or samples) with the two expression profiles most similar to each other are merged into a cluster. Then the next two most similar genes (or samples) are joined as a cluster. At each step, the most similar two clusters (including singleton clusters) are joined to form a larger cluster. This might actually involve the joining of two singleton genes (or samples), merging a gene (or sample) into an existing cluster, or merging two clusters formed at a previous step. The “distance” between two clusters which merge into a single cluster can be read from the scale along side of the dendrogram. Clusters that are merged low on the dendrogram are similar, whereas clusters which are formed by mergers high on the dendrogram may be very heterogeneous. At the top level of the dendrogram, there is a single cluster containing all of the genes (or samples). At the lowest level of the dendrogram the “clusters” are very homogeneous because they consist of singleton genes (or samples).

The tool for hierarchical clustering of genes can be used for perform a cluster analysis of the genes alone, or a cluster analysis of both the genes and the samples, where the output includes an image plot of the log-ratio values where the genes and the samples are sorted according to their dendrogram order. These two tools will be discussed separately below.

When performing a cluster analysis of genes or samples, BRB-ArrayTools will create a **Cluster viewer** worksheet in the collated project workbook, which will display the dendrogram plot. To continue viewing the sequence of plots, click on the **Next** button at the top of the page. Clicking on the **Next** button may also bring up a prompt for the user to define discrete clusters from the dendrogram by “cutting” the tree, if the analyses selected by the users require the definition of distinct clusters. All plots that are shown in the **Cluster viewer** are also automatically saved as JPEG files in the `Output` folder of the project folder, and may be subsequently edited using other commercially available graphics software.

Several options are available in the clustering procedure for the distance metric and linkage method.

Distance metric

The options currently available for the distance metric are: “one minus (centered) correlation” and “Euclidean distance”. For clustering of samples only, the “one minus uncentered correlation” is also offered as an additional option.

A Pearson (centered) correlation between two experiments X and Y is defined as

$$\Sigma_{i=1 \text{ to } N} (X_i - X_{avg})(Y_i - Y_{avg}) \,/\, [\, (\Sigma_{i=1 \text{ to } N} (X_i - X_{avg})^2)\, (\Sigma_{i=1 \text{ to } N} (Y_i - Y_{avg})^2)]^{1/2},$$

where the summation index *i* runs through the N genes in the two experiments, and X_{avg} is the mean over all the genes in experiment X, and Y_{avg} is the mean over all the genes in experiment Y. (The formula is the same for the Pearson correlation between two genes X and Y, except that the summation index *i* would run through the M experiments in the dataset.) Pearson correlation is a commonly used measure of similarity of two columns of numbers, and hence one minus Pearson correlation serves as a distance metric. The two columns used in computing correlation contain the normalized log-ratios or normalized log-intensities of the genes for the two samples being compared.

When the entire set of genes is not used for clustering samples, it may be appropriate to use the “uncentered correlation” metric. The uncentered correlation between two experiments X and Y is defined as

$$\Sigma_{i=1 \text{ to } N} X_i Y_i \,/\, [\, (\Sigma_{i=1 \text{ to } N} X_i^2)\, (\Sigma_{i=1 \text{ to } N} Y_i^2)\,]^{1/2},$$

where the summation index *i* runs through the N genes in the two experiments. If the complete set of genes is used and the experiments have been normalized, then the centered correlation between any two experiments is very similar to the uncentered correlation between those two experiments, because the means of the two normalized data columns are very similar.

Euclidean distance is a somewhat different metric. The Euclidean distance between two columns is equivalent to the root-mean-square error between the two columns; it is the square root of the sum of squared differences between the expression levels. The expression profiles of two samples have a large correlation if genes that are highly expressed relative to the mean in one are also highly expressed relative to the mean in the other. The Pearson correlation depends on the patterns of which genes are above and below the mean expression in each profile. Euclidean distance measures absolute differences between expression levels of individual genes rather than on patterns of whether similar genes are up or down regulated relative to each other. The Euclidean distance between two expression profiles is small only if the absolute expression levels of the genes in the two profiles are very similar. The Pearson correlation metric is most commonly used, but it may be worthwhile to examine clustering also with regard to Euclidean distance.

Linkage

The other option is whether to use the average linkage, complete linkage or single linkage version of the hierarchical clustering algorithm. The distance metric used defines the distance between the expression profiles of two samples. During hierarchical clustering, however, the algorithm must compute the distance between clusters formed at a previous step, or between a singleton and a previously created cluster. With average linkage clustering, the distance between two clusters is taken as the average of the distances between all pairs of elements, one from the first cluster and one from the second. With complete linkage clustering, the distance between two clusters is taken as the maximum distances between an element in the first cluster and an element in the second. With single linkage clustering, the distance between two clusters is taken as the minimum of these distances. Complete linkage clusters tend to be compact, with all members of a cluster relatively equally distant from each other. The results of complete linkage clusters, however, may be heavily dependent upon the elements that were used to start the clusters. Single linkage clusters tend to be "long and narrow" in multi-dimensional space, containing members very distant from each other but closer to some intermediate member. Average linkage clustering is most commonly used because it provides a compromise between the advantages and disadvantages of the other types.

Cluster analysis of genes (and samples)

The cluster analysis of genes will produce a dendrogram plot. After the user has "cut" the dendrogram tree, the following plots will be produced: an image plot of all the genes (a matrix of plotted colors varying from red-to-green or orange-to-blue, representing the over- or under-expression of each gene in each sample, where the rows in the image plot represent the genes, and the columns in the image plot represent the samples), a "median" image plot (similar to the previous plot, but where a row represents a cluster rather than a gene), and "profile" lineplots for each cluster (where the median log-ratio of that cluster in each experiment is plotted).

In the image plot of all the genes, the genes are sorted in dendrogram-order, so that genes that are next to each other when reading across the bottom of the dendrogram are also next to each other in the image plot. For all three plots, the user has the option to order the samples in the image plot based on a cluster analysis of the samples whose dendrogram-order will determine the plotting order of the samples, or based on a pre-defined experiment descriptor variable whose order will determine the plotting order of the samples. If a cluster analysis of the samples is chosen, then the user has the option to center the genes or not (see discussion under the next section "Options for cluster analysis of samples"). If the plotting order of the samples is based on a cluster analysis, then the user has the option of re-ordering the samples in each "profile" lineplot using a cluster analysis based only on the genes in that cluster.

Some other options include specifying the labeling of the samples, the ability to suppress printing of clusters that contain too few genes, and the color scheme (red/green or orange/blue) and color range of the image plots.

The genes composing each cluster are listed in the **Cluster listing** worksheet of the collated project workbook. The genes in each cluster can also be identified from **the Cluster viewer** worksheet by clicking on **List Genes**.

Cluster analysis of samples alone

Clustering samples is frequently performed using the entire set of genes that pass the established filter levels. In certain cases, the user may wish to use a special gene set for clustering and that is an option. There is an option for whether the genes should be median centered in computing the distance between pairs of samples. In most cDNA arrays using an internal reference channel, it is advisable to use the median gene centering option because it reduces the influence of the expression profile of the internal reference sample, which is often not itself of interest.

Another option in the cluster analysis of samples is the ability to compute reproducibility measures. Cluster analysis algorithms always produce clusters, but the clusters are not always reproducible or biologically meaningful. With gene clustering this is less of a problem because we know in advance that gene products are grouped into pathways. It may be a question whether genes in the same cluster are co-regulated, but the existence of biologically meaningful gene clusters is usually not open to question. This is not the case with clusters of samples based on disease tissue from different patients, however. The claim that there are "real" clusters in many cases represents the substantial claim that the disease is molecularly heterogeneous. Such a claim requires more evidence than the fact that the clustering algorithm produced clusters.

Probably the best evidence for reproducibility of clusters of samples is to demonstrate that the clusters that patients tissues are placed in remain the same when the analysis is repeated using RNA independently extracted from each of the same samples. In many cases, however, two experiments based on independently extracted RNA samples for each sample is not available. BRB-ArrayTools provides some alternative, though less compelling, measures of the reproducibility of the sample clusters.

The user may select the cluster reproducibility analysis in the dialog box for clustering samples. The reproducibility analysis is based on perturbing the normalized log-ratios (or normalized log intensities for single label oligonucleotide data) and re-clustering the perturbed data. Indices are computed for each cluster in the original data indicating how much the membership of the cluster changed based on the perturbation. The perturbation and re-clustering process is repeated many times (the number of repetitions is defined by the user) and the output is a summary over the replications of the indices of reproducibility for each cluster. The indices are not computed for all clusters in the dendrogram, but rather for all clusters defined by cutting the original dendrogram at a level defined by the user. A similar level is used for re-clustering the perturbed data in each replication.

Two reproducibility indices are reported. One is called the Reproducibility or R measure for each cluster. The R measure is based on pairs of samples in a cluster of the data. For each such pair of samples, the program computes the proportion of the replications that those two samples remain in the same cluster. That proportion, averaged over replications and over all pairs of samples in the same cluster, is the R measure for that cluster. An R of 1 means perfect reproducibility of that cluster. An R of 0 means no reproducibility of the cluster.

The other index used is the Discrepancy or D measure. Each cluster of the original data has a “target” cluster in a repetition of the perturbed data. The “target” cluster is that cluster of the perturbed data that contains the largest number of samples included in the original cluster. The program computes the number of discrepancies as the number of samples in the target cluster but not in the original cluster plus the number of samples in the original cluster but not in the target cluster. This number of discrepancies is computed for each original cluster and averaged over the repetitions.

The data is perturbed using Gaussian random noise. The user can specify the variance of the noise if he/she has independent data on the variation over arrays of measurements of independent labelings and hybridization of the same RNA sample. If the user does not specify a value of the variance to be used, the program computes the variance of normalized log-ratios (or normalized log intensities) across all of the arrays for each gene and uses the median variance for the generation of Gaussian perturbations. This estimation is reasonable as long as most genes are not differentially expressed across the arrays. The variance measure used may include some component of biological variation, rather than just experimental variation. The inclusion of biological variability is desirable, although true biological variability would be correlated among sets of genes and the perturbations simulated by the program are generated independently for each gene.

For more information about the cluster reproducibility analysis, see the paper “Methods of assessing reproducibility of clustering patterns observed in analyses of microarray data” by LM McShane, MD Radmacher, B Freidlin, R Yu, MC Li and R Simon in Bioinformatics 18:1462-1469, 2002 also available as a technical report at http://linus.nci.nih.gov/~brb/TechReport.htm

Interface to Cluster 3.0 and TreeView

BRB-ArrayTools now includes an interface to the Cluster 3.0 and TreeView software originally produced by the Stanford group (http://genome-www.stanford.edu). This interface allows the user to automatically send data from the BRB-ArrayTools project workbook directly into a Cluster and TreeView analysis. For instance, the user may choose to subset and manipulate the data within BRB-ArrayTools, but view the cluster analysis results in TreeView instead of using BRB-ArrayTools’ built-in clustering tools.

Multidimensional scaling of samples

Multi-dimensional scaling is a group of methods for representing high-dimensional data graphically in low (usually 2 or 3) dimensions. The objective in multi-dimensional scaling is to preserve the pair-wise similarities or distances between objects in the low-dimensional graphical representation. Multi-dimensional scaling analysis is similar to cluster analysis in that one is attempting to examine the relations among samples. But multi-dimensional scaling provides a graphical representation of the pair-wise similarities or distances among samples without forcing the samples into specific clusters.

BRB-ArrayTools provides multi-dimensional scaling analysis of the expression profiles of the samples. We provide a 3-dimensional representation displayed as a rotating cloud of spheres in which each sphere represents a single sample. Samples whose expression profiles are very similar are shown close together. The options for selecting gene sets, and similarity or distance metrics for the multi-dimensional scaling analysis are the same as for the clustering samples tool. In general, the user will use the same selections in both analyses. For more information on the options for the distance metric, please refer to the Distance metric section under the clustering samples tool. When complete data is used (i.e., data with no missing values), then the multi-dimensional scaling analysis using Euclidean distance is equivalent to a priancipal component analysis. When incomplete data is used (i.e., data which has missing values), then the multi-dimensional scaling analysis using Euclidean distance can be thought of as an approximation of a principal components analysis. When Euclidean distance is used on complete data, then BRB-ArrayTools utilizes the first three principal components as the axes for the multi-dimensional scaling representation. The principal components are orthogonal linear combinations of the genes. That is, they represent independent perpendicular dimensions that are rotations of the gene axes (if the thousands of gene axes could be represented). The first principal component is the linear combination of the genes with the largest variance over the samples of all such linear combinations. The second principal component is the linear combination of the genes that is orthogonal (perpendicular) to the first and has the largest variance over the samples of all such orthogonal linear combinations; etc. The first three principal components are usually good choices for multi-dimensional scaling representation, though they are not necessarily the best choice for observing clustered structure.

Options on the multi-dimensional scaling dialog box provide the user control of the labeling of samples in the graphical display.

The 3-D rotating plot contains controls for controlling the axis and speed of rotation. The user may also stop the rotation and use his/her mouse (holding the button down) to brush some of the points to obtain identification of the samples represented by those points.

The multi-dimensional scaling dialog box also provides an option for computing a global statistical significance test of whether the expression profiles are clustered. This global test of clustering is based upon the first three components obtained from a multi-dimensional scaling of the samples, which is equivalent to the first three principal components for complete data (i.e., data with no missing values) when the Euclidean distance metric is used. When computing the multi-dimensional scaling components for input into the global clustering procedure, the computation of the distance matrix is slightly different from the computation used in the rotating scatterplots. This is to ensure that the resulting multi-dimensional scaling coordinates are as analogous to principal components as possible, since the analogous relationship between multi-dimensional scaling coordinates and principal components exists only when the Euclidean distance is used. If the user chooses the centered correlation metric, then the samples are first centered by their means and standardized by their norms, and then the multi-dimensional scaling components are computed using a Euclidean distance on the resulting centered and scaled sample data. If the user chooses the uncentered correlation metric, then the samples are fist standardized by their norms, and then the multi-dimensional scaling components are computed using a Euclidean distance on the resulting scaled sample data. If the user chooses Euclidean distance as the distance metric, then there is no difference between the multi-dimensional scaling components used in the global test of clustering and that used in the rotating scatterplots.

The statistical significance test is based on a null hypothesis that the expression profiles come from the same multivariate Gaussian (normal) distribution. A multivariate Gaussian distribution is a unimodal distribution that represents a single cluster. The global test of clustering can be computed only when the user has at least 30 experiments in his or her dataset. More information about the global tests of clustering is available in "Methods of assessing reproducibility of clustering patterns observed in analyses of microarray data" by LM McShane, MD Radmacher, B Freidlin, R Yu, MC Li and R Simon, Journal of Computational Biology 9:505-511, 2002 and also available as a technical report at http://linus.nci.nih.gov/~brb/TechReport.htm

Using the classification tools

It is frequently of interest to determine whether samples of different phenotypes or samples collected under different conditions differ with regard to expression profile. For example, one class of samples may consist of breast tumors that contain BRCA1 mutations and the other class may consist of breast tumors that do not contain such mutations (I Hedenfalk, et al., Gene expression profiles of hereditary breast cancer, New England Journal of Medicine 344:549, 2001). Another example is comparing tumors that respond to therapy to those that do not. There are numerous microarray studies that have objectives of this type. This type of problem has been called "class prediction" in distinction to "class discovery" because the classes, or phenotypes, of the samples to be compared are known in advance of the expression profiling.

Cluster analysis is not usually the most effective tool for addressing class prediction problems. Clustering of samples is usually based on the entire set of genes represented on the experiment. Since the classes to be distinguished may differ only with regard to a relatively small subset of these genes, these differences may be dominated by variations among the thousands of other genes used in the clustering distance metric. Whereas perusal of the image plot associated with clustering the samples and the genes may reveal some genes that appear to distinguish the classes, the visual approach is error prone without statistical confirmation that the clusters observed do not represent patterns obtained by chance from screening thousands of genes for those that sort the samples.

BRB-ArrayTools contains two powerful tools for comparing expression profiles among pre-defined classes. They are found under the **Class comparison** and **Class prediction** menu items. Both tools presume that the data consists of experiments of different samples representative of the classes. The experiments should represent replication at the highest level, incorporating biological variability. That is, if we are comparing expression profiles of BRCA1 mutated breast tumors to non-BRCA1 mutated breast tumors, we need samples from breast tumors of different patients in both categories. It is not sufficient to compare replicate experiments of one RNA sample from a BRCA1 mutated tumor to one RNA sample from a non-BRCA1 mutated breast tumor. Comparing two RNA samples, regardless of how many experiments have been run with those two samples, cannot support conclusions about the influence of BRCA1 mutation on expression profiles. If cDNA arrays are used, the two class prediction analyses in BRB-ArrayTools assume that a common internal reference sample has been used in all experiments, or that a patient-specific normal tissue reference is used for the experiments.

Class comparison between groups of arrays

The Class Comparison Between Groups of Arrays Tool is used for comparing 2 or more pre-defined classes. The classes to be compared are defined by a column of the experiment design worksheet. The codes used in a column can be any set of numerical, character or character string codes. If an entry for a particular experiment is left blank in defining the column, that experiment will be omitted from the class comparison analysis. The ability of the user to define columns in the experiment design worksheet to specify class comparison analyses enables the user to import his/her complete set of experiments into BRB-ArrayTools once and to conduct a variety of different comparisons guided by different columns.

Two options presented for the Class Comparison Between Groups of Arrays Tool are important for some situations: the ability to pair samples, and the ability to average over replicate experiments. If two classes are compared and the experiments are paired, then the paired t-test option should be selected. For example, if experiments have been prepared for the primary tumor and metastatic tumors of each patient, then the paired t-test option is appropriate and may improve the statistical power of the analysis. If multiple technical replicates have been performed for some RNA samples, then the analysis must either be based on selection of a single replicate for each RNA sample or the averaging option should be used. The test is based on comparing the differences in mean log-ratios (or log-intensities) between classes relative to the variation expected in the mean differences. The variation is computed assuming that all the samples are independent. If there are multiple experiments for the sample RNA sample, then the within-class variation will under-estimate the inter-class variation in means to be expected. The user should either omit technically inferior experiments, arbitrarily choose between technically satisfactory experiments that are tightly correlated (as determined using the Scatterplot tool), or utilize the averaging option. A combination of these approaches may also be used. If the averaging option is selected, the user must specify a column of the experiment design worksheet that provides RNA sample identifiers so that the tool can identify when there are multiple experiments for the same RNA sample. When analyzing paired sample, samples with replicated pairing id and classification labels will automatically be averaged. For further information, refer to the following section entitled Specifying replicate experiments and paired samples.

The Class Comparison Between Groups of Arrays Tool computes a t-test or F-test separately for each gene using the normalized log-ratios for cDNA arrays and the normalized log-intensities for one color oligonucleotide arrays. The F-test is a generalization of the two-sample t-test for comparing values among groups. The user also has the option of using the random variance version of the t-test or F-test. This is generally advisable, particularly when the number of samples per class is small. The random variance tests are discussed in more detail below. They provide for sharing information among genes of the within-class variance in log-ratios or log signals. The class comparison tool computes the number of genes that are differentially expressed among the classes at the statistical significance level selected in the F-test menu and creates a gene list containing information about the significant genes.

Several other important statistics are also computed. The tool performs random permutations of the class labels (i.e., which experiments correspond to which classes). For each random permutation, all of the t-tests, F-tests or random variance t-tests and F-tests are re-computed for each gene. The Class Comparison Between Groups of Arrays Tool computes the proportion of the random permutations that gave as many genes significant at the level selected by the user as were found in comparing the true class labels. That p value provides a global test of whether the expression profiles in the classes are different. This test is also robust; although it uses the number of genes significant by the F-test as a statistic, it generates the permutation p-value of that statistic. In comparing classes, it is statistically easier to determine reliably whether expression profiles for pre-defined classes are different than to reliably determine exactly which genes are differentially expressed among the classes. The latter problem more directly confronts the multiple comparison problem.

For example, suppose that we select a significance level of 0.001 for including genes in the gene list. If there are 8000 genes on the experiment, then we expect by chance that 8 genes on the gene list will be false positives. If we obtain 24 genes on the gene list, then about one third of them are false positives. If we obtain 80 genes on the gene list, then about one tenth of them are false positives. Traditionally multiple comparison corrections used in statistical analyses are much more stringent, usually requiring that the chance of any false positives be very small. For most microarray analyses, such conservatism is not appropriate. However, gene lists will not be a useful basis for further experimentation if they are heavily populated by false positives. For example, if we select a significance level of 0.01 instead of 0.001, then we expect by chance that 80 genes on the gene list will be false positives. Having 80 false positives makes interpretation or planning confirmatory experiments very problematic. So using a relatively stringent 0.001 is usually appropriate. With that stringency, however, there may be a substantial chance of false negatives; that is, some genes that are differentially expressed among the classes will not be found significant at the 0.001 level. The false negative rate will depend on the sample size, the within-class variation among samples and the average fold difference between classes for the gene. Hence, although the statistical power for detecting individual genes is limited by the stringency needed to control the false positive rate, the global test of whether the classes differ with regard to expression profiles will have better power for establishing that the expression profiles are different. Because a single global test is performed, obtaining a p value of less than 0.05 is sufficient to establish that the expression profiles are different. Multiple comparison stringency needed for inferences about individual genes is not needed for the global test.

The Class Comparison Tool also provides univariate and multivariate permutation tests for the significance of individual genes. The univariate permutation tests are computed separately for each gene. The proportion of the permutations of the class label giving a t-test or F-test p value as small as obtained with the true class labels is the univariate permutation p value for that gene. Univariate permutation tests are not effective when the number of samples per class are small because there may not be enough distinct permutations to give a permutational p value below a stringent level such as 0.001. The multivariate permutation tests are much more effective in this situation. The multivariate permutation tests are described in more detail in a later section. They provide the ability to control the number of genes in the "discovery list" that are false discoveries (i.e. false positives) and the ability to control the proportion of false discoveries in the discovery list. For example, the user can specify that he/she wants 90% confidence that the discovery list contains no more than 10% false discoveries.

The multivariate permutation tests are similar in spirit to the Statistical Analysis of Microarrays (SAM) method, although they provide tighter probabilistic control on the number and proportion of false discoveries (Tucher et al. 2001 in reference list). SAM analysis is provided as a separate tool.

A new option in the Class Comparison Between Groups of Arrays Tool allows the user to control for a potentially confounding factor while comparing classes. For example, you could compare expression profiles of tissue from two types of tumors while controlling for the gender of the patients from which the tissue was taken. You could also control for technical factors such as print set of the arrays. The analysis performed is an analysis of variance for a randomized block design. Two linear models are fit to the expression data for each gene. The full model includes class variable and the block variable, and the reduced model includes only the block variable. Likelihood ratio test statistics are used to investigate the significance of the difference between the classes. The random variance option is also available for this analysis of variance and multivariate permutation tests are applied to the resulting significance levels.

Class comparison between red and green channels

The Class Comparison Between Red and Green Channels Tool is used for comparing 2 pre-defined classes in a dual-color non-reference design experiment. Four columns of the experiment design worksheet define the experiments. The red-color sample ID and the green-color sample ID define specimen labels for the samples labeled with red and green dye respectively. The red-color sample class and green-color sample class columns indicate the class labels of the two specimens on each array. The codes used in a column can be any set of numerical, character or character string codes. If an entry for a particular experiment is left blank in defining the column, that experiment will be omitted from the class comparison analysis. Each array should include samples of two phenotype classes. Each sample should be paired with one and only one samples of the opposite class. Technical replicates (arrays that contain identical pairs of samples associated with the same channels) and die-swap experiments (arrays that contain identical pairs of samples associated with the interchanged channels) are averaged. Resulting log-ratios are analyzed by the one-sample t-test with the null hypotheses that the mean of the log-ratio distribution is equal to zero.

As a special case, this tool can also analyze reference design with one class compared with the common reference. In this case, reference samples should contain the key word "Reference" in the sample ID column.

All the other options except the ability to pair samples, and the ability to average over replicate experiments other than technical replicates are identical to the Class Comparison Between Groups of Arrays Tool. Output of the tool is also identical to the Class Comparison Between Groups of Arrays Tool output.

Class prediction

The Class Prediction Tool is similar to the Class Comparison Between Groups of Arrays Tool in that the classes must be pre-defined. The user specifies the classes by identifying a column of the experiment description worksheet. Blanks labels in the column indicate that the corresponding experiment should be omitted from the analysis. Several of the class prediction methods are only applicable when there are two classes. Replicate experiments of the same RNA sample may be averaged, by entering a sample id column in the experiment design worksheet. When analyzing paired samples, samples with replicated pairing id and classification labels will automatically be averaged. For further information, refer to the following section entitled Specifying replicate experiments and paired samples.

For dual-label experiments, the class prediction methods are based on normalized log-ratios relative to a common reference RNA sample. For Affymetrix GeneChip^{TM} arrays the methods are applied to normalized log signal values. To reduce the dominating effect that genes with overall high intensity might have with some class prediction methods, the single-channel log-signal values are median-centered on a gene-by-gene basis for class prediction. This is not done for class comparison analyses because the results there are un-affected by median centering the genes.

The user specifies a significance level to be used for determining the genes that will be included in the predictors; genes that are differentially expressed between the classes at a univariate parametric significance level less than the specified threshold are included in the predictor. It doesn't matter whether the specified significance level is small enough to exclude enough false discoveries, because class prediction is not really about discovering differentially expressed genes. In some problems better prediction can be achieved by being more liberal about the gene sets used as features. The predictors may be more biologically interpretable and clinically applicable, however, if fewer genes are included.

The Class Prediction Tool creates a multivariate predictor for determining to which of the two classes a given sample belongs. Several multivariate classification methods are available, including the Compound Covariate Predictor, Diagonal Linear Discriminant

Analysis, Nearest Neighbor Predictor, Nearest Centroid Predictor, and Support Vector Machine Predictor.

Compound covariate predictor

The Compound Covariate Predictor is a weighted linear combination of log-ratios (or log intensities for single-channel experiments) for genes that are univariately significant at the specified level. By specifying a more stringent significance level, fewer genes are included in the multivariate predictor. Genes in which larger values of the log-ratio pre-dispose to class 2 rather than class 1 have weights of one sign, whereas genes in which larger values of the log-ratios pre-dispose to class 1 rather than class 2 have weights of the opposite sign. The univariate t-statistics for comparing the classes are used as the weights. Detailed information about the Compound Covariate Predictor is available in the Hedenfalk reference given above or in “A paradigm for class prediction using gene expression profiles” by MD Radmacher, LM McShane and R Simon, A paradigm for class prediction using gene expression profiles. Journal of Computational Biology 9:505-511, 2002, also available in Technical Report 01, 2001, Biometric Research Branch, National Cancer Institute, http://linus.nci.nih.gov/~brb/TechReport.htm).

Diagonal linear discriminant analysis

The Diagonal Linear Discriminant Analysis is similar to the Compound Covariate Predictor, but not identical. It is a version of linear discriminant analysis that ignores correlations among the genes in order to avoid over-fitting the data. Many complex methods have too many parameters for the amount of data available. Consequently they appear to fit the training data used to estimate the parameters of the model, but they have poor prediction performance for independent data. The study by Dudoit et al. (S Dudoit, J Fridlyand, TP Speed; Comparison of discrimination methods for the classification of tumors using gene expression data, Journal of the American Statistical Association 97:77-87, 2002) found that diagonal linear discriminant analysis performed as well as much more complicated methods on a range of microarray data seta.

Nearest neighbor predictor

The Nearest Neighbor Predictor is based on determining which expression profile in the training set is most similar to the expression profile of the specimen whose class is to be predicted. The expression profile is a vector of log-ratios or log-intensities for the genes selected for inclusion in the multivariate predictor. Euclidean distance is used as the distance metric for the Nearest Neighbor Predictor. Once the nearest neighbor in the training set of the test specimen is determined, the class of that nearest neighbor is taken as the prediction of the class of the test specimen. k-Nearest Neighbor Prediction is similar. For example with the 3-Nearest Neighbor algorithm, the expression profile of the test specimen is compared to the expression profiles of all of the specimens in the training set and the 3 specimens in the training set most similar to the expression profile of the test specimen are determined. The distance metric is again Euclidean distance with regard to the genes that are univariately significantly differentially expressed between the two classes at the threshold significance level specified. Once the 3 nearest specimens are identified, their classes vote and the majority class among the 3 is the class predicted for the test specimen.

Nearest centroid predictor

The Class Prediction Tool also offers Nearest Centroid Prediction. In the training set there are samples belonging to class 1 and to class 2. The centroid of each class is determined. The centroid of class 1, for example, is a vector containing the means of the log-ratios (or log intensities for single label data) of the training samples in class 1. There is a component of the centroid vector for each gene represented in the multivariate predictor; that is, for each gene that is univariately significantly differentially expressed between the two classes at the threshold significance level specified. The distance of the expression profile for the test sample to each of the two centroids is measured and the test sample is predicted to belong to the class corresponding to the nearest centroid.

Support vector machine predictor

The Class Prediction Tool also offers Support Vector Machine Prediction. A support vector machine (SVM) is a class prediction algorithm that has appeared effective in other contexts and is currently of great interest to the machine learning community. SVMs were developed by V. Vapnik (The Nature of Statistical Learning. Springer-Verlag, New York, 1995). We have implemented SVMs with linear kernel functions as our experience has been that more complex SVMs perform less well for this application. The SVM predictor is a linear function of the log-ratios or the log-intensities that best separates the data subject to penalty costs on the number of specimens misclassified. In the options dialog, the user can alter the default penalty cost or the cost of misclassifying a specimen of class 1 relative to misclassifying a specimen of class 2. We use the LIBSVM implementation of Chang and Lin.

Cross-validation and permutation p-value

For all of the class prediction methods requested, the Class Prediction Tool provides an estimate of how accurately the classes can be predicted by this multivariate class predictor. The cross-validated misclassification rate is computed and the results are reported. The cross-validation process omits one sample at a time. For each sample omitted, the entire analysis is repeated from scratch, including the determination of which genes are univariately significant on the reduced training sample. From that gene list, a multivariate predictor is constructed and applied to the sample that was omitted. The program records whether that prediction was correct or not. This is repeated, omitting all of the samples one at a time. The output table shows which samples were correctly and incorrectly predicted, and the overall cross-validated misclassification rate. The output table is arranged so that you can see the extent to which the different predictors agree or disagree for each specimen left out of the training set. Because of the large number of candidate predictor variables, it is essential to use cross-validation or some similar method to determine whether the model predicts accurately. Even with classes that do not differ in expression profiles, it is very easy to develop models that predict perfectly when measured in a non cross-validated manner. Such models would be useless for application with independent data. For further discussion see Simon R, Radmacher MD, Dobbin K, and McShane LM, Pitfalls in the analysis of DNA microarray data: Class prediction methods, Journal of the National Cancer Institute 95:14-18, 2003.

In addition to providing the cross-validated misclassification information, the Class Prediction Tool also provides the permutation p value for the cross-validated misclassification error rate. That is, for each random permutation of class labels, the entire cross-validation procedure is repeated to determine the cross-validated misclassification rate obtained from developing a multivariate predictor with two random classes. The final p value is the proportion of the random permutations that gave as small a cross-validated misclassification rate as was obtained with the real class labels. There is a cross-validated misclassification rate and a corresponding p value for each class prediction method requested. The user inputs the number of permutations desired. At least 1000 permutations should be specified for a valid permutation p value on the cross-validated misclassification rate. The user may specify only 1 permutation and will quickly obtain the appropriate gene list and valid cross-validated misclassification rates. Only the p values associated with the cross-validated misclassification rates will be missing.

Note that the list of genes that are used to form the class predictors will coincide with the list of genes produced by the Class Comparison Between Groups of Arrays Tool for the same analysis between the two classes. The only exception is when there is only one nonmissing value in one of the two classes, or when the variance in one of the two classes is zero. In that case, the compound covariate predictor imputes the p-value for that gene to be 1, rendering that gene nonsignificant for that permutation or analysis, whereas the Class Comparison Between Groups of Arrays Tool does not.

The Class Prediction Tool saves this gene list in the `..\ArrayTools\Genelists\User` folder using the genelist name specified by the user.

Prediction for new samples

The Class Prediction Tool can be used to classify samples not used in the model building process. In the class prediction dialog, check the box labeled “Use separate test set” . You should create a column in your experiment descriptor worksheet that indicates which experiments contain samples that are to be included in the model building process (labeled “training”), which are to be used only for classification prediction once the model building is completely finished (labeled “predict”), and which samples are to be excluded entirely from both the training and test sets (labeled “exclude”). Predictions will be made for the samples labeled “predict” using all of the class prediction methods requested in the dialog.

Ususally, you do not have a separate set of samples that you wish to withhold from the model building process when the analysis is first done. The model building process itself uses leave-one-out cross validation, and so it is not necessary to have a separate test set. Sometimes, however, after the model is built, you have additional samples whose classes you wish to predict using the previously developed model(s). We have enabled you to do this by re-building the model using the initial set of samples, and to then predict for the new samples in a combined analysis. This has been more direct for us to implement, rather than trying to save the originally developed model, since in cases like k-nearest neighbor classifiers, the entire dataset is needed for classification of new samples.

Binary tree prediction

The Binary Tree Prediction Tool is another algorithm that can be used for class prediction when there are more than 2 classes. The same methods (compound covariate predictor, diagonal linear discriminant analysis, k-nearest neighbor using k=1 or 3, nearest centroid, and support vector machines) form the foundation of both tools. Moreover, for the case of only two distinct classes, binary tree prediction is identical to the Class Prediction Tool (except it evaluates only one prediction algorithm at a time whereas Class Prediction Tool can evaluates all of them in one run). The difference between the algorithms lies in how they treat cases with 3 or more classes.

The Binary Tree Prediction Tool does not attempt to predict the class of a sample in one step. Instead, at each node of a binary tree the samples are classified into two subsets of classes. One or both of the subsets may consist of multiple classes. The user-specified prediction method is applied to construct a predictor that can distinguish the two subsets of classes. The decision on how to split the classes in each node into two subset classes is determined by the split that can be accomplished with the fewest cross-validated misclassification errors. All possible divisions of the classes into two groups are tested, and the best one (with the lowest misclassification rate) is accepted as a node of the binary tree. If the misclassification rate associated with the node is above the specified threshold, the split is not accepted. In this case, the classifier will not attempt to distinguish between the specific classes in the group.

Then, each of the resulting two groups is investigated in a similar manner. The process stops when each group contains only one sample or none of the divisions of the group into the subgroups has the cross-validation misclassification error rate below the threshold.

If cross-validation of the binary tree-prediction algorithm is requested by the user, then the entire process of binary tree building is repeated for each training set, and the overall cross-validation misclassification rate is thereby determined.

Two methods of cross-validation are implemented. For leave-one-out cross-validation method, samples are excluded from the analysis one by one, the remaining samples used to create a classifier that is used to predict a class of the one sample that was set aside.

For large data sets, the leave-one-out validation may take very a long time. The tool has an option of using K-fold validation instead of the leave-one-out validation. For K-fold cross-validation method, samples are divided into K approximately equal groups. Then, groups are excluded from the analysis one by one, the samples of the remaining K-1 groups are used to create a classifier that is used to predict a class of the samples that were set aside. The user can assign K. By default, this tool uses leave-one out validation for the data sets with 25 or fewer arrays, and 10-fold validation if the number of arrays exceeds 25.

Some of the algorithms in the standard Class Prediction Tool can be used even when there are more than 2 classes. The sets of genes used in the predictor at each node of the binary tree may differ, however, thus enhancing the ability of the algorithm to discriminate between the classes. More research is needed to investigate relative strength and weaknesses of the binary tree prediction algorithm when compared to the one-step prediction algorithms as applied to the microarray data.

Survival analysis

The Survival Analysis Tool finds genes that are predictive of survival time for patients. Since some patients/animals may still be alive at the time of analysis, their survival times from entry on study is *censored*; that is, it is at least as long as survival measured to date, but longer by an unknown amount. There are many statistical methods for analysis of censored survival data. The most popular method is Cox's proportional hazards model, and the Efron method of handling ties is implemented here. This is a regression model in which the hazard function for an individual is a function of predictor variables. In our case the predictor variables are log expression levels. The hazard function is the instantaneous force of mortality at any time conditional on having survived till that time. The proportional hazards model postulates that the logarithm of the hazard of death is a linear function of the predictor variables, linked by unknown regression coefficients. For more details see biostatistics texts or the original paper (DR Cox, Regression models and life tables, J.Royal Stat Soc B 34:187-202).

The Survival Analysis Tool tool fits proportional hazards models relating survival to each gene, one gene at a time and computes the p value for each gene for testing the hypothesis that survival time is independent of the expression level for that gene. Gene lists are created based on these p values in the same way as in the Class Comparison tools. The p values can be used to identify gene lists using multivariate permutation tests for controlling the number or proportion of false discoveries. Or the gene list can simply consist of the genes with p values less than a specified threshold (0.001 is default). For more information regarding the multivariate permutation tests for controlling the number or proportion of false discoveries, please see the preceding section on Multivariate Permutation Tests for Controlling Number and Proportion of False Discoveries.

Quantitative traits analysis

This tool finds genes that are significantly correlated with a specified quantitative variable (trait) such as age. Spearman or Pearson correlation coefficients are used as a measure of correlation and to compute parametric p-values. Most of the options are identical to the Class Comparison Tools. Two exceptions are the pairing of the samples and the random variance model that are not implemented for the Quantitative Traits Analysis Tool. Output of the tool is also nearly identical to that for the Class Comparison Tools.

Gene Ontology comparison

The **Gene Ontology Comparison Tool** provides a list of GO categories that have more genes differentially expressed among the classes than expected by chance. P-values of two permutation tests, based on the LS and KS statistics respectively, are used to select these GO categories. For a set of N genes, the LS statistic

$$LS = \sum_{i=1}^{N}(-\log(p_i)\ )/N$$

is defined as the mean negative natural logarithm of the p-values of the appropriate single gene univariate test. The KS statistic

$$KS = \max_{i=1}^{N} (i/N-p_i), \qquad p_1 \leq p_2 \leq \ldots \leq p_N$$

is defined as the maximum difference between i/N and p_i, where p_i is the i^{th} smallest p-value of the univariate test.

The statistical significance values are based on testing the null hypothesis that the list of genes that belong to a given GO category is a random selection from the project gene list, against the alternative hypothesis that it contains more genes differentially expressed between the classes being compared. The permutation p-value is not based on normal distribution assumptions. Rather, it is based on a permutation procedure. For a given GO category, the permutation procedure randomly selects N genes from the list of genes that are selected for the analysis. Here N is the number of genes in the project gene list that belong to the GO category. For each selection, associated LS and KS statistics are computed. The selection process is repeated 100,000 times to obtain a distribution of these statistics. Then, for each GO category, the LS (KS) permutation p-value is defined as the proportion of selections for which the LS (KS) statistic is larger then the LS (KS) statistics computed for the GO category with original gene list.

The tests are applied separately to each GO category. A GO category is selected if its corresponding LS or KS permutation p-value is below the threshold specified by the user (default is 0.005). Some approximations are used to speed up computations.

Results are provided as a table of selected GO categories that are ordered by the p-value of the LS test (smallest first). For each GO category, the table lists the unique identifier, the number of genes in the project gene list that belong to the GO category, and the LS and KS p-values. For each class, the geometric mean of gene expression (ratios) is provided. If only two classes are considered, the fold difference of these geometric means is also listed.

Another table lists all genes that are found in the selected GO categories. They are ordered by the p-value associated with the GO category. If a gene belongs to several GO categories, it will be listed several times, once for each GO category that contains the gene.

Gene List comparison

The **Gene List Comparison Tool** is very similar to the Gene Ontology Comparison Tool. The only difference is that user-defined gene lists are investigated rather than GO categories. The tool investigates the user-defined gene lists and selects those that have more genes differentially expressed among the classes than expected by chance. P-values of two permutation tests based on the LS and KS statistics, respectively, are used to select these gene lists (see description of the Gene Ontology Comparison Tool for a definition of the LS and KS statistics). The statistical significance values are based on testing the null hypothesis that the set of genes that belong to a given gene list is a random selection from all the genes in the project after filtering. The alternative hypothesis is that the selected gene lists contains more genes differentially expressed between the classes being compared than expected by chance.

Results are provided as a table of selected gene lists ordered by the p-value of the LS test (smallest first). For each selected gene list, the table lists the unique identifier, the number of genes in the project gene list that belong to the gene list, and the LS and KS p-values. For each class, the geometric mean of gene expression (ratios) is provided. If only two classes are considered, the fold difference of these geometric means is also listed.

Another table lists all genes that are found in the selected gene lists. They are ordered by the p-value associated with the gene list. If a gene belongs to several gene lists, it will be listed several times, once for each gene list that contains the gene.

Some options available in classification, survival, and quantitative traits tools

Random Variance Model

The random variance model is found as an option in the class comparison and class prediction tools. The *random variance* t and F tests can be selected as alternatives to the standard t/F tests. The standard t/F tests are based on the assumption that the within class variance is different for different genes and hence these variances are estimated separately for each gene. When there are few samples in each class, the variance estimates are not very accurate and the statistical power of t/F tests are poor. Some published methods assume that the variances are the same for all genes, and that is clearly a poor assumption. The random variance model takes an intermediate approach. It assumes that different genes have different variances, but that these variances can be regarded statistically as independent samples from the same distribution. Specifically, the reciprocal of the variances are regarded as draws from a gamma distribution with two parameters (a,b). The parameters are estimated from the complete set of expression data and we first test that the model is accurate. In most datasets we have examined, the model assumption is very accurate. The usual t test for comparing expression of gene i in two classes is based on the statistic

$$t = \frac{\overline{x}_{i,1} - \overline{x}_{i,2}}{\hat{\sigma}_i \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

where the numerator is the difference in the means of log expression for gene i in the two classes, n_1 and n_2 are the number of samples in the two classes, and $\hat{\sigma}_i$ denotes the square root of the usual within-class variance for gene i. The within-class variance is computed by a weighted average of the variance of log expression in each of the two classes. With the usual t test, under the null hypothesis, the t value has a t distribution with n_1+n_2-2 degrees of freedom. For the random variance model, the same formula for t is used, except that $\hat{\sigma}_i$ is replaced in the denominator by $\tilde{\sigma}_i$ where

$$\tilde{\sigma}_i^2 = \frac{\left(n_1 + n_2 - 2\right)\hat{\sigma}_i^2 + 2a\left(1/ab\right)}{\left(n_1 + n_2 - 2\right) + 2a}$$

The quantity 1/ab is an estimate of the expected variance for the inverse gamma model. Consequently, $\tilde{\sigma}_i^2$ is a weighted average of the usual gene specific variance and the average variance for all the genes. The weight given to the gene specific variance is the total number of samples in the two classes minus two. The weight given to the average variance for all the genes is 2a. It can be shown rigorously that when the model assumption holds, that the t values computed using $\tilde{\sigma}_i$ have a t distribution under the null hypothesis but the degrees of freedom is now n-2+2a. In cases where the number of samples n is small, this increase in degrees of freedom and the associated improvement in the estimation of variances results in improved statistical power for detecting differentially expressed genes. These results extend directly to the F distribution for comparing more than two classes. For details, see G Wright and R Simon, The random variance model for finding differentially expressed genes, Bioinformatics (19:2448-2455, 2003), also Technical Report, Biometric Research Branch, National Cancer Institute, http://linus.nci.nih.gov/~brb ; P Baldi and AD Long, A Bayesian framework for the analysis of microarray expression data: regularized t-test and statistical inferences of gene changes, Bioinformatics 17:509-519, 2001.

Multivariate Permutation Tests for Controlling Number and Proportion of False Discoveries

The multivariate permutation tests for controlling number and proportion of false discoveries is an option found in the class comparison, survival analysis, and quantitative traits analysis tools. Using a stringent p<0.001 threshold for identifying differentially expressed genes is a valid way for controlling the number of *false discoveries*. A false discovery is a gene that is declared differentially expressed among the classes, when in fact it is not. There are two problems with this approach to controlling the number of false discoveries. One is that it is based on p values computed from the parametric t/F tests or random variance t/F tests. These parametric p values may not be accurate in the extreme tails of the normal distribution for small numbers of samples. The second problem is that this approach does not take into account the correlation among the genes. Using stringent p value thresholds on the univariate permutation p values won't be effective when there are few samples and will not account for correlations. We have implemented multivariate permutation tests that accomplish both objectives. They are described in detail in R Simon, EL Korn, LM McShane, MD Radmacher, GW Wright and Y Zhao, *Design and Analysis of DNA Microarray Investigations*, Springer 2003, in EL Korn, JF Troendle, LM McShane and R Simon, Controlling the number of false discoveries: Application to high dimensional genomic data; Journal of Statistical Planning and Inference (In Press), also Technical Report 3, Biometric Research Branch, National Cancer Institute, 2002; http://linus.nci.nih.gov/~brb. This approach is also described by Reiner A, Yekutieli D and Benjamini Y, Identifying differentially expressed genes using false discovery rate controlling procedures, Bioinformatics 19:368-375, 2003).

The multivariate permutation tests are based on permutations of the labels of which experiments are in which classes. If there are fewer than 1000 possible permutations, then all permutations are considered. Otherwise, a large number of random permutations are considered. For each permutation, the parametric tests are re-computed to determine a p value for each gene that is a measure of the extent it appears differentially expressed between the random classes determined by the random permutation. The genes are ordered by their p values computed for the permutation (genes with smallest p values at the top of the list). For each potential p value threshold, the program records the number of genes in the list. This process is repeated for a large number of permutations. Consequently, for any p value threshold, we can compute the distribution of the number of genes that would have p values smaller than that threshold for permutations. That is the distribution of the number of false discoveries, since genes that are significant for random permutations are false discoveries. The algorithm selects a threshold p value so that the number of false discoveries is no greater than that specified by the user C% of the time, where C denotes the desired confidence level. If C=50%, then the median number of false discoveries is that specified by the user. If, for example, C=90%, then you have 90% confidence that the number of false discoveries is no greater than that number specified by you. In a similar manner, we determine threshold p values so that the resulting gene list contains no more than a specified proportion of false discoveries.

The class prediction tool produces a gene list ordered with the genes having the smallest parametric p values at the top. The length of the gene list is determined by the types of false discovery control selected. Generally, we recommend using all of the options:

univariate p value threshold (0.001); limiting number of false discoveries (10 default), and limiting proportion of false discoveries (0.10 default). The output tells you where to truncate the gene list in order to get each type of control.

The procedures for controlling the number or proportion of false discoveries are based on multivariate permutation tests. Although parametric p values are used in the procedures, s the permutation distribution of these p values is determined, and hence the false discovery control is non-parametric and does not depend on normal distribution assumptions.

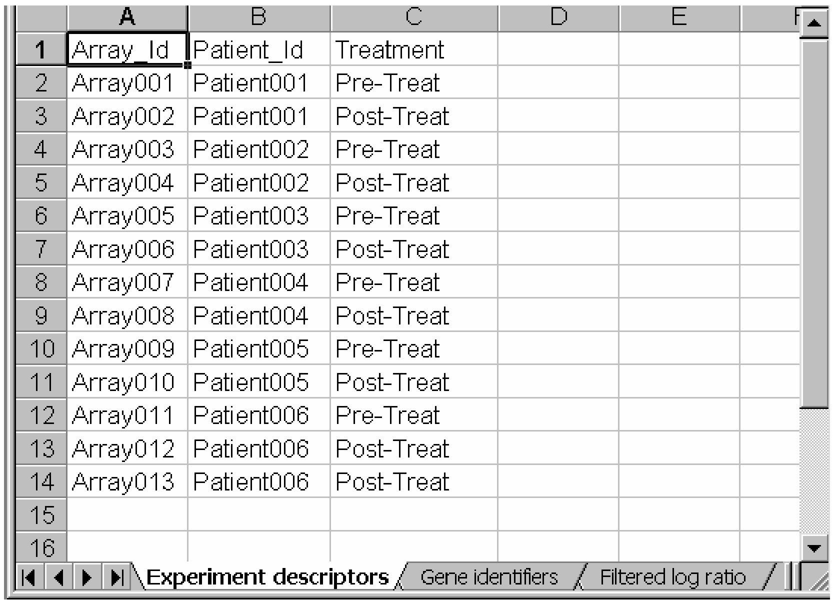
The multivariate permutation tests use the expression data much more efficiently than the univariate permutation tests. Consequently, we no longer recommend using the univariate permutation tests. We have retained the option to do the univariate permutation tests, but we have made the default to skip them. The multivariate permutation tests are particularly more effective when the number of samples is limited. In that case it may be impossible to obtain differences significant at the 0.001 level by a univariate permutation test, but quite possible to find significant genes while ensure that the number or proportion of false discoveries is controlled using the multivariate permutation tests.

The multivariate permutation tests take advantage of the correlation among the genes. For a given p value for truncating an ordered gene list; the *expected number* of false discoveries does not depend on the correlations among the genes, but distribution of the number of false discoveries does. The distribution of number of false discoveries is skewed for highly correlated data. If you specify the confidence coefficient at 50%, the program provides the length of the gene list associated with a specified median number of false discoveries or given proportion of false discoveries. You may think of those gene lists as being expected to contain the target number or proportion of false discoveries, although the median and expected number are not exactly the same. Requiring 90% confidence that the gene list does not contain more false discoveries than the target amount is a more stringent condition and the associated gene lists will be much shorter than those based on the median. When there are few samples, the median number of false discoveries may be highly variable, and we recommend using the 90% confidence level.

The HTML output file created by the Class Comparison Tools gives the parametric p-value, the univariate permutation p-value (if requested), and the geometric mean ratio (geometric mean intensity for single color oligonucleotide arrays) in each class, and gene identifying information. The list is sorted by parametric significance level. If there are two classes, the list is in two parts; those genes down-regulated in class one relative to class two followed by those genes up-regulated in class one relative to class two. When there are more than two classes, the patterns of differential expression among the classes will be more varied for genes in the list. The Class Comparison Tools save this gene list in the `..\ArrayTools\Genelists\User` folder using the genelist name specified by the user. It is often useful to cluster the samples with regard to this gene list in order to better understand the pattern of genes across samples.

Specifying replicate experiments and paired samples

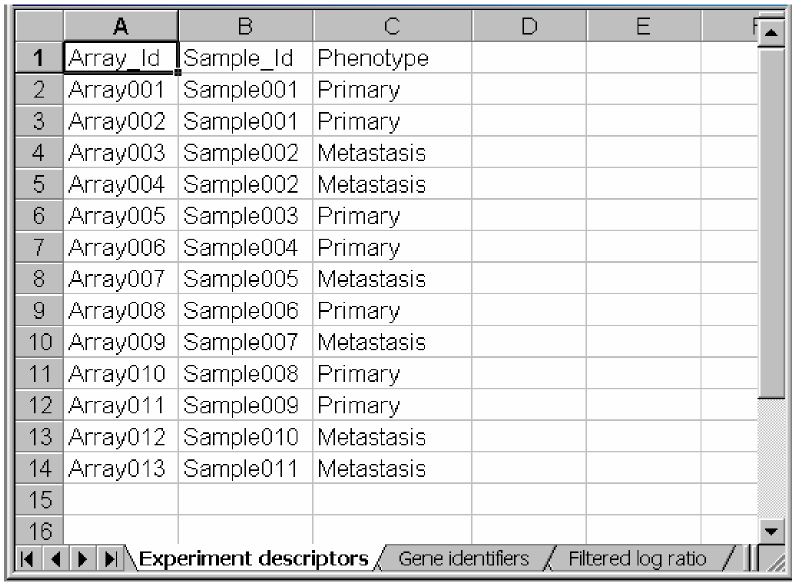
For the class comparison and class prediction analysis tools, the user may need to specify replicate experiments or paired samples, and for the survival analysis and quantitative traits analysis tools, the user may need to specify replicate experiments on the same sample. To enter paired samples into an analysis, the user must create a descriptor variable on the Experiment descriptors sheet in which a unique value is given for each pair. The following example shows the experiment design sheet for a paired analysis:



	A	B	C	D	E	F
1	Array_Id	Patient_Id	Treatment			
2	Array001	Patient001	Pre-Treat			
3	Array002	Patient001	Post-Treat			
4	Array003	Patient002	Pre-Treat			
5	Array004	Patient002	Post-Treat			
6	Array005	Patient003	Pre-Treat			
7	Array006	Patient003	Post-Treat			
8	Array007	Patient004	Pre-Treat			
9	Array008	Patient004	Post-Treat			
10	Array009	Patient005	Pre-Treat			
11	Array010	Patient005	Post-Treat			
12	Array011	Patient006	Pre-Treat			
13	Array012	Patient006	Post-Treat			
14	Array013	Patient006	Post-Treat			
15						
16						

Here, the pre-treatment samples are to be compared against the post-treatment samples for each patient. The pairing variable is `Patient_Id` and the class variable to be compared is `Treatment`. Note that both `Array012` and `Array013` contain post-treatment samples for `Patient006`. In that case, the average of the two experiments will be used for the analysis.

When an unpaired analysis is used, the user should make sure to specify whenever there are replicate experiments that have been performed using the same sample. The following example shows an experiment descriptor sheet in which replicate experiments were performed for some of the samples:



	A	B	C	D	E
1	Array_Id	Sample_Id	Phenotype		
2	Array001	Sample001	Primary		
3	Array002	Sample001	Primary		
4	Array003	Sample002	Metastasis		
5	Array004	Sample002	Metastasis		
6	Array005	Sample003	Primary		
7	Array006	Sample004	Primary		
8	Array007	Sample005	Metastasis		
9	Array008	Sample006	Primary		
10	Array009	Sample007	Metastasis		
11	Array010	Sample008	Primary		
12	Array011	Sample009	Primary		
13	Array012	Sample010	Metastasis		
14	Array013	Sample011	Metastasis		
15					
16					

In this case, `Sample_Id` is the replicate variable to average over. Since `Array001` and `Array002` have the same `Sample_Id`, their average will be used for the analysis. Similarly, `Array003` and `Array004` will be averaged for the analysis. All the other experiments will be used as-is, since they have no replicates.

Gene Ontology observed v. expected analysis

The Gene Ontology Observed vs. Expected analysis is an option found in the class comparison, class prediction, survival analysis, and quantitative traits analysis tools.

The purpose of the Gene Ontology Observed vs. Expected analysis is to give information whether or not the list of significant genes selected by the analysis is different from a genelist randomly selected from all genes in the analysis, for a given Gene Ontology. This is expressed by the Observed vs. Expected ratio. Before we define the Observed and Expected ratio, the term "genes in the analysis" should be clarified. The "genes in the analysis" refers to the set of genes which were input into the class comparison, class prediction, survival analysis, and quantitative traits analysis tool, i.e. the genes passing the filtering criteria.

The **Observed** is defined as the number of genes in the list of significant genes selected by the analysis tool which fall into a Gene Ontology category. The **Expected** is defined as the average number of genes which would be expected to fall into that Gene Ontology category in a subset of genes randomly selected from the genes in the analysis. A Gene Ontology category is considered to consist of not only the genes which are described by that Gene Ontology term, but also any gene which is described by any children of that Gene Ontology term.

For example, suppose we are computing the observed and expected counts of the Cellular Component term “Cytoplasm”. We choose a set of genes, denoted as A, which has total N number of genes with a Cellular Component term, in the analysis of Class Comparison. From these N genes, we get n significant genes by certain criteria, which forms a subset S. In the set A, there are M genes belonging to the Cellular Component term “Cytoplasm” or any of the term’s children. And in the subset S, there are m genes belonging to the Cellular Component term “Cytoplasm” or any of the term’s children. Then, the Observed is m and the Expected is $n \times M/N$. The ratio of Observed vs. Expected for “Cytoplasm” is $m/(n \times M/N)$, simplified by mN/Mn.

The user may specify a threshold for the minimum number of genes which must be in a Gene Ontology category to be considered (default=5), as well as a threshold for the minimum Observed vs. Expected ratio to be considered (default=2) in the **Options** page of the analysis dialogs.

Programmable Plug-In Faciltiy

BRB-ArrayTools has a plug-in facility that enables users or methodology developers to extend the tool set built in to BRB-ArrayTools by providing their own tools. Users can then easily add-in the new tools to their version of BRB-ArrayTools and use them in the analysis of their data. The extensions must be written as functions in the R statistical programming language. R is an open source and very powerful statistical language that is popular with academic statistical scientists. It is in many ways a public domain version of S+, although each language has some features missing in the other. There is a growing body of R functions developed for the analysis of microarray data and the plug-in facility can give BRB-ArrayTools users and R software developers access to each other via BRB-ArrayTools.

Detailed information about the creation and use of add-in R functions is described in a separate document entitled BRB-ArrayTools Plug In Guide. Here we will give an overview of the plug-in facility. In considering R plug-ins there are two perspectives, one for the R tool developer and one for the BRB-ArrayTools user analyzing his/her data and wanting to utilize plug-ins.

For the plug-in user, the process of installing and accessing a plug-in is quite simple. The BRB-ArrayTools main menu contains a “Plug In” entry and a “Load Plug In” sub-menu entry. The Load Plug In dialog box askes for the path name of the plug in and it asks whether the user wants the plug-in added to the plug-in menu for future use. When the user clicks the “Load” button on the Load Plug In dialog box, the plug-in is launched and it will prompt the user for any further information it needs. BRB-ArrayTools will pass the R plug-in any data that it needs in the user’s Project Workbook that has already been collated by the user. The output of the analysis performed by the plug-in will be directed to a file and that file will be opened either by the plug-in itself or by the user. The user only needs to know the name of the plug-in. In using plug ins obtained from others, the user has to install the plug in in the PlugIns folder of the BRB-ArrayTools directory. There is a .r sub-folder for r functions, a .plug sub-folder for plug-in interfaces created by developers, and a .txt sub-folder for readme files created by developers.

The creation of R plug-ins for BRB-ArrayTools is also relatively easy. The first step is to write the R function that performs the desired analysis. This function will be launched by the BRB-ArrayTools user as described above, and BRB-ArrayTools will pass to the R function the information in the active collated workbook that the function needs. The data will be passed via the COM (Common Object Module) architecture and the developer need not read any files to obtain the desired data. In writting the R function, the developer can name the available data objects provided by BRB-ArrayTools any way he/she chooses. Because the R function runs in the background as a COM object, output from the function should be written to a file. The R function may open the file before terminating or the user can open the file.

The second step in creating a plug-in is to create the interface between the R function and BRB-ArrayTools. We provide a wizard to make this step easy. The wizard enables the developer to tell BRB-ArrayTools what data objects the R function needs and what names the R function wants associated with those data objects. The wizard also enables the developer to prompt the user for additional data input. The input dialog will be in the format of the Micrososft Windows type of dialog boxes used by BRB-ArrayTools but the VBA coding of these dialogs will be automatically generated by BRB-ArrayTools using information provided by the developer in his/her dialog with the wizard. For more information, see the BRB-ArrayTools Plug In Guide.

Further help

Some useful tips

Excluding experiments from an analysis

Some of the analyses within BRB-ArrayTools can be run on a subset of the experiments in the collated project workbook. For the hierarchical clustering and multidimensional scaling tools, an experiment will be excluded from the analysis if it has a blank label within the designated experiment descriptor variable. For the scatterplot of phenotype averages and the classification tools, an experiment will be excluded from the analysis if it has a blank label within the selected phenotype class variable.

Extracting genelists from HTML output

Sometimes the user may need to extract columns of data from the HTML tables that are produced by BRB-ArrayTools. For example, the user might wish to include the HTML genelist table produced by the classification tools in a Word document for publication. Or the user might need to extract the t-values and midpoint values from the compound covariate predictor output for use in classifying new samples. Or the user might wish to sort the genelist table by a particular column.

HTML tables can be easily converted to tables in Word or columns in Excel. You can do this in one of two ways. One way is to select the HTML table by left-clicking and dragging the mouse over the entire table, and then copying to the buffer through **Edit → Copy** (or **Ctrl-C**). Then open up a blank Word or Excel document, and paste the contents of the buffer into the document through **Edit → Paste** (or **Ctrl-V**).

Another way is to open the HTML file within Word or Excel using the **File → Open** menu. In the **Open** dialog box, go to the **Files of type** field and select **HTML Document** (in Word or Excel 97) or select either **All Microsoft Excel Files** or **Web Pages** (in Excel 2000). Now browse for the HTML document which you would like to convert. Once the table has been converted to a Word table or to cells within Excel, it is very easy to add to, delete, or edit any of the rows or columns in the table. Note that all hyperlinks are preserved when converting HTML tables to Word or Excel.

Creating user-defined genelists

The best utility to use for creating user-defined genelists is Notepad. This is because Notepad always saves files as ASCII text files, without adding the hidden formats into the file which some word processors do. To open Notepad, click on **Windows Start Button → Programs → Accessories → Notepad**.

To save any list of genes from a column of cells within Excel, simply select the cells by left-clicking and dragging your cursor over the cells, and copying to the buffer through **Edit → Copy** (or **Ctrl-C**). Then open up a blank text file in Notepad, and paste the contents of the buffer into the text file through **Edit → Paste** (or **Ctrl-V**). It is possible to save the list of selected genes from a BRB-ArrayTools scatterplot in this way.

To save any list of genes from an HTML table, first convert to Word or Excel, then copy the desired column of gene identifiers into a Notepad document. Of course, the genelists from the Class Comparison or the Compound Covariate Predictor tool are already automatically saved into the `..\ArrayTools\Genelists\User` folder. However, the user may be interested in saving the genes from an HTML table for purposes other than to enable subset analyses within BRB-ArrayTools, since the HTML tables report the genes in sorted order by p-value or another criterion.

Using the PowerPoint slide to re-play the three-dimensional rotating scatterplot

When running the three-dimensional rotating scatterplot, users who have PowerPoint 2000 or later installed will also have a PowerPoint file created in the `Output` folder of the project folder. The name of this file is specified on the **Options** page of the **Multidimensional scaling of samples** dialog, and is named `MDS.ppt` by default, unless the user changes this name. To play the three-dimensional rotating scatterplot from the PowerPoint file, open up the PowerPoint file, and choose to **Enable macros** when prompted. Once the slide is opened, go into **Slide Show** mode by clicking on the **View → Slide Show** menu item in PowerPoint. Once you are in **Slide Show** mode, then click on the **Click Here to Display** link to view the rotating scatterplot. You may edit the slide in **Edit** mode (such as changing the title, removing the warning label, or changing the caption on the link by right-clicking on it and opening the **Properties** dialog). However, the **Click Here** link will open up the rotating scatterplot only if you click on it during the **Slide Show** mode and not during the **Edit** mode. To go into the **Edit** mode, click on the **View → Normal** menu item in the PowerPoint menu.

Users who do not have PowerPoint 2000 or later will not have this PowerPoint file created for them. However, these users may still save a screenshot of the scatterplot or re-play the rotating scatterplot from a DOS command, as described below. To save a still-shot of the screen, simply press the **PrintScreen** button on the keyboard, then go into any application which has image editing capabilities, and paste the screenshot (usually by pressing the **Ctrl-V** key combination).

Changing the default parameters in the three-dimensional rotating scatterplot

To re-play the rotating three-dimensional scatterplot from a DOS command, follow these instructions. (Note that the following steps can also be used to show the three-dimensional rotating scatterplot with larger point sizes or change the default title on the plot.)

1) Open an **MS-DOS** window (this can usually be found by going to the **Windows Start Button**).

2) Find the folder which contains the `java.exe` file (usually this will be in the `C:\Program Files\JavaSoft\Jre\1.2\bin` folder). Use the `cd` command in the **MS-DOS** window to change to that directory. If the folder is in a different drive, then you may first need to change to that drive by typing either `c:\` or `d:\` at the command prompt.

3) Set the CLASSPATH parameter (in capital letters) to the `java` sub-folder of the `ArrayTools` installation folder.

4) Invoke the `ThreeDimensionalGraph` java application by typing a java command at the prompt, with the following required parameters. (The keyword `ThreeDimensionalGraph` is case-sensitive.)

   ```
   java ThreeDimensionalGraph <mds file> <class file>
   <background> <title> <width> <height> <shape>
   ```

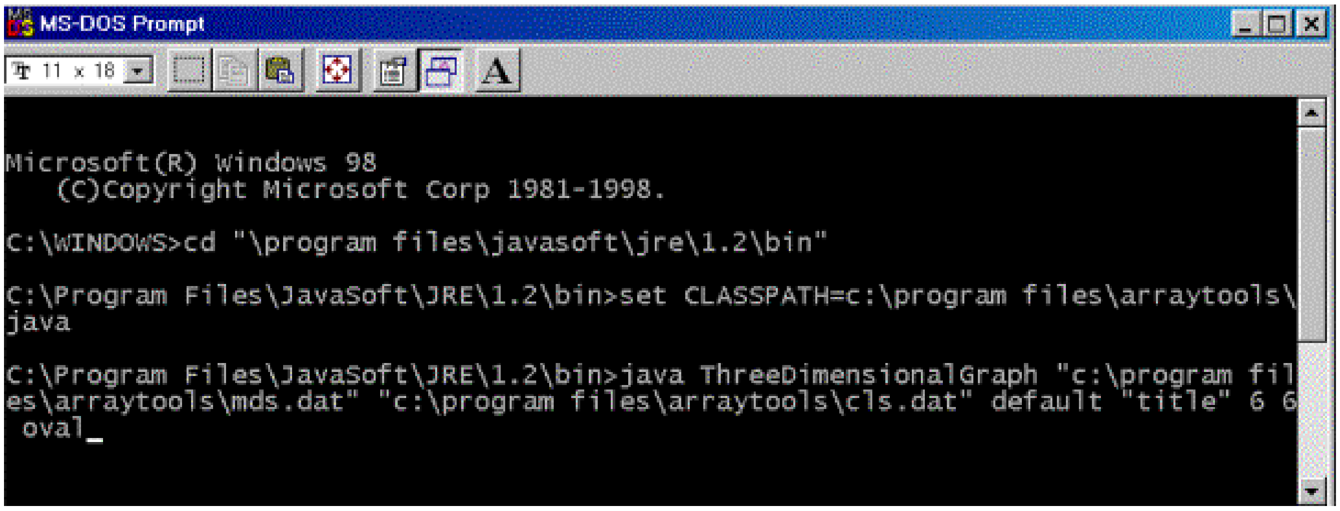
   The command line parameters are as follows:

`mds file` =	the file which contains the multidimensional scaling coordinates along with other data. The full path of the mds file should be given in quotes. This file is usually found as `mds.dat` in the `ArrayTools` installation folder.
`class file` =	the file which contains the class names and associated plotting colors. The full path of the class file should be given in quotes. This file is usually found as `cls.dat` in the `ArrayTools` installation folder.
`background` =	the color to use for the background of the rotating scatterplot. The allowable values for this parameter are: `default` (gray), and the 13 standard Java color names.
`title` =	a string enclosed in quotes, which will be used as the title for the rotating scatterplot.
`width` =	an integer representing the width of the plotted points.
`height` =	an integer representing the height of the plotted points.

`shape` =	the shape of the plotted points. The allowable values for this parameter are: `oval` (becomes a circle when height=width), `rectangle` (becomes a square when height=width), `diamond`, `triangle`, and `star`.

If you find that the DOS command window does not allow you to type in such a long command, then you can move the `mds.dat` and `cls.dat` files to the root folder in order to shorten their pathnames within the DOS command.

The following figure shows an example of a DOS session which was used to re-play the multidimensional scaling plot:



```
Microsoft(R) Windows 98
   (C)Copyright Microsoft Corp 1981-1998.

C:\WINDOWS>cd "\program files\javasoft\jre\1.2\bin"

C:\Program Files\JavaSoft\JRE\1.2\bin>set CLASSPATH=c:\program files\arraytools\
java

C:\Program Files\JavaSoft\JRE\1.2\bin>java ThreeDimensionalGraph "c:\program fil
es\arraytools\mds.dat" "c:\program files\arraytools\cls.dat" default "title" 6 6
 oval_
```

Stopping a computation after it has started running

Sometimes a user may want to cancel a computation after it has started running. If the computation is running in Excel, then the user may stop the computation from within Excel by pressing the **ESC** key, or by pressing the **Ctrl-Break** key combination.

However, if the computation is running within the R Server, it is not possible to cancel the computation from within Excel, but the computation may be stopped through the Windows manager. To stop a computation that is running within the R Server, you must go to the **Windows Task Manager** or **Close Program** window (by pressing the **Ctrl-Alt-Del** key combination), and kill the **StatConnectorSrv** process. In Windows NT, this is usually listed as `STATCO~1.EXE` under the **Processes** window of the **Task Manager**, whereas in Windows 98 this is listed as **Statconnectorsrv** in the **Close Program** window. Once the **StatConnectorSrv** has been terminated, Excel will issue a series of messages notifying the user that an error has occurred. You may ignore these error messages, since the error was induced by the termination of the **StatConnectorSrv** process. If the error message asks if you want to debug, click **No**. After the R-(D)COM has been closed, you should click on **ArrayTools → Utilities → Refresh R Session** to start a fresh R session.

If the computation is running as a batch job within an MS-DOS window, then the computation may be cancelled by simply closing the MS-DOS window.

Automation error

An automation error is an error which occurs when one of Excel's built-in functions is unable to run. This error often occurs at the save step, when the project workbook is unable to be saved to the disk for some reason. One possible cause for this error is that a user's disk may already be full. However, many users have encountered this error during the collation step, even when their disks had not been full. The exact cause of this error is not yet known, as the error is often not reproducible on another computer system running the same software when collating the same dataset. However, a workaround does exist which enables users to continue using their project workbook even when this error does occur.

The project workbook is saved to disk twice during the collation step. Before the collation process actually begins, a template project is saved to disk, containing all the proper worksheets but no data. After the collation process is completed, the project is re-saved to the disk, overwriting this original template. When the automation error occurs during the second save step, the user can often still open the original template project workbook and use it to regenerate all the data. Because the collation process did finish before the automation error at the second save step, all the data has already been written to binary files in the project folder. After opening the empty project workbook, the user may force the project to regenerate all the data by re-filtering the project using a slightly different set of spot filtering criteria (for example, by changing the intensity threshold). The re-filtering process will regenerate all the data which had already been written to binary files during the collation step, and the user may then change the filters back to their original settings, re-filter again, and then save the project workbook manually.

Excel is waiting for another OLE application to finish running

Sometimes when a user is running a large computation (usually occurs if the user is running a hierarchical clustering of more than approximately 4000 genes), the user may receive the following message from Excel: "Excel is waiting for another OLE application to finish running." This message is NOT an error, but merely informing the user that the computation has not yet finished. Because the actual computation for the analysis may be performed in R (such as the case for hierarchical clustering), the Excel application must wait until the computation has finished in R before Excel can continue on. When this message appears, the user can ignore the message and just keep waiting for the computation to finish. Even if the user does not click OK to dismiss the message, the message will disappear on its own. Or if the user does not want to wait, then the user may choose to stop the computation, by following the instructions given above in the section Stopping a computation after it has started running.

Collating data using old collation dialogs

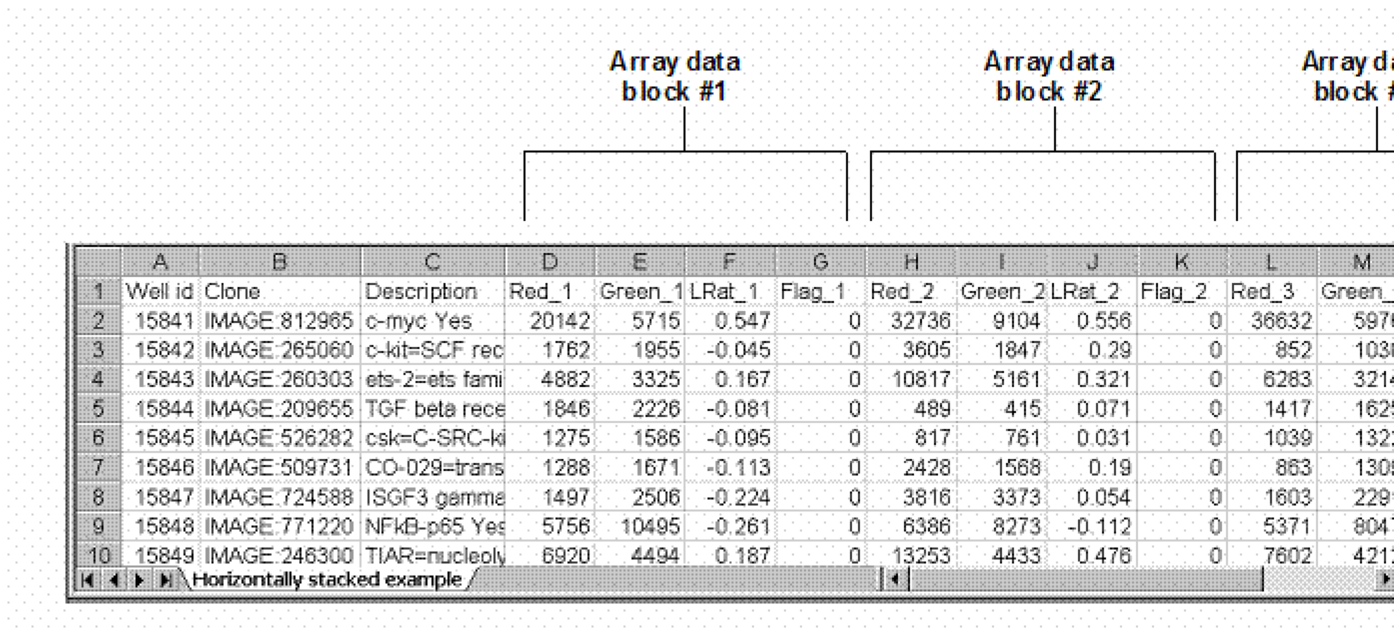
Prior to v3.1, BRB-ArrayTools used collation dialogs in which users must enter the format specifications directly, primarily by specifying column numbers where data elements were to be found. The collation dialogs used in previous versions are no longer necessary, since v3.1 now has a data import wizard which automatically reads the input files and displays the columns which are to be specified. However, the old collation dialogs are still supplied with BRB-ArrayTools v3.1 for backward-compatibility.

Example 1 - Experiments are horizontally aligned in one file

Format of the raw data files

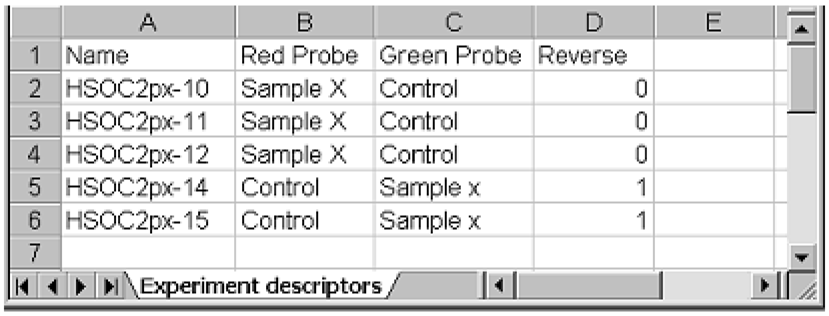
In this file format, the expression data from each array is represented by a block of columns in the data file, and the array data blocks are horizontally aligned in one file. There may be other columns, such as gene identifiers, that precede the array data blocks. However, there should be no other miscellaneous data columns following the last array data block. If there are any other data columns following the last array data block, then the user should either delete those last columns or move them to the beginning of the file so that they precede the first array data block.

The file may be a tab-delimited ASCII text file, or an Excel spreadsheet (a single worksheet within an Excel workbook). If the file is an Excel spreadsheet, then BRB-ArrayTools will automatically convert it to text format. The following figure shows an example of a horizontally aligned expression data file with four columns in each data block:



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Well id	Clone	Description	Red_1	Green_1	LRat_1	Flag_1	Red_2	Green_2	LRat_2	Flag_2	Red_3	Green_
2	15841	IMAGE:812965	c-myc Yes	20142	5715	0.547	0	32736	9104	0.556	0	36632	597
3	15842	IMAGE:265060	c-kit=SCF rec	1762	1955	-0.045	0	3605	1847	0.29	0	852	103
4	15843	IMAGE:260303	ets-2=ets fami	4882	3325	0.167	0	10817	5161	0.321	0	6283	321
5	15844	IMAGE:209655	TGF beta rece	1846	2226	-0.081	0	489	415	0.071	0	1417	162
6	15845	IMAGE:526282	csk=C-SRC-ki	1275	1586	-0.095	0	817	761	0.031	0	1039	132
7	15846	IMAGE:509731	CO-029=trans	1288	1671	-0.113	0	2428	1568	0.19	0	863	130
8	15847	IMAGE:724588	ISGF3 gamma	1497	2506	-0.224	0	3816	3373	0.054	0	1603	229
9	15848	IMAGE:771220	NFkB-p65 Yes	5756	10495	-0.261	0	6386	8273	-0.112	0	5371	804
10	15849	IMAGE:246300	TIAR=nucleoly	6920	4494	0.187	0	13253	4433	0.476	0	7602	421

The following figure shows the experiment descriptor file. The experiment descriptor file describes the samples used in the dataset. This particular dataset is a replication study. Each row of the experiment descriptor file describes the sample on one array. The order of the experiments listed in the experiment descriptor file are assumed to correspond to the order of the array data blocks in the expression data file. Thus, array data block #1 will be matched with HSOC2px-10 in row 2 of the experiment descriptor file, array data block #2 will be matched with HSOC2px-11 in row 3 of the experiment descriptor file, etc. The Reverse descriptor in column D indicates the experiments on which the fluors were reversed.

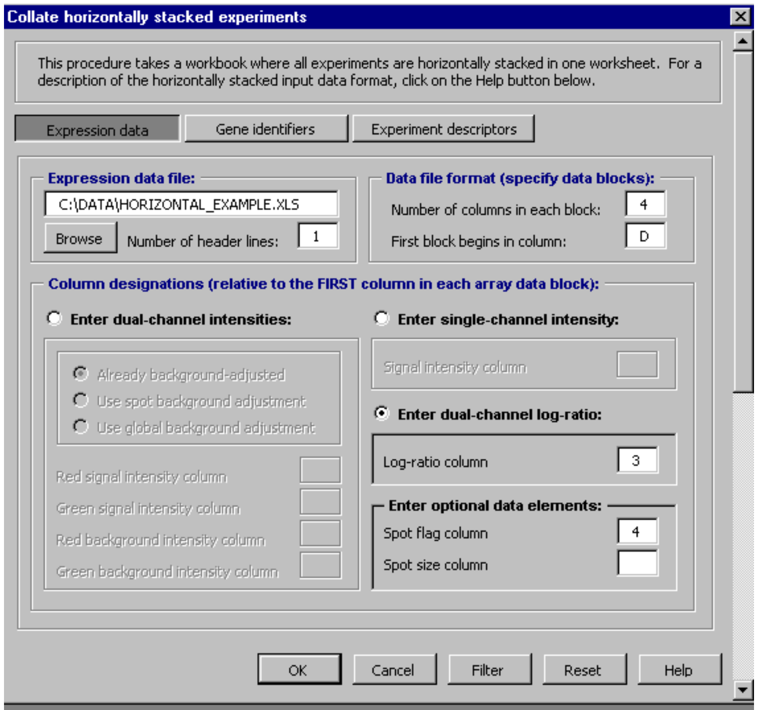


	A	B	C	D	E
1	Name	Red Probe	Green Probe	Reverse	
2	HSOC2px-10	Sample X	Control	0	
3	HSOC2px-11	Sample X	Control	0	
4	HSOC2px-12	Sample X	Control	0	
5	HSOC2px-14	Control	Sample x	1	
6	HSOC2px-15	Control	Sample x	1	
7					

Collating parameters

There are two ways in which the data may be specified in the collating dialog form. One way is to enter the red and green signal columns, and let BRB-ArrayTools compute the log-ratio. Another way is to enter the log-ratio column directly. The user is not permitted to enter both the log-ratio and the dual-channel signals, since the log-ratios can be computed once the dual-channel signals are entered. It is preferred that the dual-channel signals be entered (if they are available) rather than the log-ratio, since BRB-ArrayTools can use the signal intensities for filtering, lowess (intensity-based) normalization, and creating diagnostic scatterplots. If the log-ratios are entered directly, then these functionalities are not available to the user.

The figure below shows the collating parameters which would be entered on the **Expression data** page of the collating dialog box to describe the horizontally aligned data file shown above, if the signal intensities were to be entered rather than the log-ratio. The user should browse for or type in the name and path of the file that contains the expression data. Note that the column designations of the red, green, and spot flag variables are *relative to the first column of the array data block*. In other words, a designation of “4” for the spot flag column means that the spot flag is in the *fourth* column of each array data block. Any extra columns within the array data blocks that are not specifically designated in the **Data variables** section of the collating dialog box, such as the third column (the log-ratios), will be ignored during the collating step. Since BRB-ArrayTools automatically calculates the log-ratio from the red and green signals, it is not necessary to enter the log-ratios as input.



This procedure takes a workbook where all experiments are horizontally stacked in one worksheet. For a description of the horizontally stacked input data format, click on the Help button below.

Expression data file: C:\DATA\HORIZONTAL_EXAMPLE.XLS

Number of header lines: 1

Data file format (specify data blocks):

Number of columns in each block: 4

First block begins in column: D

Column designations (relative to the FIRST column in each array data block):

Enter dual-channel intensities: [ ]

[✓ Already background-adjusted] Use spot background adjustment / Use global background adjustment

Red signal intensity column:

Green signal intensity column:

Red background intensity column:

Green background intensity column:

Enter single-channel intensity: [ ]

Signal intensity column:

[✓ Enter dual-channel log-ratio:]

Log-ratio column: 3

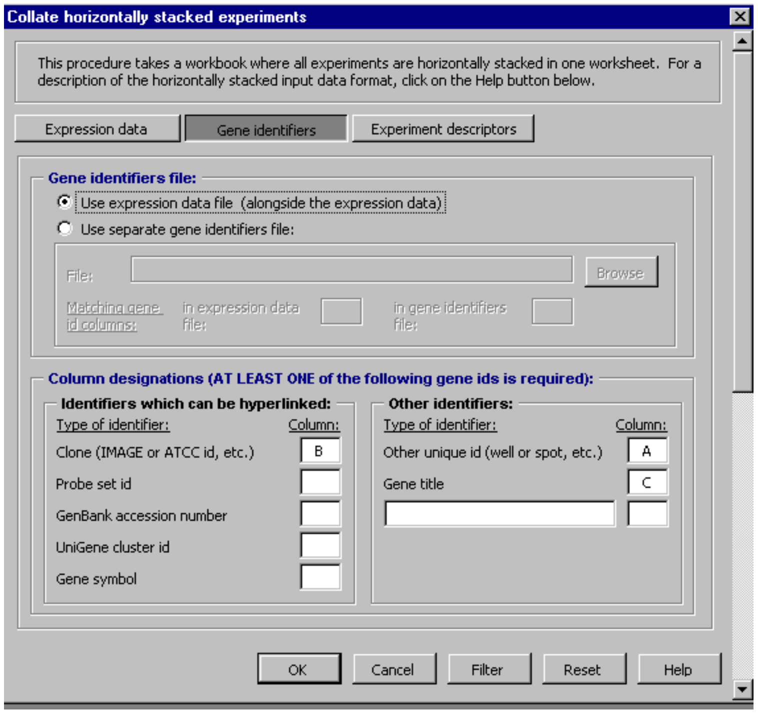
Enter optional data elements:

Spot flag column: 4

Spot size column:

The figure below shows the collating parameters which would be entered on the **Expression data** page of the collating dialog box to describe the horizontally aligned data file shown above, if the log-ratios were to be entered directly instead of the signal intensities.

The figure below shows the collating parameters which would be entered on the **Gene identifiers** page of the collating dialog box. In this case, the gene identifiers are found in the same rows alongside the expression data, not in a separate file. The gene identifiers are specified by their column designation under the **Column format** section, which are given relative to the file specified in the **File** section.



Collate horizontally stacked experiments

This procedure takes a workbook where all experiments are horizontally stacked in one worksheet. For a description of the horizontally stacked input data format, click on the Help button below.

Gene identifiers file:

- [✓ Use expression data file (alongside the expression data)]
- Use separate gene identifiers file:

File:

Matching gene id columns: in expression data file: in gene identifiers file:

Column designations (AT LEAST ONE of the following gene ids is required):

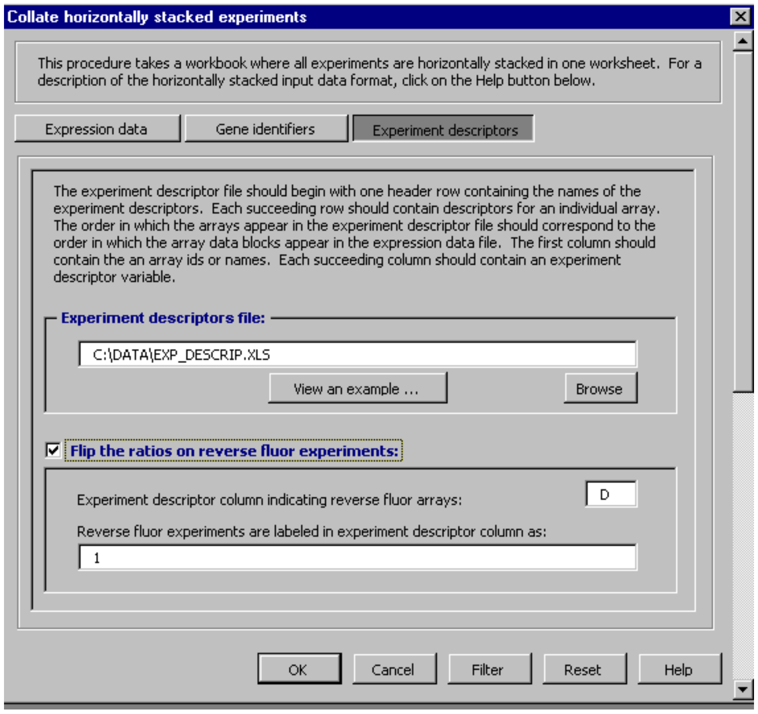
Identifiers which can be hyperlinked:

Type of identifier:	Column:
Clone (IMAGE or ATCC id, etc.)	B
Probe set id	
GenBank accession number	
UniGene cluster id	
Gene symbol	

Other identifiers:

Type of identifier:	Column:
Other unique id (well or spot, etc.)	A
Gene title	C

The figure below shows the collating parameters which would be entered on the **Experiment descriptors** page of the collating dialog box. The user should browse for or type in the file and path name of the experiment descriptor file. In this case, any array with a label of `1` in column **D** of the experiment descriptor file would have its intensity ratios "flipped" so that the log-ratios in that array are "reversed" to have the opposite sign.



Collate horizontally stacked experiments

This procedure takes a workbook where all experiments are horizontally stacked in one worksheet. For a description of the horizontally stacked input data format, click on the Help button below.

The experiment descriptor file should begin with one header row containing the names of the experiment descriptors. Each succeeding row should contain descriptors for an individual array. The order in which the arrays appear in the experiment descriptor file should correspond to the order in which the array data blocks appear in the expression data file. The first column should contain the an array ids or names. Each succeeding column should contain an experiment descriptor variable.

Experiment descriptors file: C:\DATA\EXP_DESCRIP.XLS

☑ Flip the ratios on reverse fluor experiments:

Experiment descriptor column indicating reverse fluor arrays: D

Reverse fluor experiments are labeled in experiment descriptor column as: 1

The **Filter** dialog box, which allows the user to change the default filtering parameters, will be reviewed in a separate section of this manual. After all the collating and filtering parameters have been entered, the user will be prompted to enter a name for the **project folder** to be used exclusively for this dataset, and to enter a name for the **collated project workbook** to be created.

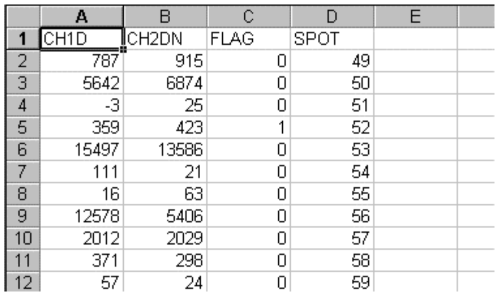
Example 2 - Experiments are in separate files

Format of the raw data files

In this file format, the expression data from each array is stored in a separate file. All files must have the same column format, though the genes contained in the rows of the separate files may differ. Each file may be a tab-delimited ASCII text file, or an Excel spreadsheet (a single worksheet within an Excel workbook). If the files are a Excel spreadsheets, then BRB-ArrayTools will automatically convert them to text format. The `BreastSamples` dataset enclosed with this software package is an example of this data format. This dataset was obtained from published data by Ross, et al.

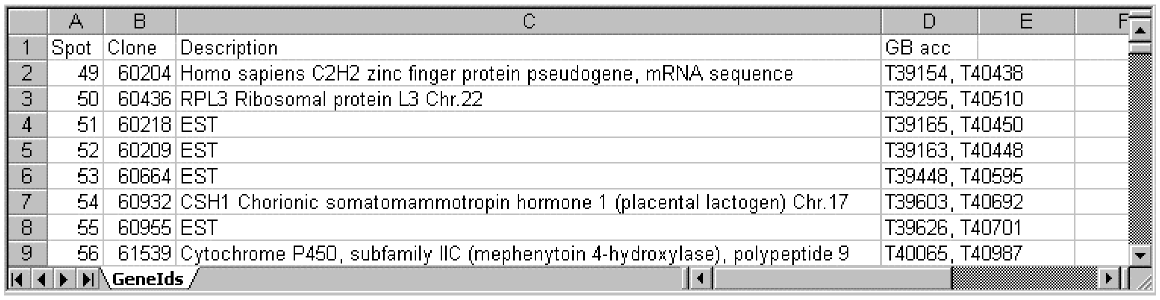
To view the `BreastSamples` dataset, unpack the `BreastSamples.zip` file using WinZip or other utility. When you unpack the dataset, you should get a folder called `BreastSamples`, containing a gene identifier file called `GeneIds.xls`, an experiment descriptor file called `ExpDesc.xls`, and a subfolder called `ExpressionData`. Inside the `ExpressionData` subfolder, there are thirteen expression data files. The expression data files in this dataset are already aligned.

The following figure shows the format of each of the expression data files:



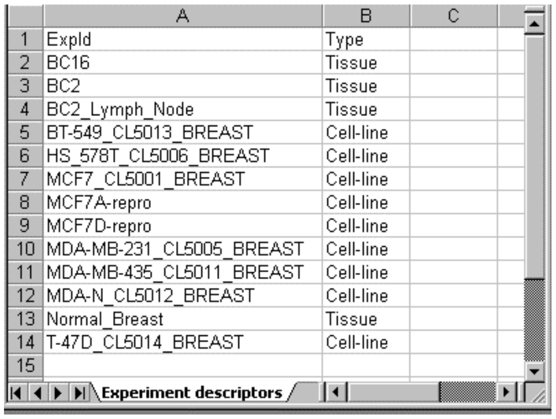
	A	B	C	D	E
1	CH1D	CH2DN	FLAG	SPOT	
2	787	915	0	49	
3	5642	6874	0	50	
4	-3	25	0	51	
5	359	423	1	52	
6	15497	13586	0	53	
7	111	21	0	54	
8	16	63	0	55	
9	12578	5406	0	56	
10	2012	2029	0	57	
11	371	298	0	58	
12	57	24	0	59	

The following figure shows the format of the gene identifiers file:



	A	B	C	D	E
1	Spot	Clone	Description	GB acc	
2	49	60204	Homo sapiens C2H2 zinc finger protein pseudogene, mRNA sequence	T39154, T40438	
3	50	60436	RPL3 Ribosomal protein L3 Chr.22	T39295, T40510	
4	51	60218	EST	T39165, T40450	
5	52	60209	EST	T39163, T40448	
6	53	60664	EST	T39448, T40595	
7	54	60932	CSH1 Chorionic somatomammotropin hormone 1 (placental lactogen) Chr.17	T39603, T40692	
8	55	60955	EST	T39626, T40701	
9	56	61539	Cytochrome P450, subfamily IIC (mephenytoin 4-hydroxylase), polypeptide 9	T40065, T40987	

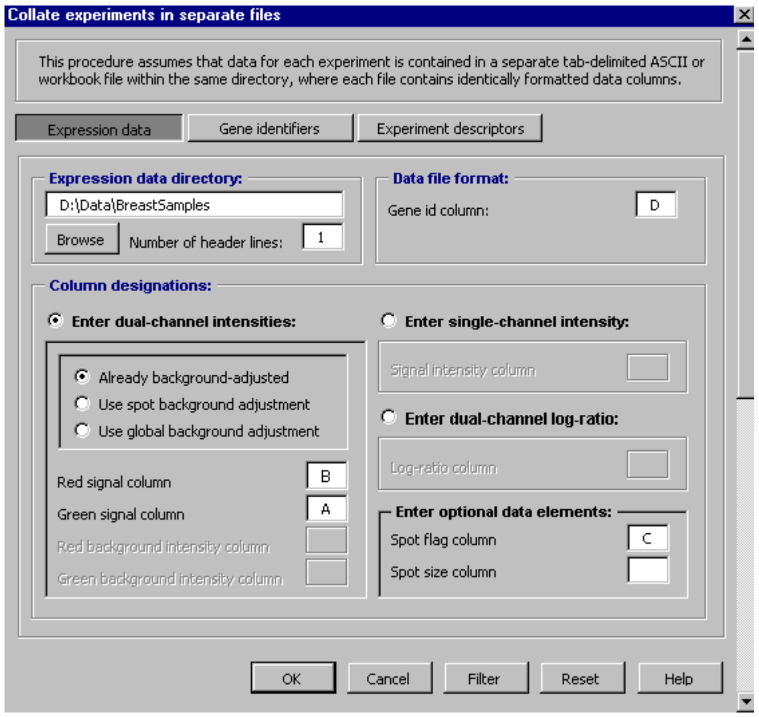
The following figure shows the format of the experiment descriptors file:



	A	B	C
1	ExpId	Type	
2	BC16	Tissue	
3	BC2	Tissue	
4	BC2_Lymph_Node	Tissue	
5	BT-549_CL5013_BREAST	Cell-line	
6	HS_578T_CL5006_BREAST	Cell-line	
7	MCF7_CL5001_BREAST	Cell-line	
8	MCF7A-repro	Cell-line	
9	MCF7D-repro	Cell-line	
10	MDA-MB-231_CL5005_BREAST	Cell-line	
11	MDA-MB-435_CL5011_BREAST	Cell-line	
12	MDA-N_CL5012_BREAST	Cell-line	
13	Normal_Breast	Tissue	
14	T-47D_CL5014_BREAST	Cell-line	
15			

Collating parameters

The following three figures show the collating parameters that would be entered on the **Expression data** page, **Gene identifiers** page, and **Experiment descriptors** page of the collating dialog box, for collating the `BreastSamples` data using `SPOT` as the unique gene identifier that links the expression data with the gene identifiers.


Collate experiments in separate files

This procedure assumes that data for each experiment is contained in a separate tab-delimited ASCII or workbook file within the same directory, where each file contains identically formatted data columns.

Expression data directory: D:\Data\BreastSamples

Number of header lines: 1

Data file format:

Gene id column: D

Column designations:

[✓ Enter dual-channel intensities:]

[✓ Already background-adjusted]

Use spot background adjustment

Use global background adjustment

Red signal column: B

Green signal column: A

Red background intensity column:

Green background intensity column:

Enter single-channel intensity:

Signal intensity column:

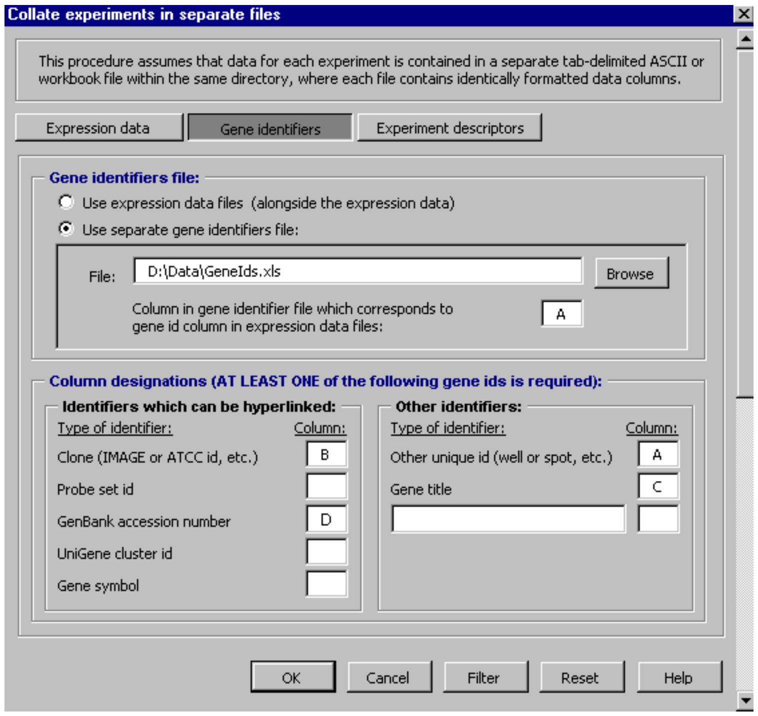
Enter dual-channel log-ratio:

Log-ratio column:

Enter optional data elements:

Spot flag column: C

Spot size column:


Collate experiments in separate files

This procedure assumes that data for each experiment is contained in a separate tab-delimited ASCII or workbook file within the same directory, where each file contains identically formatted data columns.

Gene identifiers file:

Use expression data files (alongside the expression data)

[✓ Use separate gene identifiers file:]

File: D:\Data\GeneIds.xls

Column in gene identifier file which corresponds to gene id column in expression data files: A

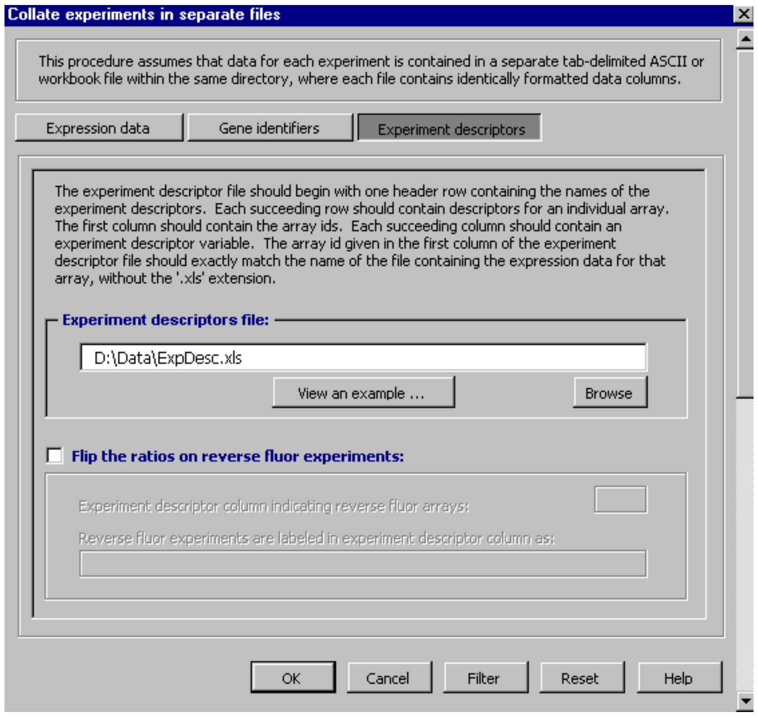
Column designations (AT LEAST ONE of the following gene ids is required):

Identifiers which can be hyperlinked:

Type of identifier:	Column:
Clone (IMAGE or ATCC id, etc.)	B
Probe set id	
GenBank accession number	D
UniGene cluster id	
Gene symbol	

Other identifiers:

Type of identifier:	Column:
Other unique id (well or spot, etc.)	A
Gene title	C


Collate experiments in separate files

This procedure assumes that data for each experiment is contained in a separate tab-delimited ASCII or workbook file within the same directory, where each file contains identically formatted data columns.

The experiment descriptor file should begin with one header row containing the names of the experiment descriptors. Each succeeding row should contain descriptors for an individual array. The first column should contain the array ids. Each succeeding column should contain an experiment descriptor variable. The array id given in the first column of the experiment descriptor file should exactly match the name of the file containing the expression data for that array, without the '.xls' extension.

Experiment descriptors file: D:\Data\ExpDesc.xls

Flip the ratios on reverse fluor experiments: [ ]

Experiment descriptor column indicating reverse fluor arrays:

Reverse fluor experiments are labeled in experiment descriptor column as:

The **Filter** dialog box, which allows the user to change the default filtering parameters, will be reviewed in a separate section of this manual. After all the collating and filtering parameters have been entered, the user will be prompted to enter a name for the **project folder** to be used exclusively for this dataset, and to enter a name for the **collated project workbook** to be created.

Troubleshooting the installation

Using BRB-ArrayTools with updated R and R-(D)COM installations

BRB-ArrayTools is designed to be used with R 1.8.1 or later, and R-(D)COM 1.2 or later. Currently there are no known bugs with using BRB-ArrayTools with R 1.8.1 and R-(D)COM 1.2, other than as mentioned here. If you had already upgraded your R or R-(D)COM to a later version *before* installing BRB-ArrayTools, then BRB-ArrayTools should be able to function with these later versions. However, if you upgrade your R-(D)COM to 1.2 *after* installing BRB-ArrayTools, the R-(D)COM 1.2 installer has the unfortunate effect of automatically unloading BRB-ArrayTools from Excel, because the R-(D)COM 1.2 installer inadvertently overwrites an Excel registry variable which is used by BRB-ArrayTools. If you find that this has occurred and **BRB-ArrayTools** no longer shows up in the **Tools → Add-ins** dialog, then following these steps should restore your BRB-ArrayTools installation:

1) Unload the **Rexcel** add-in from the **Tools → Add-ins** dialog.
2) Uninstall **BRB-ArrayTools** from the **Control Panel's Add or Remove Programs** dialog.
3) Re-install **BRB-ArrayTools** using the original `ArrayTools_v3_0_2.exe` installer file.
4) Now when you open up Excel, **BRB-ArrayTools** should be properly loaded. If you choose, you may then re-load **Rexcel** after re-installing BRB-ArrayTools.

In most cases, un-installing and re-installing BRB-ArrayTools has usually been able to solve problems which have occurred due to upgrading of R. If un-installing and re-installing BRB-ArrayTools does not solve the problem for you, then please send us a bug report to let us know.

Testing the R-(D)COM

If you experience difficulty with the RServer while using BRB-ArrayTools, and you had *not* separately installed the R-(D)COM 1.2. using a separate installation file downloaded from the Comprehensive R Archive Network website, then you may wish to test the R-(D)COM to see if it was registered properly. If you did not already have the R-(D)COM installed before installing BRB-ArrayTools, then the `ArrayTools_v3_0_2.exe` installer should have automatically installed and registered the R-(D)COM into your R 1.6.1 (or latest) binaries folder after unpacking all the files. However, there are glitches which may occur on some user's machines while registering the DLL. To test the R-(D)COM, double-click on the `vbtest.exe` file in your R 1.6.1 (or latest) binaries folder (usually it is something like `C:\Program Files\R\rw1061\bin`). When the **StatConnector Test** screen comes up, click on **Start**. If the R-(D)COM was registered properly, then a script will start to execute and the window will show the current R version being used.

If the R-(D)COM was not properly registered, then you must manually register the R-(D)COM. To manually register the R-(D)COM:

1) Click on the **Windows Start Button** and use the **Find** function to search for the location of the `regsvr32.exe` file. (Usually it will either be in the `system` or `system32` folder of your `Windows` or `Winnt` folder.)

2) Now open a MS-DOS window (usually by clicking on **Windows Start Button → Program → Command Prompt**). Within the MS-DOS window, use the `cd` command to switch to the folder containing the `regsvr32.exe` file. Use the `dir` command to double-check that you have the folder containing that file.

   i.e., type something like the following sequence of commands:

```
c:
cd c:\winnt\system32
dir regsvr32.exe
```

3) Now type something like the following command:

```
    regsvr32  "c:\program
files\r\rw1061\bin\StatConnectorClnt.dll"
```

You should substitute the location of your own R installation folder for `c:\program files\r`. You may or may not need the quotation marks, but you can try it both ways. You will know that the command has succeeded when you get a window popping up with a message like:

"DllRegisterServer in c:\program files\r\rw1061\bin\StatConnectorClnt.dll succeeded."

4) Once the `regsvr32` command has succeeded, go back and test the R-(D)COM again by double-clicking on the `vbtest.exe` file in your R 1.6.1 (or latest) binaries folder.

Spurious error messages

If you are a previous user of BRB-ArrayTools v2.1 or earlier, you may get spurious error messages such as "This workbook is referenced by another workbook and cannot be closed" when you open up Excel. Although this message is not harmful and should not affect the operation of BRB-ArrayTools, it can be very annoying. If you are getting this error message, then go to the **Tools → Add-ins…** dialog (while you have an active workbook open, or else the **Add-ins** menu item will be greyed out), and deselect both the **BRB-ArrayTools** add-in and the **RServer** add-in (if it is still showing in the list of available add-ins). Now close down and re-open Excel. When Excel has re-opened, go back to the **Tools → Add-ins…** dialog, and this time select only the **BRB-ArrayTools** add-in. From now on, leave the **RServer** add-in deselected (if it is still listed), and keep the **BRB-ArrayTools** add-in selected. This should keep the spurious error messages from coming up.

Reporting bugs

To make comments or ask questions of a general nature, please post a message on the message board at the BRB-ArrayTools website:

http://linus.nci.nih.gov/BRB-ArrayTools.html

Bug reports should be e-mailed to the BRB-ArrayTools Development Team at: arraytools@emmes.com. Because many emails get automatically deleted as spam, please follow these guidelines to ensure that your email gets past the filter:

1) Always include an appropriate subject header which indicates the topic of the email.
2) If you are sending attachments, then please send them in a SEPARATE email from the main text, so that we will know in case the email with attachments gets deleted by our virus filter.

Due to limited resources, we cannot guarantee that we will be able to diagnose and fix all bugs reported by users. However we do try our best to respond to users within a reasonable time, and to advise the user or fix the problem whenever we can.

References

Published references

S Dudoit, J Fridlyand, TP Speed; Comparison of discrimination methods for the classification of tumors using gene expression data, Journal of the American Statistical Association 97:77-87, 2002

I Hedenfalk., D Duggan, Y Chen, M Radmacher, M Bittner, R Simon, P Meltzer, B Gusterson, M Esteller, M Raffeld, et al. Gene expression profiles of hereditary breast cancer, New England Journal of Medicine 344:539-548, 2001.

EL Korn, JF Troendle, LM McShane and RM Simon. Controlling the number of false discoveries: Applications to high-dimensional genomic data. Journal of Statistical Planning and Inference (In Press).

LM McShane, MD Radmacher, B Freidlin, R Yu, MC Li and RM Simon. Methods of assessing reproducibility of clustering patterns observed in analyses of microarray data. Bioinformatics 18:1462-1469, 2002.

MD Radmacher, LM McShane and RM Simon. A paradigm for class prediction using gene expression profiles. Journal of Computational Biology, 9:505-511, 2002.

D Ross, U Scherf, M Eisen, C Perou, C Rees, P Spellman, V Iyer, S Jeffrey, M Van de Rijh, M Waltham et al. Systematic variation in gene expression patterns in human cancer cell lines, Nature Genetics 24:227-235, 2000.

RM Simon, EL Korn, LM McShane, MD Radmacher, GW Wright and Y Zhao. Design and Analysis of DNA Microarray Investigations, Springer, New York NY, 2003.

V Tusher, R Tibshirani and G Chu. Significance analysis of microarrays applied to transcriptional responses to ionizing radiation. Proceedings of the National Academy of Sciences USA 98:5116-5121, 2001.

GW Wright and RM Simon. A random variance model for detection of differential gene expression in small microarray experiments. Bioinformatics 19:2448-2455, 2003.

Technical reports

In addition, the following two references may be found as technical reports from the BRB website: http://linus.nci.nih.gov/~brb/TechReport.htm

Technical Report 001:A paradigm for class prediction using gene expression profiles. by Michael D. Radmacher, Lisa M. Mcshane, and Richard Simon

Technical Report 002:Methods for assessing reproducibility of clustering patterns observed in analyses of microarray data. by Lisa M. McShane, Michael D. Radmacher, Boris Freidlin, Ren Yu, Ming-Chung Li, and Richard Simon

Acknowledgements

The BRB-ArrayTools Development Team currently consists of Richard Simon, Amy Peng Lam, Michael Ngan, Leonid Gibiansky, and Yi Xia. This software uses many algorithms and programs developed by various members of and people associated with the Biometrics Research Branch. Many code fragments and hints were also taken from the Excel Developer Tip Archives (http://www.j-walk.com/ss/excel/tips/index.htm) from JWalk & Associates, Inc. Lastly, acknowledgements go to the R Core Development Team for their production of the R software, and Thomas Baier and Erich Neuwirth for their development of the R Server and Component Object Model (http://www.ci.tuwien.ac.at/R). The R Server distributed with BRB-ArrayTools has been modified from its original form. For details of the modifications, please see the 'Readme.doc' file.

Support Vector Machine algorithm implemented in the BRB ArrayTools was developed by Chih-Chung Chang and Chih-Jen Lin (see http://www.csie.ntu.edu.tw/~cjlin/libsvm/).

Cluster 3.0 software was developed by Hoon et. al (M. J. L. de Hoon, S. Imoto, J. Nolan, and S. Miyano: Open Source Clustering Software. *Bioinformatics*, 2003, in press.). It is represents an enhanced version of Cluster, which was originally developed by Michael Eisen of Berkeley Lab.

To view the clustering results generated by Cluster 3.0, we use Alok Saldanha's Java TreeView. Java TreeView is not part of the Open Source Clustering Software.

License