Troein, C., Vallon-Christersson, J., and Saal, L. H. (2006). An introduction to BioArray Software Environment. *Methods Enzymol.* **411,** 99–119.
Tusher, V. G., Tibshirani, R., and Chu, G. (2001). Significance analysis of microarrays applied to the ionizing radiation response. *Proc. Natl. Acad. Sci. USA* **98,** 5116–5121.
Zhang, L., Miles, M. F., and Aldape, K. D. (2004). A model of molecular interactions on short oligonucleotide microarrays: Implications for probe design and data analysis. *Nature Biotechnol.* **21**(7), 818–821.

[9] TM4 Microarray Software Suite

By ALEXANDER I. SAEED, NIRMAL K. BHAGABATI, JOHN C. BRAISTED, WEI LIANG, VASILY SHAROV, ELEANOR A. HOWE, JIANWEI LI, MATHANGI THIAGARAJAN, JOSEPH A. WHITE, and JOHN QUACKENBUSH

Abstract

Powerful specialized software is essential for managing, quantifying, and ultimately deriving scientific insight from results of a microarray experiment. We have developed a suite of software applications, known as TM4, to support such gene expression studies. The suite consists of open-source tools for data management and reporting, image analysis, normalization and pipeline control, and data mining and visualization. An integrated MIAME-compliant MySQL database is included. This chapter describes each component of the suite and includes a sample analysis walk-through.

Introduction

The Human Genome Project was envisioned as a grand endeavor that would change biology by providing a catalog of genes in humans and other model organisms. Although a large number of genome sequencing projects, including that of the human genome, have been declared finished, the collection of the sequence itself has not fundamentally altered our approach to understanding biological systems. Rather, it has been the development of techniques and technologies that allow us to analyze patterns of expression for sets of genes, proteins, or metabolites approaching the total number that are active in an organism at any given point in time.

Since their introduction in 1995 (Lipshutz, 1995; Schena, 1995), DNA microarrays have matured significantly to become the most widely used technique for the analysis of global patterns of expression and represent a technology that is now used routinely as a means of generating testable hypotheses prior to other studies. DNA microarrays consist of an arrayed

METHODS IN ENZYMOLOGY, VOL. 411

0076-6879/06 $35.00
DOI: 10.1016/S0076-6879(06)11009-5

collection of probes bound to a solid substrate that are used to interrogate the levels of gene expression using hybridization to labeled nucleic acids and detection of those hybridization events. Although microarray technology is still evolving, the development of robust and reliable commercial platforms, combined with a significant decrease in the cost of an assay, has resulted in an explosion of gene expression data. The challenge of doing an expression profiling experiment is no longer in the generation of data, but rather in effectively capturing the information and using it to explore the biology of the systems under study.

In that regard, the role of software in a study involving microarrays cannot be overstated. Specialized tools are available to complement the experimental procedure and subsequent data analysis. Data management software is used to capture vital information describing the laboratory portion of a microarray experiment. Scanned microarray slides are processed and quantified using image analysis software. Normalization utilities ready data for comparisons and further analysis. Data mining and visualization tools can then help explore data from many perspectives. When used together, such software becomes a system to maximize the utility of the microarray experiment and gain better insight into the biology of interest.

We have developed a suite of software applications to support gene expression studies. This suite, called TM4, consists of a comprehensive set of tools that allow users to collect, manage, and effectively analyze data from microarray experiments. This chapter describes the TM4 suite and each of its components. The chapter concludes with an example analysis using a real data set and several analysis techniques.

The four major applications of TM4 are Madam, Spotfinder, Midas, and MeV. Each application in the suite is publicly and freely available. This includes the source code, which is OSI certified as open source under the artistic license (http://www.opensource.org/licenses/artistic-license.php).

Madam is the primary data entry, tracking, and reporting system of TM4. A series of data entry forms provide users with an organized method of recording their experimental parameters and data. Query and reporting tools present important data on a variety of entities, such as a single hybridization or an entire study. This application also serves as a repository for other tools in the data management and reporting realm. These include a polymerase chain reaction (PCR) scoring and microtiter plate loading utility, a study design tool, and a free-form SQL query window. Madam works closely in conjunction with a MIAME-compliant relational database to carry out its functions. The role of such a database is described elsewhere (Troein *et al.*, 2006).

Spotfinder is a multichannel image analysis tool. This application provides the means to load the output of a microarray scanning operation—typically a

pair of 16-bit tagged image format file (TIFF) images (Timlin, 2006). Semi-automatic grid construction and several methods to adjust the placement of each grid cell manually allow for accurate spot detection. The intensity of each spot can then be quantified and written to an output file along with related spot parameters and flags (Minor, 2006). A number of quality control displays are available, helping users detect systemic issues in slide production.

Midas is a normalization and filtering tool used to process raw data output from Spotfinder and prepare it for further analysis and data mining. Users create a project file, chaining together multiple normalization, filtering, and quality control (QC) modules, using an intuitive graphical workflow builder. The input options provide ways to consistently process single, paired, or whole studies worth of raw expression data. An intuitive graphing system illustrates the effects of normalization with a variety of detailed plots. These graphs can be embedded in a Midas summary report, a pdf-formatted file that also contains a description of the data processing procedure used.

MeV is the main data analysis and visualization tool of TM4. Users can load raw or normalized data from a variety of input file types. A broad range of algorithms is available, including those for clustering, classification, and statistical tests. The intuitive graphical interface simplifies navigation between algorithm results. An integrated scripting interface and XML-based format provides a means to analyze data sets in a regimented and reproducible fashion.

Although these applications were designed with interconnectivity in mind, each piece can be used independently of the others. Aside from the .mev format of TM4 (tab-delimited text with standardized column headers and comment rows), several other popular input and output formats are supported. While originally designed for two-dye fluorescent microarray systems, TM4 has been expanded to support other technologies, such as the Affymetrix Genechip platform (Dalma-Weiszhauz *et al.*, 2006).

A SourceForge web site (http://sourceforge.net/projects/tm4) serves as the central code repository for TM4. This site also hosts the application downloads, user mailing lists, and discussion forums. The TM4 development team actively provides technical support via email. System requirements for each application are detailed in the documentation included with the download. The entire TM4 suite, including software, documentation, and sample data, can be downloaded from http://www.tm4.org.

The TM4 suite was originally developed at The Institute for Genomic Research, under the direction of principal investigator Dr. John Quackenbush. Grants to Dr. Quackenbush for TM4 development were provided by The National Cancer Institute, The National Science Foundation, The National Heart, Lung and Blood Institute, and the NHLBI's Programs for Genomics Applications (PGA). Details regarding the ongoing development of TM4

and the teams responsible are available at the aforementioned SourceForge site. Beyond the main TM4 development team, many organizations and individuals have contributed to this open source project. Their contributions and affiliations are listed in the documentation for each application. The development of TM4 continues through collaborative efforts of groups worldwide, but with work now concentrated at three primary sites: John Quackenbush and his group at the Dana-Farber Cancer Institute and Harvard School of Public Health; members of the Pathogen Functional Genomics Resource Center's microarray software group at The Institute for Genomic Research; and Roger Bumgarner and his group at the University of Washington.

MADAM

Madam (also referred to as MADAM) is the data manager of TM4. It handles the tasks of data entry, tracking, and reporting while serving as an interface to a relational microarray database. Madam offers a series of data entry pages, which provide the user an easy method to load the database with information about their microarray experiments. Several report types display vital information about various stages of the experiment and let the user track the progress. Madam also houses several distinct tools with data management functions.

In addition to these roles, Madam is also capable of generating output in the MicroArray Gene Expression Markup Language (MAGE-ML) format. The submission of microarray data to public repositories is often required when publishing the results of a microarray study. MAGE-ML is the standard format for microarray data exchange and submission. If the user populates the database via the data entry pages correctly, Madam can generate MAGE-ML files that describe the entire microarray experiment. Two popular microarray data repositories are ArrayExpress (Brazma *et al.*, 2003, 2006) and Gene Expression Omnibus (GEO; Barrett and Edgar, 2006; Edgar *et al.*, 2002).

Madam is distributed with a MIAME-compliant relational database and the MySQL database platform. The database is a critical component for the operation of this software and nearly all of the functions of Madam involve interactions with the database in some manner. Madam cannot function without the database. Accordingly, Madam has features that assist the user with MySQL database installation and administration, including the creation of user accounts and Java Database Connectivity (JDBC) configuration.

Madam was designed with two-channel spotted arrays in mind, but efforts are currently underway to expand the interface and underlying database to accommodate other platforms as well, including Affymetrix GeneChips. Detailed operating instructions for this application can be found in the program manual included with the software distribution.

MAD: The Microarray Database

The database includes over 60 tables that store information about nearly every aspect of the microarray experiment. Some tables track the identities of the genes, clones, and oligonucleotides that compose spotted arrays and the microtiter plates in which they are located. The slide geometry is similarly important, including the number of spotted elements and printing pens, spacing between blocks, and the identities of the plates used. Each hybridization and the two probes used in each hybridization, including their origins, are also tracked. Data from outside the microarray laboratory are also stored, such as postimage analysis expression results in raw and normalized forms. The database schema (Fig. 1) illustrates each of the tables and the fields they contain, as well as the links between them.

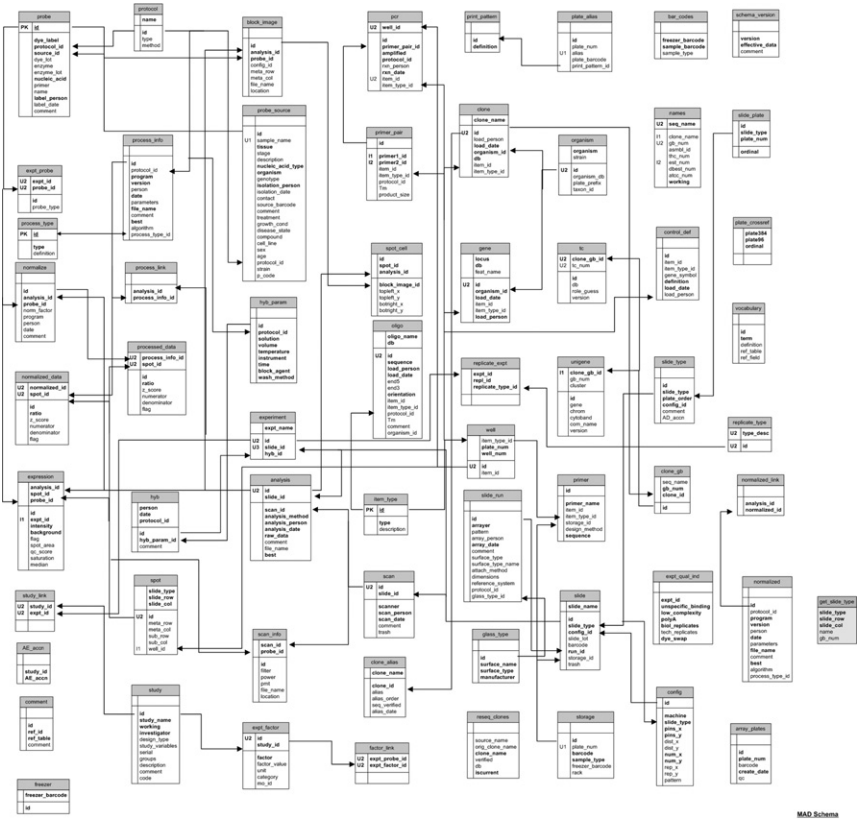

FIG. 1. Schema of the microarray database, MAD. These tables are used extensively for every function of Madam.

Many of these fields store data required by the MIAME (Minimal Information About a Microarray Experiment) specification (http://www.mged.org/Workgroups/MIAME/miame.html). The MIAME specification (Brazma *et al.*, 2001) describes data that are needed to enable the interpretation of the results of a microarray experiment in an unambiguous manner and to possibly reproduce the experiment. It follows that Madam users should try to enter information as completely and correctly as possible to ensure that the MIAME requirements are met. More information about MIAME can be found elsewhere in this volume.

Although the included database is built for the MySQL DBMS, other relational databases can be used. Madam has been connected to both Sybase and Oracle DBMS.

Madam Interface

The main Madam interface (Fig. 2) consists of four parts that are contained within an application window. A menu bar runs across the top of the window and provides access to the *File, Entry, Tools,* and *Help* menus. The *Help* menu contains the *Help Manual* menu item, a detailed and interlinked guide to the Madam interface, and all of the functions of the application. The *File, Entry,* and *Tools* menus contain items relevant to specific aspects of Madam and are described in subsequent sections.

The Navigation Panel is located on the left side of the interface. It consists of a set of tabs: *Entry, Edit, Report, Application,* and *MAGE-ML*. These correspond to each of Madam's major functions. Selecting a tab will display relevant controls for that function in the area immediately below the tabs.

The working panel is found on the right side of the interface. This is the area where most of the activity is based; the content will change depending on the task the user is working on. It can display forms for data entry and MAGE-ML writing, HTML-based reports, and some interfaces for supplementary tools.

At the bottom of the interface is the event log. This area reports important system messages, errors, and significant user activities and is persistent through all the aspects of the software.

Data Entry and Editing Pages

The task of loading data into the microarray database is facilitated through the use of data entry pages of Madam. Each page corresponds to a specific entity, such as a labeled probe or a glass slide. The data entry pages are active by default when Madam is started. To navigate here at any time, the user can click on the *Entry* tab in the *Navigation Panel*. When the

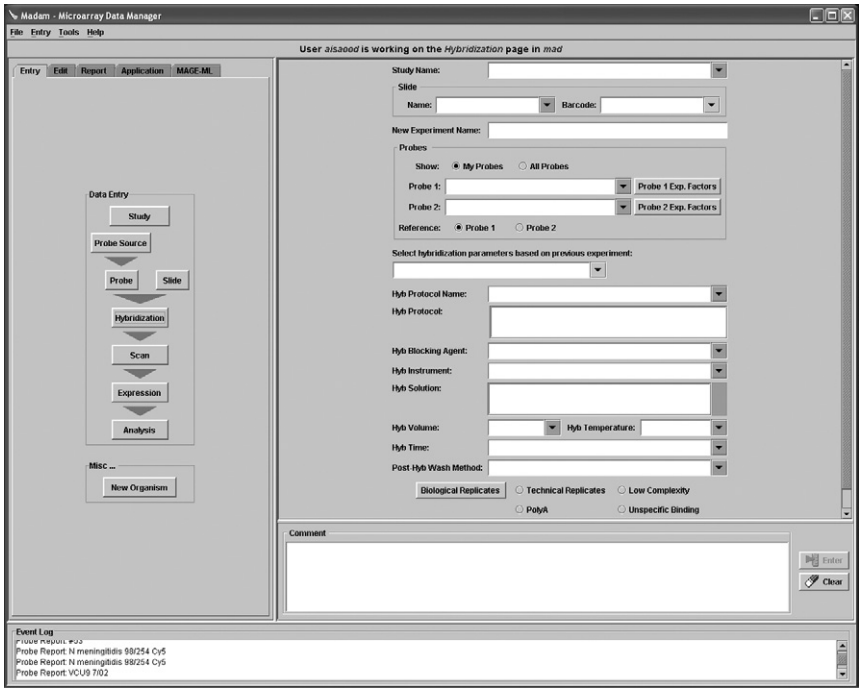

FIG. 2. Madam graphical user interface. Data entry buttons are visible in the *Navigation Panel* on the left side. The *Working Panel*, on the right, is currently displaying the *Hybridization* entry form. The *Event Log* at the bottom records recent activity and displays system messages.

Entry tab is selected, the *Navigation Panel* will display a flowchart of sorts, consisting of ordered buttons. Each of these buttons corresponds to one of the data entry pages and the layout of the buttons mimics the typical order in which each page will be used. Clicking on one of these buttons will bring up the appropriate data entry page form in the *Working Panel*. Another method to navigate to a data entry page is by selecting the desired page from the list contained in the *Entry* menu of the main menu bar.

Every data entry page shares several common features designed to simplify the process of filling out the form. All fields have a descriptive label alongside them to indicate the role of the field. Each entry field corresponds to a table and field combination in the database and this information is sometimes valuable to the user. By holding the mouse pointer over the field label for a moment and reading the tooltip, the user can learn both the table and the field name used in the database to store data entered in that part of the form.

Real-time validation is in effect and fields whose contents are not valid are noted by a red color, either within the text field itself or by a red element next to the input area of the fields. The user can view the reason by holding the mouse pointer over the field for a moment and reading the tooltip. Some fields are colored when the form is first displayed. This is because those fields are required; because they do not have values by default, they fail the validation for this reason.

Many of the drop-down lists in Madam accept input from the user either by clicking one of the entries of the list or by typing text into the field. Typically these lists are populated with data directly from the database and as such they can become quite long. The user can type a few characters into the field and hit the enter key, thus removing all items from the list that do not start with the characters already in the field.

At the bottom of every form is a text area for comments and *Clear* and *Enter* buttons. The *Enter* button indicates that the user is finished entering data in the form and wants to proceed to the next step. It is important to note that the *Enter* button will be disabled if any fields on the form are not currently valid. Clicking the *Enter* button brings up a confirmation dialog. This dialog displays a table with each field and the corresponding value noted. The user is given the opportunity to review the contents before beginning the upload of these data. If there are any errors, clicking the *Cancel* button will return the user to the entry page. If everything is as desired, clicking the *Submit* button will start the process of uploading data into the database. The progress bar and descriptive text messages in the dialog will indicate the status of this process.

The *Study* entry page captures information about a series of related microarray hybridizations (experiments) and the variables and experimental parameters involved. The *Probe Source* page is used to upload details about the biological source of a labeled probe used in hybridization. With this data, the *Probe* page can be used to enter information describing the probe itself, including the fluorescent dye used.

Describing the design of a microarray, from the geometry of the spots to the identities of each array element, is perhaps the most complex data entry task in Madam. This information is loaded using a page called *Slide*. To simplify the task, there are three different methods that can be used to create a new slide entry in the database. The user is free to choose the most appropriate method.

Two probes and one slide can be selected to form hybridization. These selections and details about the protocol and chemistry of the hybridization can be entered using the *Hybridization* page. The next page, *Scan*, is ready to record the settings used when the aforementioned slide is scanned. This page also records information about the TIFF image files that are produced by the scanner as output.

The *Expression* page is useful for uploading the raw intensities and flags calculated for each spot during the image analysis phase. This page requires a .mev format expression file as input. Information about normalization and data processing can be entered using the *Analysis* page, but the focus of this page is changing as the storage requirements for normalized and analyzed data evolve. Future releases of the software will reflect the current standards.

The *New Organism* button is set apart from the rest of the data entry buttons, as it is not used commonly. The *New Organism* entry page appears in a separate window when invoked and can be used to insert a new organism into the database. Selecting the appropriate organism is important when defining the source of a labeled probe or uploading plates using the *PCR Score* tool, described later.

Once data are uploaded, it cannot be altered using these data entry pages. Madam instead offers a set of data-editing pages. These can be accessed by clicking on the *Edit* tab of the *Navigation Panel*. Doing so will bring up a set of list boxes where the user can select the name of the entity to edit. Five options are available: *Study, Probe Source, Probe, Slide,* and *Experiment*. Selecting a name and clicking the arrow next to it will display a data-editing form in the *Working Panel*. These forms function in the same manner as the data entry forms, including real-time validation. The main difference is the replacement of the *Enter* button with the *Edit* button. Clicking *Edit* will bring up a confirmation dialog nearly identical to the data edit confirmation. Only the fields that have changed will be displayed and both the current and original values will be shown alongside the field names. Clicking the *Submit* button in this dialog will begin the process of updating the database with these edited data.

Report Generation

Through the reporting interface of Madam, a user can view and export HTML-based reports that encapsulate vital details about entities that were uploaded using the data entry pages. The user can click on the *Report* tab of the *Navigation Panel* at any time to bring up the report selection interface. The *Navigation Panel* will display five sets of controls for selecting study, experiment, slide, slide type, or probe reports. For each type there is a drop-down list and a *View* button. To generate a report of a given type, the user should select the appropriate entity identifier from the drop-down list and click the *View* button. The selected report is shown in the *Working Panel*.

Each time a report is generated its name will be added to a list near the bottom of the *Navigation Panel*. This list can be used to track the history of viewed reports and quickly recall one by clicking on the name. At the bottom

of the panel there are four buttons. The *Save* button can be used to output the currently visible report as an HTML file for later viewing in a web browser. The *Print* and *Print Preview* buttons both send the visible report to a printer; the latter also shows an image approximating the report's printed appearance. Finally, the *Clear* button resets the entire report interface.

MAGE-ML Writing

MAGE-ML is a standardized XML format for microarray data that has gained wide acceptance in the community. It can be used to distribute microarray descriptions and results to colleagues or for submissions to public microarray databases. Submitting microarray data to public databases such as ArrayExpress or GEO is important when publishing the results of microarray experiments, thus giving others the ability to view your data sets and potentially reproduce the results.

Madam provides a means of writing MAGE-ML files. To do this successfully, the user must first make sure all their data have been entered accurately and as completely as possible using the data entry pages. Missing data can cause problems in the MAGE-ML files that are produced. The MIAME-compliant database that is associated with Madam is capable of storing the necessary information to produce complete MAGE-ML files.

The MAGE-ML interface is invoked by clicking on the *MAGE-ML* tab in the *Navigation Panel*. The panel will then display a tree containing the various objects that can be encoded into the MAGE-ML format. Selecting an object from the tree will bring up a form in the *Working Panel* that is specific for each object type. The user can fill in this form in the same manner as the data entry pages. Once all the fields pass the validation test, the MAGE-ML button at the bottom of the form can be clicked, thus starting the file writing process.

Each object that can be written requires the user to describe the people involved with the project, from the experiment itself to the file generation and submission. These people can be selected from a list and labeled with a role. The list is populated by using the *Preference* menu item located in the *File* menu. Names, organizations, and contact information for all the requisite people can be stored through this interface and retrieved from within the MAGE-ML writing forms.

Two chapters in this volume describe in more detail the importance of MAGE-ML and how it is used in the context of submissions to public microarray databases. Further information about the MAGE-ML object model and file format can be found at http://www.mged.org/Workgroups/MAGE/mage.html.

Related Tools

Madam serves as a home for several distinct tools that are involved for some data entry, retrieval, and management tasks. Although each part can operate outside the context of Madam, there are several advantages when they are bundled together. Perhaps the most important is the utility of having Madam regulate the database administration tasks. As such, it is not necessary for any of these smaller tools to try to establish a new database connection or to start and stop the database software. Having all the tools accessible from one location is also convenient for users and facilitates interactivity between each piece.

These six tools are accessible from the *Application* tab of the main navigation pane. The user can click the button that corresponds to the desired tool to launch it. An alternative is to use the *Tools* menu from the main menu bar and select the appropriate tool from the list. Each of these tools included with Madam is described.

ExpressConverter handles file conversion operations for the TM4 suite. It accepts a variety of common scanner input formats, including GenePix and Agilent, and converts them to the .mev format. For other file types, *ExpressConverter* offers a customizable file converter that allows the user to describe their tabular text format and then convert files of that type to .mev. Annotation files can also be created using this customized converter to complement the expression files.

ExptDesigner is a tool that can help plan a series of microarray hybridizations (Ayroles and Gibson, 2006; Neal and Westwood, 2006). Loop and reference experiment designs are supported. The user begins by selecting the probes from the database that are involved in these hybridizations. These probes will be available for selection in the design view panel, an interface that consists of two visual tools. Users can create hybridization by drawing a directional arrow from one probe to the other in the network view or by clicking a square corresponding to the two desired probes in the matrix view. Each hybridization is then added to a list that can be exported.

PCR Score is an application designed to create and manage data associated with the microtiter plates used for microarrays (Eads *et al*., 2006). Users can upload information describing the contents of 96-well plates, including those containing oligonucleotides and PCR products. In the latter case, a scoring interface allows the user to indicate the success of the PCR reactions. The 96-well plates can then be combined into the 384-well plates often used for array printing.

Mabcos is a microarray bar-coding system. This application prints bar codes for several types of laboratory objects, including freezers and microtiter plates. Series of bar codes can be scanned, tracking the probes, plates,

or slides involved in an experiment. Bar-code scanning helps ensure that the microarray printing is performed correctly. Data can be transferred from a traditional scanner or a PalmOS-based device.

Miner is a tool that writes .mev format expression files from data in the database. The user can specify an *Experiment Name* that corresponds to the hybridization for which to retrieve data. A filter based on PCR results and a buffering option for partial files are available.

The *Query Window* is an interface for submitting free-form SQL queries directly to the database. This supplements the data entry, editing, and reporting functions of Madam by providing a finer level of control over data. Users who are familiar with the schema of the database and the SQL syntax can perform a wide range of operations, including insertions and deletions. Query results and the corresponding SQL can be saved to text files.

Administration Tools

The administration aspects of Madam are handled by a related application called the *Madam Administrator*. This program can be launched from the same directory as the main Madam executable. Some common administrative tasks that can be performed include the creation or removal of local array databases, changing the JDBC connection settings used by Madam to communicate with the database, and changing the values that appear in some parts of Madam's data entry pages and MAGE-ML forms. Details of these operations are found in the program manual included with the Madam distribution.

Spotfinder

Image processing is a key component of the microarray experiment (Minor, 2006; Timlin, 2006). Each two-color spotted microarray slide will typically produce two gray-scale 16-bit images in TIFF format. Each image corresponds to a single labeling dye such that the two images complement each other spatially and need to be processed in parallel. The microarray TIFF image is the end product of the portions of the microarray experiment conducted in the laboratory.

Despite being digital media, in essence, and storing all necessary information about the conducted experiment, the image file is not a data set ready for data analysis (Minor, 2006). The goal of image analysis is to digitize microarray image files and produce output data sets for each slide. These data sets can then be normalized and used as input for clustering, visualization, and statistical analysis tools. The image processing software itself is a specialized tool that provides parallel analysis of microarray TIFF images.

Critical steps include the definition and digitization of spots, calculation of local background, and reporting intensities into output data files.

Image Analysis Goals

The challenge is spot detection and digitization or extraction of intensities. The spots on an array correspond to the genes printed on the slide during the printing step. The hybridization procedure attaches two fluorescent markers on the same target for every spot. After slide scanning at two different excitation wavelengths, two separate images are generated, one for each fluorescent marker. It is commonly accepted to refer to these two images and data extracted from them as the two channels of the microarray experiment. The fluorescent dye signal is expected to be proportional to the overall efficiency of the hybridization as well as the gene expression level. By measuring the integrated signals from both dyes on every spot, we are able to approximate the level of gene expression for both conditions of hybridization.

Two main problems must be addressed before spot intensities can be measured: spots have to be localized spatially on the image and a local background value has to be estimated for every spot. The problem of individual spot locations cannot be solved globally for the entire image. Rather, a good approach is to proceed locally by splitting the image area into subarrays, each of which consists of a group of spots or even individual spots.

Background correction of the measured microarray spot intensities is a procedure used to derive true values from raw experimental data. This correction aims to remove the additive components from multiple sources: substrate background, cross-talk from the other channel dye, nonspecific hybridization response, and so on. Because a spotted microarray image has a nonuniform background distribution over the whole image area, local background correction becomes highly desirable. It is commonly accepted that the background estimated locally for every spot is the best method, if correction is desired. Background correction normally results in expansion of the dynamic range of data for both intensities and ratios; the outcome is more prominent for low-intensity spots and almost invisible for highly expressed genes. It should be viewed as favorable data transformation, as it increases resolution on the expression ratio scale and makes it easier to distinguish the differences between genes with close expression ratios.

Background correction may result in some negative intensity values. This can occur when the local background estimate is larger than the spot intensity, as it is for some weak spots (''black holes''). These spots can be filtered out automatically by applying the signal-to-noise ratio criteria. Realistically, such filtering will not result in data loss, as the weak spots cannot be considered as reliable data points in downstream analysis.

Approaches

There are a few approaches to solving the spot location problem. All of them utilize a known geometrical pattern of printed arrays; arrays typically consist of subgrids arranged in meta-rows and meta-columns. The subgrids themselves consist of the spots that form rows and columns in a rectangular pattern, allowing simple description by a few parameters. A less common geometry of a hexagonal or ''orange packing'' layout of spots is also possible and can be described by a small number of parameters. Using this, one method of determining spot locations is to apply a predefined grid to the image that can be aligned manually or automatically in subsequent steps. A user-assisted semiautomatic grid alignment is also an option. Another solution is not to use a grid explicitly, but apply an automatic or semiautomatic procedure for spot positioning based on the geometric pattern of the printed spots. Two examples of this include the spot-finding procedure based on seeded region growing algorithm (Yang *et al.*, 2002b) or Fourier transform based procedure (Gaidukevitch *et al.*, 2000).

There are not many ways to estimate the local background around a spot. The most popular method is based on the assumption that the background surrounding a spot is the same as the background in the spot itself. Using the rectangular area around the spot for local background estimation by this method is possible if a grid has been defined. The only caveat of this method is the possible overestimation of the background when the pixels closest to the spot are considered part of the background. This systematic bias can be avoided by not involving pixels from the area immediately surrounding the spot in the background calculation. While this method, with some modifications, is widely used by many image analysis tools, one can argue that the real background in the spot area may stem from the nonspecific hybridization or nonlabel fluorescence signal from the target area. Measurement of nonhybridized or nondye hybridized sites has been suggested as an experimental method for the background estimate (Yang *et al.*, 2001) and, on occasion, has been used effectively in practice (Johnston *et al.*, 2004).

Spotfinder 3

Since its first release in 1999, Spotfinder has passed through many upgrades and version changes. Spotfinder 3.0, released in 2004, was a significant redesign of the traditional architecture: a multiplatform application allowing the analysis of arrays containing more than two dyes. Currently, Spotfinder executables are available for three major desktop platforms: Windows, Linux, and Mac OSX. Due to the large size of the TIFF images that are usually analyzed (often 20 MB or more), it is recommended that Spotfinder be run on computers with at least 256 MB of RAM and a CPU

clock speed of at least at 800 MHz. A 16-MB video card (32+ is better) is strongly desirable for Windows desktops. The latest release, Spotfinder 3.1.0, accepts both 8- and 16-bit TIFF images, stored in separate files or one multi-TIFF image file. Data output is generated as tab-delimited text files (.mev) and platform-independent binary files for grids (.sfg) and whole raw data sets (.sfd).

Spotfinder has an intuitive graphical user interface (GUI) with a menu bar on top, dialog tool box on the lower left side, and a number of tab pages in the center of the main program window (Fig. 3). The menu bar contains *Image*, *Grid*, *Data*, and *Settings* menus. The collection of tabs gives the user access to the pages: *General, Overlay, Analysis, RI plot, Data,* and *QC view*. The *Analysis* page is the most functionally rich element of the Spotfinder GUI; all user activities in Spotfinder related to grid design and alignment are focused on the *Analysis* page. The dialog tool box is a control for alternative selection of several dialogs. *Gridding and Processing* has the controls for setting the parameters for automatic grid adjustment and choosing a segmentation method. The *Post-Processing* dialog can be used for *QC filter* settings and for turning on/off the background correction algorithm. The *Cell Editor* dialog controls can be used to activate the *Cell Editor* for interactive cell selection on

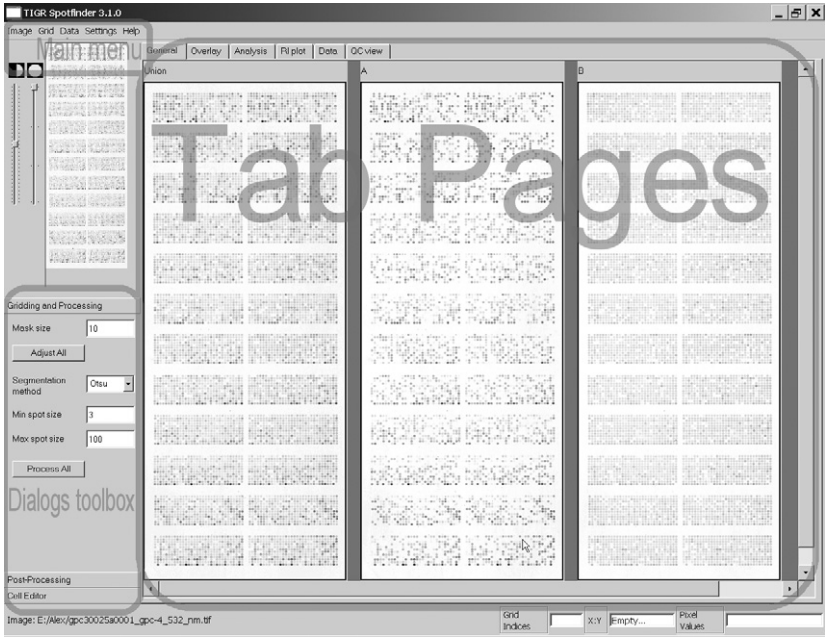

FIG. 3. Spotfinder graphical user interface.

the *Analysis* page, *RI plot* page, or *Data* page. When active, the *Cell Editor* displays the selected spot with its cell shown. The *Cell Editor* also can be used to resize and move cells, and to reprocess the selected spot.

General Steps

The following sequence is typical for a grid-based spot finding procedure using a local background calculation method over the area surrounding each spot.

Grid Composition

Designing and constructing a new grid requires the entry of several slide-specific parameters that are usually determined when the slide is printed. The user can get this information from the settings log file of the printing robot or by making measurements directly on the image file using Spotfinder. Normally the spots on the array are printed in blocks (sometimes referred to as grids or subgrids). This discussion deals with rectangular packed grids. The blocks are organized into meta-rows, which are rows of grids, and meta-columns, or columns of grids. The number of meta-rows and meta-columns on the array is defined by the number of the printing pins in the *X* and *Y* dimensions. Note that all spots in a single block are printed by an individual pin. The set of parameters needed for grid construction include the number of pins in *X* dimension (pinX), the number of pins in *Y* dimension (pinY), spot spacing (distance between spots, measured in pixels) in both *X* and *Y* dimensions, and the number of rows and columns in every block. To measure these parameters on screen, the user can switch to the *Analysis Page* and use the mouse pointer to navigate it to any spot on array, making a reading of the mouse cursor coordinates at the bottom of the program window. The initial spot spacing parameter, which defines the whole grid size, can be set with significant tolerance. As a result, blocks may be smaller or larger than the array of spots that they cover; this can be ignored for the moment.

Grid Expansion and Shrinking

The first step in grid alignment is to move the whole grid set to a desired position. The best way to place the entire grid is to keep track of the upper left-most grid at first and use it as an alignment indicator. After the upper left point of this grid is positioned in place, the spot spacing can be adjusted to fit the correct grid size. It can be done by applying *grid expand*, sequentially increasing or decreasing the spot size until each rectangle of the grid has a spot centered within. The suggested method is to watch only the upper left-most grid when applying this procedure to all subgrids on the array.

Ideally, all pins in the print head are evenly spaced; hence, all printed blocks should have a consistent offset (George, 2006). However, it is often found the pins are slightly bent; as a result, the blocks of spots are misaligned relative to each other. After positioning and sizing the upper leftmost subgrid we can proceed with adjusting the position of the other subgrids. The automatic grid adjustment procedure can be used to iteratively place the remaining subgrids. This procedure uses a mask, of a size no less than the typical spot size, to calculate the target function for every block. This mask is assigned to every spot cell and centered. The target function is defined as the integral of all the pixels in the spot cells under all the masks in the grid. Every block is moved in small steps to a maximum of one cell size up, down, left, and right. At every step the integral is calculated and stored as an element of a two-dimensional (2D) matrix. After this has been completed, the 2D matrix is searched for the maximum calculated integral value. The subgrid is then moved to the position corresponding to that maximum value. This automatic procedure requires that three conditions are met. First, every subgrid has to be set correctly in size and with the correct row and column numbers. Second, each subgrid has to be aligned roughly, within a tolerance of one cell size. Finally, at least half of the spots in the block should have significant signal (i.e., strong spots) to provide reliable landmarks to position the subgrid correctly. It is important to note that this procedure may give unsatisfactory results for subgrids with empty rows or columns on the edges, that is, the first and last columns or rows. The automatic grid adjustment can be applied repeatedly, as sometimes it is necessary to use the procedure a few times to reach sufficient grid alignment. If repeated applications do not produce satisfactory results, the only solution is to align the grid manually. The user, in this case, can use the mouse or keyboard arrow keys to position each subgrid accurately.

Spot Detection

Spot detection, also referred as spot finding or segmentation, is the next key step of image analysis. The goal of spot detection is to separate spot pixels from background ones. The image has to be segmented or divided in two subsets: one including the signal or spot pixels and the other consisting of the background pixels. A number of methods are widely used for the segmentation of microarray image spots. It has been shown (Yang *et al.*, 2001) that the choice of segmentation method applied has no significant effect on the ultimate results of image analysis. All segmentation methods used in microarray image processing can be categorized as being either histogram or shape based. The histogram-based methods do not take into account the spatial information about the analyzed spot, such as spot shape. Instead, they apply a sorting algorithm to the whole set of pixels

in the cell and set a threshold that separates signal and background pixels based only on the values of the pixels. Auxiliary input parameters, such as spot size and spot pixel dynamic range, can be used to facilitate the histogram segmentation. In contrast, the shape analysis-based methods mainly rely on spatial information of the image; they look primarily on how pixels with high values are grouped together spatially. A few software tools use a method based on a seeded growing algorithm (Yang *et al.*, 2001, 2002); this is related to the shape-based methods.

The original segmentation method implemented in Spotfinder is a histogram-based algorithm that expects only a single parameter from the user: estimated spot size. This method provides good results for images with a low variation in spot size. However, images can have significant spot size variation in certain conditions. One cause could be the variation in temperature and humidity during the slide printing. Spotfinder introduced a segmentation method based on the *Otsu* algorithm (Liao *et al.*, 2001; Otsu, 1979) to address this problem. This method requires the user to input two parameters: minimum and maximum spot size on the array. The *Otsu* method runs through the original histogram of pixel values and places a threshold dividing the area into two groups—background and signal. The threshold returned by this method maximizes between-group variance. A third segmentation option in Spotfinder is a manual method in which the user interactively applies a predefined circle as mask for spot segmentation.

Spot Digitizing

Following the detection of spot boundaries by the segmentation method, the next step in image analysis is spot quantitation or digitizing. During this stage the pixels inside the spot are counted and added together to calculate the integrated intensity of each spot. The spot mean and median can also be determined.

The important issue of pixel saturation is addressed at this step. Due to the natural limitation of the dynamic range, the value of each pixel cannot exceed the 16-bit maximum, which is 65,536. If the input fluorescence signal is too high and exceeds the linear range, the corresponding pixel is assigned the maximum possible value of 65,536. In principle, the problem can be solved by rescanning the slide with different settings for sensitivity or scanner detector power. Doing so can bring all pixel values to a lower range but also may set low-intensity spots at the background level. Alternatively, if the percentage of saturated pixels is not too high, they can just be ignored and a rescan of the slide can be avoided. The check for saturated pixels is conducted when pixels inside a spot are analyzed. If any pixel in the spot is saturated at least in one channel it will be excluded, that is, removed from the spot data set in all channels. All reported values for the spots—mean, medians, integrated

intensities, and their standard deviations—are computed after the saturated pixels are removed. As a measure of this correction the resulting saturation factor is reported. The saturation factor is defined as the ratio of the non-saturated pixels over the original number of pixels in the spot.

Local Background Correction

Because the direct measurement of spot background is not feasible, the local background estimate is based on some assumption. The simplest way is to assume that the background in the area surrounding a spot is the same as inside the spot itself. The local background can then be measured by analyzing the pixels just outside the spot boundary. Normally the local background is estimated by the median of these exterior pixels. To apply background correction to integrated intensities, this value must be multiplied by the spot area and subtracted from the raw integral intensity.

Reported Parameters

Spotfinder reports a number of parameters for each spot that is detected: integrated intensity, mean, median, total background (integral), background median, background standard deviation, integrated intensity standard deviation, mean standard deviation, median standard deviation, flag, QC score, and p value. These parameters are reported for each channel in the spot; if it is a two-dye array there will be two integrated intensities, etc. Some parameters are reported only once for the spot, independent of the number of channels. These include spot area, saturation factor, and total QC score.

Quality Control Parameters

The QC procedure, which analyzes and reports QC parameters, is an important part of any image processing software. Spotfinder provides a number of QC parameters: total and individual channel spot QC scores, flags assigned by the QC filter, and p values from a two sample t test. Flags assigned to each spot by the QC filter may change based on the QC filter settings. QC scores and p values, however, are independent from the QC filter settings.

The QC filter is enabled by default and performs spot shape and signal-to-noise ratio analysis based on the user-set parameters. The spot shape is analyzed under the assumption that the ideal spot has to be similar in shape to a circle. Real spots may have a less circular shape that can become even more distorted if the detected spot has originated from background fluctuations rather than a true target. The ratio of spot area to spot perimeter is calculated to check if it is significantly different from the ratio for a perfect circle of equal size. If the spot ratio deviates from that of the circle by more than 20% it can be considered a badly shaped spot. The signal-to-noise

ratio check is based on the selected pixel value threshold. The threshold can be written as

$$T = \alpha M + \beta SD, \quad (1)$$

where M is local background median, SD is background standard deviation, and parameters α and β are coefficients and can have values in the range of $[0, 4]$. The default settings are $\alpha = 1, \beta = 1$. The spot is considered strong and will pass this test if more than 50% of the pixels in the spot are higher than the selected threshold. Any spot that fails on at least one of the criteria will be flagged as a ''bad'' spot. Spots that pass these tests are subject to further scrutiny. The number of pixels in these spots is counted (spot area) and different warning flags are assigned to spots smaller than 50 and 30 pixels. The user may choose to disable QC filtering, ending up with an output data set that has every detected spot flagged as ''good.''

QC scores will be calculated and reported by Spotfinder, even if the QC filter is disabled by the user. The QC score has a value between 0 and 1; higher scores indicate better spot quality. The total QC score for each spot is the mean of QC scores for that spot from all channels. The QC score for each channel is calculated as the geometric mean of the shape and the signal-to-noise QC scores for the spot. The shape QC score is calculated in the same way as it was described earlier, but also including normalization to the spot size and scaling into the range [0, 1]. The signal-to-noise QC score is defined as the portion of pixels in spot above T, where T is defined by Eq. (1) with $M = 2$ and $SD = 0$.

Visualization of Quality Controls in Spotfinder Views

Spotfinder provides a set of visual displays for graphical presentation of QC results. The user can navigate to them using the GUI tabs *Analysis*, *RI plot*, *Data*, and *QC view*.

As a major hub of the Spotfinder interface, the *Analysis* page also graphically displays the immediate results of image processing—the contours of each detected spot are painted in one of two colors depending on the flag assigned by the QC filter: magenta for good flags (A, B, C, and S) or green for bad flags (X, Y). The shape and the location of the contour inside every cell give the user immediate visual information about the alignment of the cell and the success of spot detection (Fig. 4).

The Spotfinder *RI plot* page (Fig. 5) with ''diamond plot'' lines (Sharov *et al.*, 2004) can be used for quick visual examination of the slide's ratio and signal level dynamic range, correctness of background detection, and saturation correction. A ratio-intensity plot (RI plot), widely used for presentation of microarray data, is the $\log_2(MA/MB)$ plotted as function of $\log_2(MA*MB)$, where MA and MB are spot means in channel A and B, respectively. Diamond lines indicate the theoretical limits for spots with

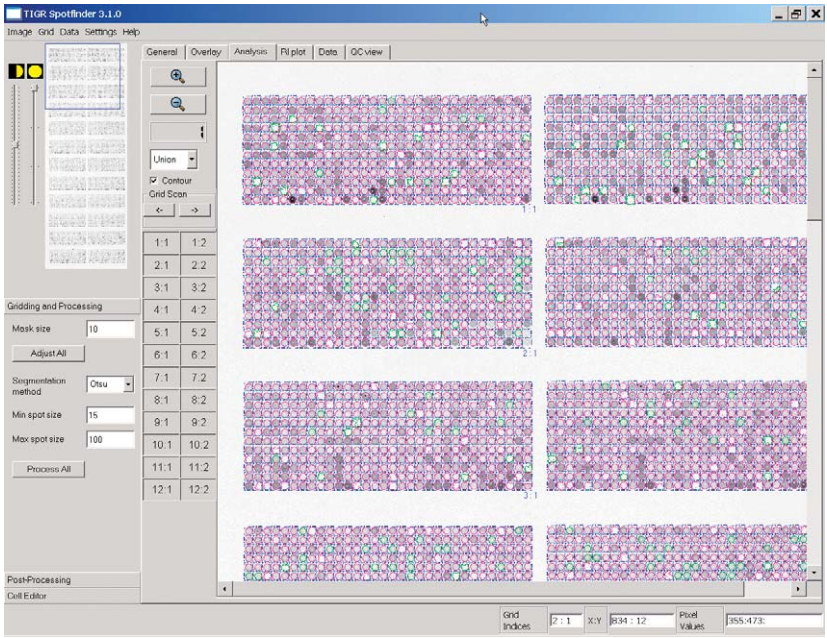

FIG. 4. Analysis page shown after whole slide processing is complete. Contour lines are colored red for good spots and green for bad ones.

extreme intensities: completely saturated spots at least in one channel are limited by the right side of the RI plot diamond and spots with zero intensity in at least one channel are limited by the left side of the RI plot diamond. Correspondingly, the left-most tip of the diamond is the location of spots in which both channels produce zero intensities, and the right-most tip of the RI plot is the location of spots with complete saturation in both channels. None of the spots on the array should be expected outside of the RI plot diamond. In essence, the RI plot diamond lines are the physical limits for bit-depth limitations in one channel on the right side of diamond and zero measured signal on the left side of the diamond.

The *QC view* page allows the user to view the subgrids with individual cell rectangles colored according to a four-color scheme. Three colors—yellow, blue, and gray—are used to indicate spots with measured differential expression levels above, below, and between two chosen preset levels, respectively. These two levels of $\log_2$(ratio) are preset to 1 and –1, by default, to display those genes that are up- or downregulated by a factor of two or more in blue and yellow colors, and coloring genes that fall within that range as gray. They can be changed by the user to visualize the interesting expression ratio

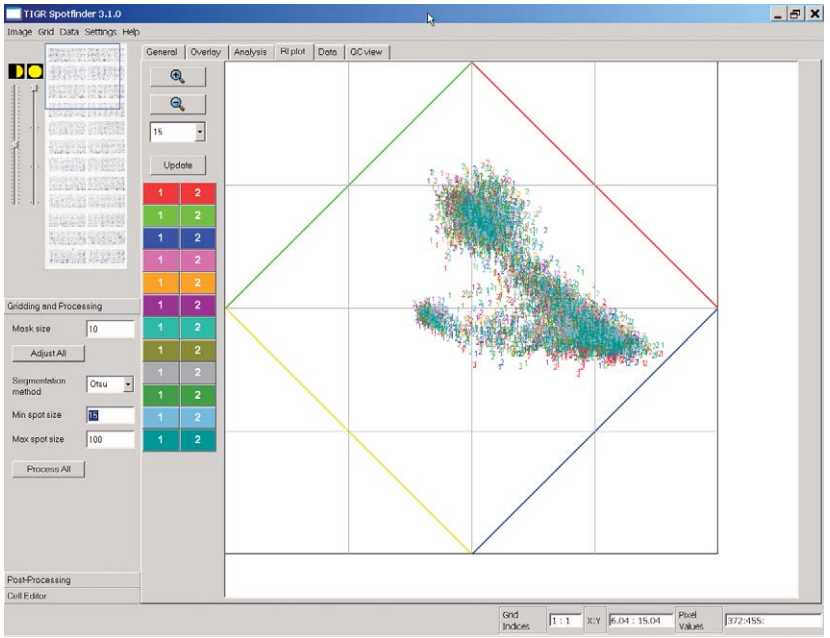

FIG. 5. RI plot view in Spotfinder showing the ratio-intensity log graph for the whole analyzed slide. The four lines forming a diamond are the limits of the log-ratio plot. Red and blue lines are the full saturation limits lines in one channel, whereas the other channel has a valid number. Yellow and green lines are the zero values limit lines at least in one channel.

profile on the slide. For instance, if the user is looking for fourfold up and downregulated genes, the values of 2 and –2 should be entered on the *QC view* page. The color green is used for bad or undetected spots. Colored cells can be displayed on QC view in their true positions in the subgrid or combined together in blocks. In the latter method the area of rectangular blocks is proportional to the number of cells of a certain color on the subgrid. Relative amounts of bad/undetected color cells in subgrids are expected to be approximately the same; therefore, any noticeable increase of green color areas may indicate poor alignment of that particular subgrid.

Spotfinder Protocol Description

Program Settings

Check the program settings and change them based on the actual slide type; if the number of channels is changed, it is necessary to close and restart the program. The user also may change the visualization scale factor

depending on the image size and available video card memory; it is recommended to keep default settings for the initial use. Use the menu bar to go to *Settings→General Settings*. Make sure that the *Channel Number* is set to 2 for a two-dye experiment and that the *Scale Factor* also equals 2.

Image Loading

To load two TIFF images stored in distinct files, select both TIFF files at once by holding the keyboard Ctrl key, clicking on one file and then on the other one. For loading TIFF images stored in the same file, select only that one file to load. Spotfinder automatically detects if the selected file is encoded in 8- or 16-bit format. The file names are sorted in alphabetic order for placement in channels A and B for a two-color array, and A, B, C, and D for a four-color array. The output data file columns will follow the same order. The user may swap the images in channels A and B to change the order if necessary.

Loading Existing Grid from File

A previously saved grid can be retrieved from the SFG file by clicking on the *Load grid from file* option from the *Grid* menu. The Spotfinder focus should be switched to the *Analysis* page by clicking the *Analysis* tab. This page is where the user will interact with Spotfinder for grid construction, alignment, movement, and processing tasks. If any arrays of this same type have been analyzed previously, load the grid file used previously for this slide type. The grid would likely involve only a position alignment, as it has the correct grid size but not necessarily the correct location.

Grid Construction

New array types require the construction of a new grid. The Spotfinder grid design assumes that the slide was printed by using rectangular pin (pen) settings. Every distinctive subgrid or block on the slide is printed by one dedicated pin. The pins are arranged in rectangular pattern such that the subgrids form meta-rows and meta-columns. To design a new grid, go to the menu *Grid→Compose Grid*. This will bring up the *Grid Design* dialog. In this dialog the user is asked to input eight parameters describing the geometry of the grid. These parameters are the numbers of meta-rows and meta-columns, the distance between neighboring pins in horizontal (PinX) and vertical (PinY) dimensions, the number of rows and columns in each subgrid, and the spot spacing in the horizontal and vertical dimensions. These parameters can often be retrieved from the slide print specification used by the robot that printed this slide. If this specification is not

available, all parameters can be evaluated interactively on the Spotfinder *Analysis Page* by measuring relative distances with the mouse pointer. All distances are expressed in image pixels. Each of the subgrids can be moved, rotated, expanded, or shrunk interactively to fit the spots arrangements on an image. These operations can also be applied to all the subgrids simultaneously.

1. *Grid movement.* Move the whole grid set to align the upper left-most spot with the top left cell of the first (upper left) subgrid. To do this, click the right mouse button while the mouse pointer is not inside any grid. The *All Grids* menu will be activated. Choose *Move All* from the *All Grids* menu and move the mouse slowly or use keyboard arrows keys to move all subgrids simultaneously into the appropriate position. When the placement is correct, terminate the *Move* mode by pressing the *End* key on the keyboard or by clicking the left mouse button. The user can repeat this action as many times as is necessary. Undo/redo grid commands are available for convenience.

2. *Grid expansion.* If the number of rows and columns is set correctly but the subgrids do not fit the image, it may be necessary to expand or shrink the subgrids. Bring up the *All Grids* menu (as described earlier) and choose the *Expand All* command. By using the keyboard arrow keys, the user can expand or shrink all subgrids together, either horizontally or vertically. Both the expansion and the shrinking operations are performed while keeping the left and top edges of each grid fixed. Only the right and bottom edges of each subgrid are moved when these operations are performed.

3. *Changing cell size in grid.* If it is necessary, the cell size can be adjusted by selecting *Cell Size All* from the *All Grids* menu. Use the keyboard arrow keys to increase or decrease cell size in the vertical and horizontal dimensions. One arrow key press corresponds to a cell size increase or decrease by one pixel. Terminate the *Cell Size All* mode when finished by pressing the *End* key on the keyboard or by clicking the left mouse button. When increasing cell size try to avoid touching the spots by growing neighboring cells. As long as this touching can be avoided, overlapping of the adjacent cells is actually safe and desirable because it increases the area around each spot for local background calculation.

4. *Grid rotation.* Spotfinder provides the ability to rotate subgrids for better alignment of images that have an angular offset. Activate this command mode by choosing *Rotate All* from *All Grids Menu* and use the up and down arrow keys on the keyboard to rotate all grids synchronously. The rotation of each subgrid is performed around its top left corner. It is better to use only the upper left-most subgrid as an indicator of alignment. The rest of the subgrids are expected to have the same angular offset

due to the nature of the parallel arrangement of the pins during slide printing.

The user can repeat steps 1–4 in any sequence any number of times for all subgrids or any single selected subgrid to improve grid adjustment. Make sure at this point to use only the first subgrid (top left) as an indicator of proper grid size settings when the *ALL Grid* command is used.

Grid Adjustment

After setting the correct grid size the top left subgrid should be aligned and positioned correctly while the others likely have some positional offset due to the natural bending of the printing pins. These subgrids can be adjusted manually or by using the automatic procedure. The automatic procedure requires the user to provide an estimated spot size. Spot size is used to detect the location of the brightest spots that serve as landmark targets in each subgrid. The automatic grid adjustment procedure can be applied as many times as needed; in many situations it comes to satisfactory grid alignment after a few applications. However, if it fails to provide a good grid adjustment the user must adjust the grid manually by using the mouse and keyboard arrow keys. Manual subgrid position adjustment is performed by moving each grid individually with the mouse or keyboard arrow keys while in *Move* mode, which is activated from the *Move* command of the *Grid* menu.

Grid Processing

Select the segmentation method and input all required parameters. When setting minimum and maximum spot sizes for the Otsu method the rule of thumb is to set the range of spot sizes as close as possible to the visible range on the slide. However, range minimization can be potentially dangerous, as it may cause instability in the Otsu method iteration procedure. The actual spot size range on the slide can be measured interactively with the help of the mouse pointer on the *Analysis Page*. For the *Histogram* method, set the spot size slightly higher than what is expected. To start grid processing press the *Process All* button on the *Gridding and Processing* pane. Spot detection, segmentation, and local background correction steps are all performed during processing.

Grid Alignment Examination

Checking the *Contour* check box on the *Analysis Page* will show the spot boundaries. After the processing is completed, the user is able to see the contours of the spots colored in green for bad spots or magenta for good spots. If green contours appear in cells with spots that otherwise look good,

this may indicate misalignment or wrong settings for the segmentation method used. Go to the *QC View Page* for subgrid alignment checking. Visually compare the relative size of the green area on different subgrids in the array. They should be approximately the same unless the array was designed intentionally with some special subgrids (e.g., all replicates are printed in one subgrid). Any subgrid with a disproportionately large green area should be checked for misalignment. If the alignment is shown to be correct, the higher number of bad spots in this subgrid can be considered indicative of the low expression of genes printed in this subgrid.

Postprocess Data Tuning

The default QC filter operation is the last step of processing. The user can change QC filter settings in the *Post-processing* dialog without having to reprocess the slide. Switch to the *Post-processing* dialog to enter new QC filter settings. Background correction can be disabled or enabled by using the check box of this dialog. The QC filter can be set more or less stringent by changing the cutoff threshold defined by the signal-to-noise ratio (see earlier discussion). By varying parameters α and β of Eq. (1) the user can set threshold T at the level where a reasonable distinction between weak spots and strong spots, produced by noise, is visible on the *Analysis page*. After making the desired changes, press the button *Update Changes*. The result can be observed in the *Analysis Page*, on the spreadsheet data table of the *Data page* and on the *RI plot page*.

Annotation Import

A variety of annotation file formats (.ann, .dat, .gal) can be loaded by Spotfinder for the purposes of displaying annotation alongside expression data and to map to an output data file. At first the user needs to construct or load the grid in Spotfinder to ensure correct mapping. To load an annotation file, go to the main menu bar and select *Data* → *Load Annotation File*; the selected file will be loaded and checked for the correct total number of rows (spots). Once loaded, the annotation can be viewed on the *Data* page or on the status bar at the bottom of the Spotfinder GUI for spots selected on the *Analysis* page. To map annotation to the data set in Spotfinder and in any future .mev files, go to the main menu and select *Data* → *Set UID from annotation* or *Data* → *Set DBID from annotation.* This will change default UIDs in the *MEV data* tab of the *Data* page and create an additional DBID column with DBIDs from the annotation file. The new mapping will be stored in any .mev files generated by Spotfinder in the next step.

Report Output Data

To save expression data in a .mev file, go to the menu D*ata → Save Data to MEV file* and enter a file name in the *Save File* dialog. This creates a tab-delimited data file that is used by all software tools in the TM4 suite. To save grid information in a binary, platform-independent SFG file (Spotfinder Grid file), go to the menu *Grid → Save Grid in File* and input the grid file name in the dialog. The grid and all raw data can also be saved in a platform-independent, binary data SFD file (Spotfinder Data file). To create an SFD file, go to the menu *Data → Save Data to SFD file*; the file save dialog will ask for a file name to save as. The SFD file stores all processed raw data and spot contour vectors needed for graphical representation of the contours on the Spotfinder *Analysis page* in a later session. The SFD file can be used later by any user who needs to view the results with the RI plot and spot contours; the postprocessing operations can be applied to SFD data to generate a new .mev file for the same data set but with different QC filter settings.

MIDAS

Microarray analysis is a comparative analysis. In a two-color experiment, cDNA or mRNA abundances are compared between two samples. During a microarray experiment, the different samples are dyed with Cy3 (green) and Cy5 (red) fluorophores and are cohybridized to a glass slide. After scanning the slide and performing an image-processing procedure, the intensities for each spot, for both green and red channels, are recorded.

An underlying assumption in microarray analysis is that differences between the two intensities for each spot faithfully reflect the cDNA or mRNA abundance differences between the two samples. This is the basis for investigating cDNA or mRNA abundance differences in tens of thousands of spots in a microarray slide simultaneously by clustering using pattern recognition or other data mining techniques. This assumption, however, is compromised by all kinds of errors or biases introduced during the experiment and image processing. Predicting the bias, adjusting the raw intensities for each spot accordingly so that they better reflect the true picture about the cDNA/mRNA abundances is a crucial data preprocessing step before downstream analysis can be carried out. It is also important to remove those spots within an array with ''unacceptable'' intensities, as defined by varying criteria. These data preprocessing steps are called *Normalization and Filtering*. Other normalization and filtering methods are found elsewhere (Ayroles and Gibson, 2006; Gollub and Sherlock, 2006; Reimers and Carey, 2006).

Midas (also called MIDAS) is the data normalization and filtering tool in the TM4 microarray data analysis software suite. It contains a number of normalization and filtering modules, as well as significant gene identification

modules. The software provides a user-friendly graphical scripting feature, which allows these modules to be pipelined together to form an analysis workflow. Midas has a strong graphing feature for users to investigate the analysis results. A graphical PDF analysis report can also be generated by user's request.

Building a Pipeline

A typical Midas data analysis pipeline is composed of three steps: (1) reading raw data input files, (2) defining the analysis workflow by queuing one or more analysis modules, and (3) writing processed data output files.

After an analysis pipeline is defined in the Midas Workflow window, parameters should be set in the parameter sheets associated with each module defined in the pipeline (Fig. 6). The analysis pipeline and the parameters can then be saved into a Midas project file (.prj) under a Midas project folder. The Midas project folder will be the location for all analysis results, output data files, analysis plots, reports, and error messages, if there are any. The analysis example at the end of this chapter describes

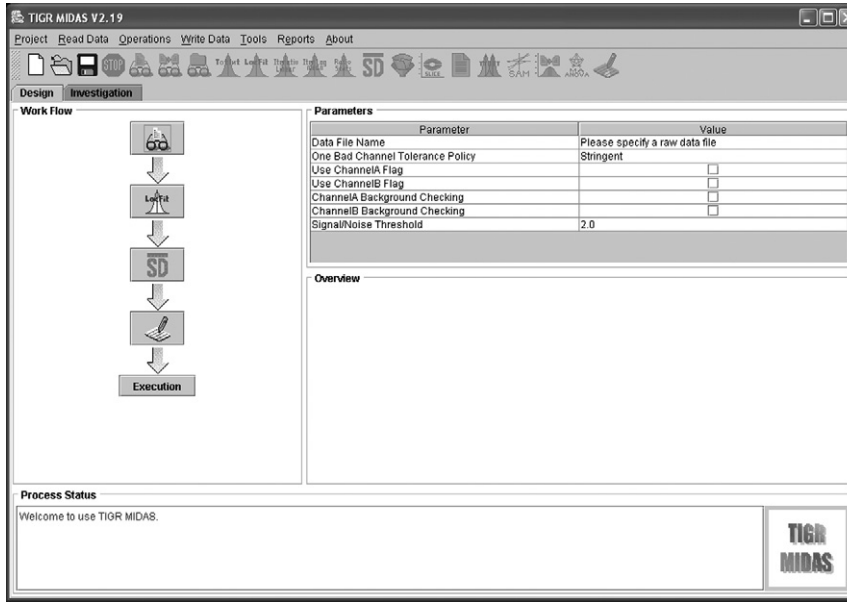

FIG. 6. Midas graphical user interface.

the steps involved in building an analysis pipeline using several popular modules.

Normalization Modules

Midas includes a number of normalization algorithms. These can be used in sequence with each other or with other modules. Choosing appropriate normalization and filtering methods can be one of the more challenging aspects of using microarrays. Applying inappropriate methods to experimental data might lead to information loss or information distortion. An example is that removing raw data by some filtering methods might have negative consequences for some downstream gradient correction methods.

A common approach to normalization is global normalization. In this approach, averages of the overall expression levels for all genes within an array across different arrays are set to be equal. This follows from the assumption that while genes can be expressed differentially, the amount of transcription is essentially similar across samples. Furthermore, it is also common to set the averaged overall expression levels for each array to be zero. This follows from the assumption that within each array, overexpressed genes and underexpressed genes are roughly balanced. Global normalization methods are mostly useful for normalizing hybridization arrays for gene profiling or similar samples comparison purposes. They might not be valid normalization approaches when the compared samples are too different across arrays or when using comparative genomic hybridization arrays.

Table I provides some general guidance for applying the right normalization and filtering methods. Keep in mind that the correction of a bias or error assumes that the experimental design and array samples do not undermine the assumptions of the applied algorithm.

Total Intensity Normalization

Total intensity normalization (Quackenbush, 2002) assumes the summed intensities for each of the two channels, channel A and channel B, for all spots within an array should be equal. If there is any observed difference, it is caused by some dye-specific systematic bias and thus should be adjusted. The algorithm calculates the factor between the two summed intensities ΣIA and ΣIB and scales intensity A (IA) or intensity B (IB) of each spot so that the goal of equal summed intensities in the two channels is achieved.

Lowess Normalization

Lowess normalization (Quackenbush, 2002; Yang *et al.*, 2002a,c) assumes that spots having different overall intensities [measured by $\log_{10}(IA \cdot IB)$] should have different systematic bias added to their expression levels

TABLE I
GUIDELINES FOR SELECTING NORMALIZATION AND FILTERING METHODS

Issues to be addressed	Applicable methods
Averaged overall expression within an array not zero observed unexpectedly	Total intensity; iterative log-mean centering; ratio statistics; Lowess
Print tip-dependent bias observed	Standard deviation regularization
Intensity-dependent bias observed	Lowess
Nonlinear correlations observed unexpectedly between the two channel intensities (logarithm transformed)	Iterative linear regression
Inconsistent expressions between dye-swapped experiments observed	Flip-dye consistency normalization and filtering
Selecting significantly expressed genes from a single array	Slice analysis
Selecting significantly expressed genes when replicated arrays are available	Statistical methods such as "*t* test" and "SAM"
Noisy raw expressions	Background filtering; low-intensity filtering

(measured by $\log_2{}^{IB}/_{IA}$). Thus the goal of this normalization method is to extract the intensity-dependent systematic bias for each spot and use it to adjust the raw *IA* or *IB* for each spot. The Lowess algorithm estimates the adjustment factor for the $\log_2{}^{IB}/_{IA}$ value of a spot by finding those spots in the neighborhood of this spot, based on their intensities, and computing their commonly shared bias by a maximum likelihood technique, which applies a locally weighted model to spots' expression data in each neighborhood. A related algorithm, Loess, differs from Lowess because of the model used in the regression: Lowess uses a linear polynomial, whereas Loess uses a quadratic polynomial. The "neighborhood" is defined by a parameter called the *smoothing parameter*, which defines the percentage of all spots within a physical scope. The physical scope can be either *block*, meaning all spots printed by the same print tip, or *global*, meaning all spots on the array.

Iterative Log-Mean Centering Normalization

Iterative log-mean centering normalization (Quackenbush, 2002) assumes that the majority of the spots within an array show a balanced distribution of expression levels (measured by $\log_2{}^{IB}/_{IA}$). For these spots, their $\log_2{}^{IB}/_{IA}$ values should have a mean value of 0. Aside from this majority, a few outlier spots, those with very high or very low $\log_2{}^{IB}/_{IA}$

values, contribute significantly to the calculation of the overall $\log_2{}^{IB}/_{IA}$ mean. This algorithm uses an iterative procedure to remove the outliers and calculate the $\log_2{}^{IB}/_{IA}$ means for the outlier-removed spots until the means converge. The algorithm then scales the intensities of each spot by this converged mean value.

Iterative Linear Regression Normalization

Iterative linear regression normalization (Finkelstein *et al.*, 2000) assumes the correlation between $\log_{10}IB$ values and $\log_{10}IA$ values for all spots within a physical scope on the array displays a $y = x$ linear relationship. The physical scope can be either *block*, meaning all spots printed by the same print tip, or *global*, meaning all spots within the array. The algorithm calculates the slope and intercept between the $\log_{10}IB$ values and $\log_{10}IA$ values for spots within the specified physical scope, iteratively. During each iteration, the outlier spots, which are defined as those having $\log_{10}IB$ or $\log_{10}IA$ residuals greater than a user-defined threshold range, are removed. The final slope and intercept are achieved when the calculated correlation coefficients converge. The final slope and intercept are then used to adjust the IA and IB of each spot so that the $\log_{10}IA$ and $\log_{10}IB$ distribution displays such a linear relationship.

Standard Deviation Regularization

Standard deviation regularization (Yang *et al.*, 2002c) assumes that variances of the expression levels of the spots (measured by $\log_2{}^{IB}/_{IA}$) within different physical scopes should be the same. The physical scope can be either *block*, meaning all spots printed by the same print tip, or *global*, meaning all spots within the array. Based on this assumption, the IA and IB values of each spot are adjusted so that the same standard deviation, and thus the variance, of the $\log_2{}^{IB}/_{IA}$ values of the spots prevails among the specified physical scope. For example, the variances for all blocks on an array could be set equal to each other by this method.

Ratio Statistics Normalization

Ratio statistics normalization (Chen *et al.*, 1997) assumes that there exists a sample-independent single-mode ${}^{IB}/_{IA}$ distribution function with mean μ and standard deviation σ. The mean can be estimated through an iterative process described in the reference paper. The algorithm also assumes that the population of ${}^{IB}/_{IA}$ values for all spots in an array should approximately demonstrate a mean value of 1. The calculated mean μ can then be used to normalize the IA and IB of each spot so that the ${}^{IB}/_{IA}$ population mean becomes 1.

Flip-Dye Consistency Normalization and Filtering (Quackenbush, 2002)

In a pair of flip-dye arrays s1 and s2, the $\log_2{}^{IB}/_{IA}$ for any spot in s1 is expected to have an expression value of $-log_2{}^{IB}/_{IA}$ for the corresponding spot in s2 due to the fact that the two spots are dye-swapped replicates of each other. Therefore, if the $\log_2{}^{IB}/_{IA}$ values for all spots in s1 versus $\log_2{}^{IB}/_{IA}$ values for all spots in s2 are studied for their correlation, a linear relationship is expected. The flip-dye consistency normalization algorithm checks the consistencies for each spot's expression values between s1 and s2 by calculating the $c = \log_2{}^{IB_1}/_{IA_1} - \log_2{}^{IA_2}/_{IB_2}$ histogram, where IA_1 and IB_1 denote the two-channel intensities for a spot in s1 and IA_2 and IB_2 denote the two-channel intensities of the corresponding spot in s2. By assuming this histogram follows a normal distribution with a mean of 0, those spots with c values that fall beyond a user-defined consistency range are removed. These are considered to be inconsistent data between the flip-dye replicates. The rest of the spots are output as consistent spots. For each of these consistent spots, the $\log_2{}^{IB}/_{IA}$ value is presented as the geometric mean of $\log_2{}^{IB_1}/_{IA_1}$ value and $\log_2{}^{IB_2}/_{IA_2}$ value.

Filtering Modules

Filtering modules reduce the size of the data set by removing elements that do not meet certain user-defined criteria. These modules can be added to the workflow before or after normalization modules. Filtering modules applied before normalization remove the ''bad'' or unreliable elements defined by certain quality control criteria to allow only ''cleaner'' data to be used as input for normalization procedures. Such modules include flag filtering, background filtering, and low-intensity filtering. In contrast, filtering modules applied after normalization remove elements that may not be important or interesting, given the research goals. These ''postnormalization'' filtering methods include in-slide replicate analysis and cross file trim, as well as the significant gene identification modules described in the following sections.

Flag Filtering

During the image processing stage, some spots might be flagged as ''bad'' due to a variety of reasons, such as saturation. The flag-filtering feature allows these flags be read before data are processed. Flagged spots will be excluded from any downstream processes.

Background Filtering

During the image processing stage, the user may request that background intensities be calculated along with the signal intensities IA and IB. These background intensities can be used to calculate the signal-to-noise

ratios for each spot. The background filtering feature excludes those spots with signal-to-noise ratios below a user-defined threshold from the downstream processes.

Low-Intensity Filtering

The low-intensity filtering feature excludes those spots with channel A intensity *IA* or channel B intensity *IB* lower than user-defined thresholds from the downstream processes.

In-Slide Replicate Analysis

In-slide replicates are technically replicated spots printed within an array. These replicated spots are theoretically expected to demonstrate the same $\log_2{}^{IB}/_{IA}$ expression values. Observed variances among the replicates are caused by random errors. In-slide replicate analysis combines the replicated spots, which are defined as those spots in an array sharing the same annotation identifier, for example, feature name, into a single output data spot. The expression value $\log_2{}^{IB}/_{IA}$ of this combined spot is equal to the geometric mean of the $\log_2{}^{IB}/_{IA}$ values of the replicates that were combined.

Cross File Trim (Percentage Cutoff Trim)

When multiple data files with the same number of spots are analyzed together, it is often desirable to check the consistency of the expression value of a spot across all the files. This occurs after each file is normalized and filtered, but before processed data are written to output files. The consistency of a spot is calculated as a percentage of the number of files showing the spot as being ''unfiltered'' divided by the total number of files used.

Cross file trim allows the consistency percentage of each spot to be compared with a preset consistency threshold percentage. Spots that do not pass the threshold comparison are filtered in the output files by setting their *IA* and *IB* intensities to 0. This filtering method is also referred to as ''percentage cutoff'' trimming in MeV.

Significant Gene Identification Modules

Slice Analysis

It is well known that variances of expression levels of spots (measured by $\log_2{}^{IB}/_{IA}$) vary as the intensities change. This fact makes a simple ''fold change'' criteria for identifying differentially expressed gene within an array less than ideal.

A modified approach is to study how the expression levels of the spots are distributed across the array as the overall intensity of the spots [measured by $\log_{10}(IA \cdot IB)$] varies. In this approach, each spot is associated with a group of spots, called a ''slice.'' A slice consists of those spots that have similar overall intensities as the query spot. The mean and standard deviation of the expression values in each slice are calculated. A differential expression z score for a spot can then be defined as the difference between the expression level of the spot and the mean expression value for the slice that the spot belongs to, divided by the standard deviation of the slice that the spot belongs to. A spot is identified as ''significantly expressed'' if its differential expression z score is greater than a user-defined threshold.

Slice analysis (Yang *et al.*, 2002a), a method to identify significantly expressed genes, classifies genes within a single array by their intensity-dependent differential expression z scores as described earlier.

One-Class t *Test and One-Class SAM*

When multiple arrays representing technical or biological replicates of the same genes are available, significantly expressed genes can be identified by applying scientific statistical analysis. Two such methods are implemented in Midas: one-class t test and one-class SAM (Chu *et al.*, 2002; Tusher *et al.*, 2001). These methods can also be found in MeV. Users who are interested in applying the one-class t test and one-class SAM are encouraged to read the corresponding sections in the MeV description and the sample analysis walk-through that follow.

Graphs and Reports

A variety of analysis graphs are plotted and saved during the execution of a Midas analysis pipeline. These graphs, such as the R-I plot (Fig. 7, left) and flip-dye diagnostic plot (Fig. 7, right), are saved within the Midas project folder and can be studied under the ''Investigation'' tabbed panel. Graphs of interest can be exported to the graphical PDF reports (Fig. 8) by the user's request.

MeV

After spot scanning and normalization comes the data analysis step that is usually of most interest to microarray practitioners, namely mining data to look for biologically significant patterns of gene expression (Ayroles and Gibson, 2006; Downey, 2006; Neal and Westwood, 2006; Reimers and Carey, 2006; Royce *et al.*, 2006). The MeV (MultiExperiment Viewer) software incorporates an extensive array of clustering, statistical, and visualization

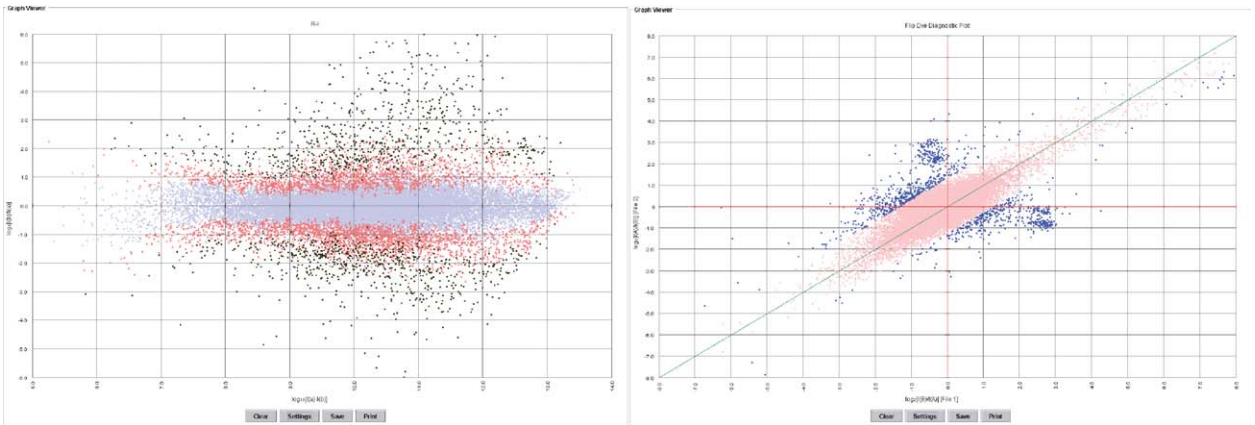

FIG. 7. (Left) An R-I plot showing significantly expressed genes classification results after slice analysis is applied. The outlier genes, which have their intensity-dependent, differential expression z score greater than twofold of standard deviation, are colored red; genes z scores below onefold of standard deviation are colored blue; the remainder are colored green. (Right) A flip-dye diagnostic plot showing consistencies about expression values between a flip-dye pair. The solid diagonal line represents the theoretical perfect consistency relationship. Genes in blue are considered to be consistent between the flip dye using twofolds of standard deviation cut as the consistency criteria. The other genes are colored red.

FIG. 8. Midas PDF analysis report.

tools that can be used to analyze preprocessed microarray data. An intuitive and feature-rich interface makes it easy to use the software, eliminating the need for a programming or scripting language. In addition to the .mev file format used by the TM4 suite, MeV (also known as TMeV) works with file formats generated by a number of other platforms or analysis programs (Affymetrix MAS 5.0 output, RMA output, Agilent or Genepix scanner files, and a more generic tab-delimited text file format containing log ratios from multiple samples). Thus, MeV is a versatile end-stage analysis tool that can be used at the last stage of a TM4 pipeline or as a stand-alone program to analyze data that have been processed with other analysis tools.

Data Representations and Distance Metrics

In MeV, the expression level corresponding to each spot on a slide is represented as an *expression element* (Fig. 9). An expression element is typically a $\log_2$ transformation of an expression ratio in the case of two-color arrays, where a ratio is calculated by dividing the fluorescence intensity from one channel by the fluorescence intensity from the other channel for a given spot on a slide. In the case of single-channel arrays (such as Affymetrix chips), an expression element is the normalized single intensity value for a probe set. Hereafter, for convenience we use the term *gene* to refer a spot or

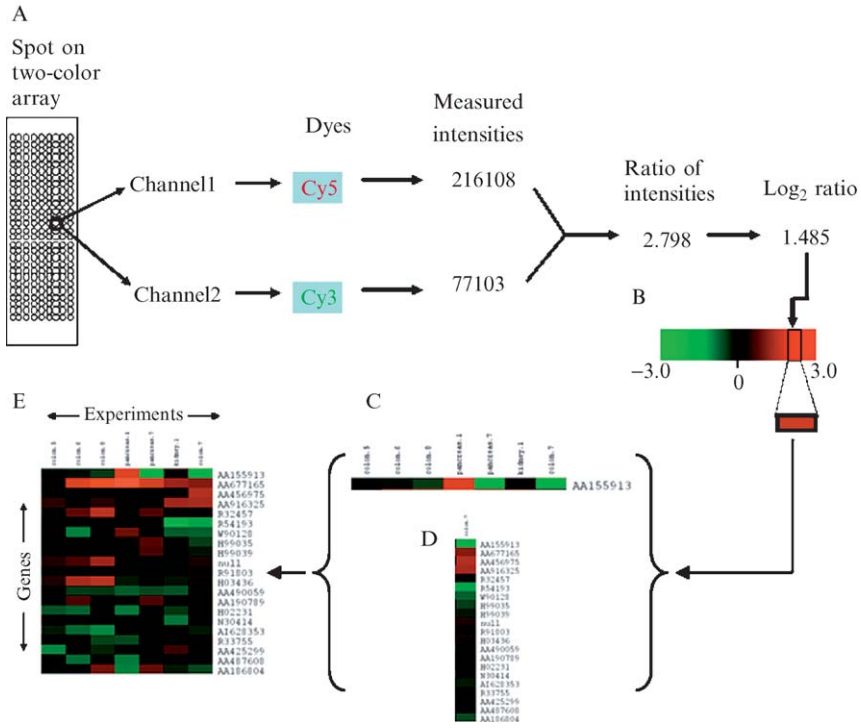

FIG. 9. Data representations in MeV. (A) Numerical and (B) false-color representations of an expression element, (C) a gene expression vector, (D) an experiment expression vector, and (E) an expression matrix.

a probe set, even though the DNA sequence corresponding to that spot or probe set may not span the entire length of a gene in a biological sense. Because an *experiment* corresponds to a *slide* on which a given *hybridization* was carried out, these three terms are often used interchangeably.

An *expression vector* (Fig. 9C and D) is a set of expression elements for a given gene or experiment. For a gene expression vector, each element comes from a separate experiment in which the intensity of that spot was measured. An experiment expression vector contains the expression elements of a set of genes in a given experiment.

An *expression matrix* (Fig. 9E) in MeV is a two-dimensional array of expression elements from a set of genes over multiple experiments. By convention, each row is an expression vector from a given gene, and each column corresponds to an expression vector from a given experiment. The expression matrix in MeV (and generally in microarray data representations)

is shown in a false-color view on a red–green scale by default, with green representing low expression and red representing high expression. These colors can be customized.

Another important concept is that of *distance*. A distance metric is a numerical estimate of how similar the expression patterns of two expression vectors are. The smaller the magnitude of distance, the greater the similarity of the two patterns. Many algorithms in MeV use distance metrics to put expression vectors in clusters that contain vectors of similar expression. There are many types of distance metrics, some of which use very different criteria from one another to estimate similarity. Thus, two vectors might be judged very similar by one distance metric and quite unlike one another by another metric. It is important to select a distance metric that is appropriate to the underlying question being asked. For instance, the Pearson correlation distance is appropriate when one is interested in finding genes showing similar patterns of expression over a set of experiments, such as a time course, regardless of the magnitude of expression. However, if the primary interest is in grouping together genes that have similar levels of expression (over- *vs*. underexpressed), then the Euclidean distance might be a better choice. MeV offers 11 distance metrics, any of which can be applied to the distance-based algorithms in the package.

Data Mining in MeV: A Brief Algorithm Overview

One should be aware that there is often not one ''correct'' analysis approach to any particular data set. What is important is to know what an algorithm is doing, how it makes decisions during cluster creation, how input parameters affect results, and what features of data may be revealed by an analysis. A powerful feature of MeV is the ability to overlay results obtained from multiple methods to reach a consensus or to reveal different aspects of data. At times finding an approach requires some level of trial and error to find methods and suitable parameters.

The mechanics of executing an analysis algorithm in MeV are quite simple. Once data are loaded, the analysis is initiated by selecting the corresponding button in the toolbar or the menu item from the ''algorithms'' menu. All algorithms initially open one or more dialog boxes that are used to collect input parameters. The lower left corner of each dialog contains an information button (Fig. 10), which opens a help window with information about the input parameters.

Some algorithms require rather large amounts of computer memory space and it is generally recommended to have 512 MB to 1 GB of RAM. In addition to memory requirements, several algorithms are computationally intensive and can take several minutes or, in some cases, hours to complete.

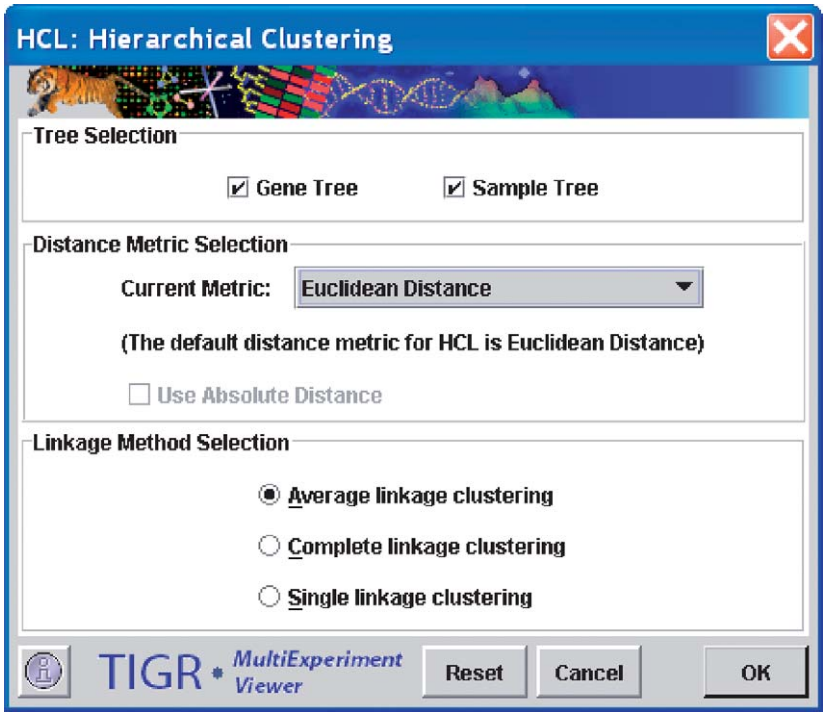

FIG. 10. HCL algorithm parameter selection dialog. The lower left of each algorithm dialog contains the parameter information button.

All algorithms present progress logs or progress bars to provide a status report during algorithm execution.

MeV currently provides 24 analysis techniques. In terms of the objectives they attempt to accomplish, these algorithms can be classified into three broad categories: exploratory techniques, hypothesis testing techniques, and classification techniques. Exploratory techniques look for broad patterns in the data set; examples of algorithms in this category include hierarchical clustering (HCL) and principal components analysis (PCA). Hypothesis testing techniques use information about the experimental design to identify a subset of genes that show statistically significant differences in patterns of expression across groups of samples; examples of such techniques include TTEST, SAM, and ANOVA. Classification techniques use information about the known class membership of some genes or samples to assign the remaining genes or samples into these classes; algorithms such as SVM and KNNC fall into this category.

Alternatively, these analysis techniques can be categorized based on the nature of their underlying algorithms. These broad categories are agglomerative methods, divisive methods, methods to assess confidence in clustering results, neural network approaches, statistical tools, classification algorithms, data visualizations and component analysis, and biological theme discovery. We use these categories based on algorithm heuristic in describing some of the following algorithms. The cited references, manual, and training slides available at the TM4 web site provide greater detail about these algorithms.

Agglomerative Methods

Agglomerative methods start by considering each expression vector as a distinct and independent object. Vectors are fused into clusters based on similarity, which is determined based on the selected distance metric. A cluster so formed from two elements is then considered as a single object, a cluster of size two, rather than as two distinct elements. In subsequent rounds, objects are fused to form bigger clusters based on intercluster similarity using the same distance metric as described earlier to define intercluster distance. The method continues joining the most similar objects at each stage until all objects are assigned to one large cluster.

HIERARCHICAL CLUSTERING (HCL). Hierarchical clustering (Eisen *et al.*, 1998; Weinstein *et al.*, 1997) is likely the most widely used agglomerative method for preliminary data exploration. HCL constructs a binary tree by successively grouping the genes or samples based on similarity. A set of vectors falling under a node in the tree tend to be more similar to each other than to vectors in other sections of the tree. By observing how gene or sample expression patterns are arranged in the tree, one can select and focus on subtrees that contain consistent patterns of interest.

The HCL tree viewer in MeV permits one to dynamically select a subtree to assign to a cluster or to slice the tree into any number of distinct clusters based on a similarity value cutoff (Fig. 11). Hierarchical clustering is a popular analysis option for getting an overview of patterns in the data set.

Divisive Clustering Methods

Divisive clustering methods begin with all vectors in one cluster, which is then partitioned into distinct clusters. The objective is to create clusters such that all elements within a cluster are similar to one another, and each cluster is dissimilar to the others. No relationship is specified among the clusters.

K-MEANS/K-MEDIANS CLUSTERING (KMC). K-means clustering (Soukas *et al.*, 2000) is a divisive technique that divides the genes or samples into a set of k clusters. Initially, the vectors are assigned randomly to a predefined

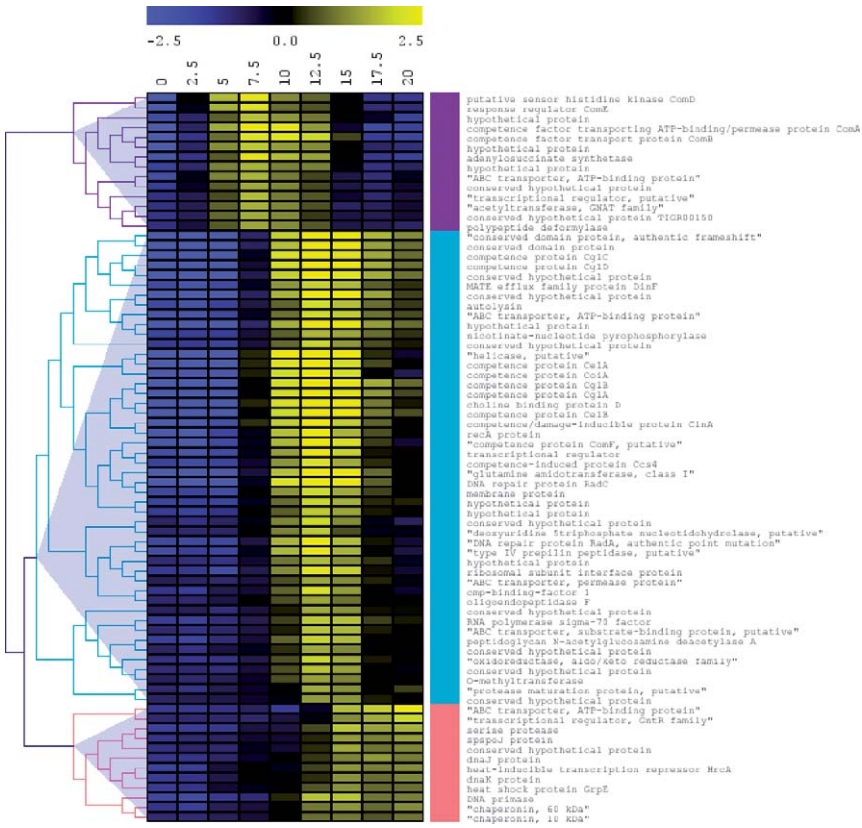

FIG. 11. Hierarchical cluster of time course data using the Pearson correlation distance metric. Prominent patterns of expression have been selected as clusters.

number of clusters. The assignment is iteratively refined by shuffling vectors among clusters and updating the mean or median profile of each cluster until each vector is assigned to the cluster whose mean or median it is closest to. This method is useful when one has a reason to assume that data should partition into a specified number of clusters. During clustering analysis, vectors are sometimes divided into too many clusters, such that there are two or more clusters that have mean patterns that are similar. This suggests that those clusters should be merged. In other cases where too few clusters have been created, the clusters will tend to be large and contain quite diverse and variable patterns in each cluster. The number of clusters can be chosen by trial and error to hone in on a partitioning that appears to appropriately split data into distinct clusters.

CLUSTER AFFINITY SEARCH TECHNIQUE (CAST). The cluster affinity search technique (Ben-Dor *et al.*, 1999) partitions data into clusters that contain members guaranteed to have a minimum specified "affinity" to other members of the cluster. The affinity of a particular gene is related to the total similarity of that gene to all other genes in the cluster being created. A nice feature of this algorithm and some others like it is that the number of clusters to create is not predefined. Clusters are created until all items are assigned to clusters of the largest size possible while ensuring that all genes within a cluster have some minimal affinity for the cluster.

GENE SHAVING. Gene shaving (Hastie *et al.*, 2000) is a divisive clustering technique that partitions the genes into clusters such that genes within a cluster have low gene-to-gene variability, while having high variance across samples. Thus, a cluster of genes created by this algorithm will tend to have similar expression profiles that tend to vary substantially across samples. One important difference from many other divisive clustering techniques is that clusters from gene shaving are not always mutually exclusive so that a given gene may appear in more than one cluster. The procedure attempts to make successive clusters almost uncorrelated with previously created clusters so that if a gene appears in more than one cluster, each such cluster might highlight different aspects of the variability of that gene.

QTCLUST. QTClust (Heyer *et al.*, 1999), like CAST, is a clustering technique in which the number of clusters is not specified by the user, but is determined by two inputs: the maximum possible distance between two genes in a cluster (called the cluster diameter) and the minimum number of genes that a cluster must contain (the cluster size). For calculating cluster diameter, gene-to-gene distance is computed using a jackknifing procedure in which each sample is left out in turn. This reduces bias that might be introduced by individual outlier samples. Clusters are created in sequence, and the genes that are unassigned after the creation of a cluster are subjected to successive rounds of clustering until no more clusters can be created that satisfy both the cluster diameter and the cluster size thresholds. At the start of each round of clustering, all unassigned genes serve as potential seeds for a new cluster. The largest cluster created from all seeds in a given round is retained, and the procedure is repeated on the remaining unassigned genes. Allowing each eligible gene to serve as a potential seed for further clustering prevents the algorithm from being biased by the order in which data are presented to it.

Assessing Confidence in Clustering Results: Support Trees, Figures of Merit, and K-Means Support

Clustering algorithms are guaranteed to organize data into clusters, even when no clear patterns exist. It is therefore helpful to assign measures of confidence on the clustering results to assess whether the clustering is

meaningful. This is done by repeating the clustering analysis many times with the same parameters on the same data set or a resampled data set or by gradually changing the magnitude of an input parameter and then comparing the results across all runs. MeV offers three methods to assess confidence in clustering results.

HCL SUPPORT TREES (ST). The ST module in MeV builds a hierarchical tree by the same algorithm employed by the HCL module of MeV. The difference here is that after finding the initial tree, the expression matrix is resampled with replacement many times to produce bootstrapped expression matrices. An HCL tree is built on each of these bootstrapped matrices and compared to the original tree. Each node in the original tree is assigned a value between 0 and 100, indicating the percentage of times over all resampling trials that a node containing those elements occurred in a tree obtained from a resampled matrix. These bootstrap confidence values are displayed on the tree as colors or as numerical values. Higher node values indicate that the vectors under that node clustered together frequently regardless of resampling, which indicates that the cluster represented by that node was not unduly influenced by a small subset of data.

Other algorithms in this category are figures of merit (Yeung *et al.*, 2001), which iteratively step through different values of k searching for an optimal value based on a comparison of within-cluster and between-cluster distances, and K-means support, which iterates K-means at a fixed value of k searching for stable clusters.

Machine Learning Methods

Machine learning-based clustering approaches are suitable for partitioning large data sets that contain a lot of random noise in addition to distinct expression patterns of interest. This means that these approaches are very applicable to microarray data. These approaches represent the clusters being created as a set of nodes connected as a network, where each node has a representative expression profile that is trained by data to better conform to a subset of data. As each vector is presented to the network, the node or nodes most similar to that vector adapt to become even more similar to the presented vector. By presenting the vectors to the system many times, the nodes conform to represent clusters that are inherent in data. Once the adaptation is complete, each vector is placed into a cluster related to the node with the most similar representative expression profile.

SELF-ORGANIZING MAPS. Self-organizing maps (Kohonen, 1982; Tamayo *et al.*, 1999) in MeV are a very efficient neural network implementation that permits millions of training/adaptation cycles to be run in a relatively short time. The algorithm requires an initial topology of the network, which means that an estimate of the number of expected clusters

must be provided. Similar to KMC, the suitability of this estimate can be assessed based on the results of multiple runs.

SELF-ORGANIZING TREE ALGORITHM (SOTA). The self-organizing tree algorithm (Dopazo and Carazo, 1997; Herrero *et al.*, 2001) is a hybrid approach that bridges divisive and neural network approaches to produce a binary tree structure where each terminal node or leaf in the tree is a cluster. Starting with all genes in a single node or cell, the cell then divides and partitions the vectors optimally between the two offspring cells. On each division the most variable cell is split until a predetermined number of divisions or a cluster variability threshold is met. In addition to the described benefits of the machine learning methods, SOTA does not require a predetermined cluster count.

Statistical Tools for Extracting Significant Gene Lists

The basic clustering methods described previously focus on finding correlated patterns of gene expression within the data set. This is often useful for time course data or for general data mining for prevalent patterns. In the case where the experimental design contains biological or technical replicates and the samples are partitioned into discrete sets that represent experimental conditions, statistical tests can be applied to find genes that show differential expression under the conditions being studied. In addition to extracting genes of interest, each gene will have a corresponding *p* value describing the likelihood that the observed finding was due to chance. Microarray experiments are being designed increasingly to take advantage of statistical tools.

TTEST *(TTEST)*. MeV provides three *t* test (Dudoit *et al.*, 2000; Korn, *et al.*, 2001, 2004; Pan, 2002; Welch, 1947; Zar, 1999) designs: one sample, between subjects, and paired. The one-sample *t* test is useful for identifying genes that show consistent over- or underexpression across a series of biological or technical replicates. The between-subjects *t* test is useful for finding genes that are significantly differentially expressed between two independent groups of samples (e.g., two strains of mice). The paired *t* test can be used to find genes showing differential expression between two conditions assayed on the same samples (such as before and after administering a drug to a group of individuals).

ONE-WAY AND TWO-FACTOR ANOVA. Two types of ANOVA designs are offered: One-way (Zar, 1999), for comparison of three or more independent groups, and a completely randomized two-factor design (Keppel and Zedeck, 1989; Manly, 1997; Zar, 1999) for analyzing variation across two conditions (such as sex and strain). See Ayroles and Gibson (2006) for more about ANOVA. The *t* test and ANOVA modules offer error rate correction

options (such as Bonferroni corrections) for multiple testing (Dudoit *et al.*, 2003), as well as false discovery rate (FDR) computations (see later).

SIGNIFICANCE ANALYSIS OF MICROARRAYS. A false discovery rate can also be computed using the popular SAM module (Tusher *et al.*, 2001; implemented as in Chu *et al.*, 2002), which includes options for five experimental designs, four of which are analogous to the *t* test and one-way ANOVA designs, while the fifth is suitable for survival data. FDR computations are often a desirable alternative to conventional statistical tests (such as *t* tests and ANOVA) in microarray data analysis. The simultaneous analysis of thousands of genes leads to highly inflated error rates for individual genes when doing conventional statistical tests. FDR analysis can help circumvent this problem by allowing the identification of a list of potentially significant genes while still keeping overall error rates low.

TEMPLATE MATCHING. Template matching (Pavlidis and Noble, 2001) is useful for finding patterns of expression that are similar to a user-specified pattern (as judged by the magnitude and sign of the correlation coefficient between the patterns of interest or the *p* value of this coefficient).

Classification Algorithms/Supervised Clustering Approaches

Supervised methods use existing biological information about specific genes or samples (the ''training set'') that are functionally related to ''guide'' the clustering algorithm. The existing information is the presumed class membership of each vector in the training set. This information is used to classify other vectors (the unknowns) based on how similar their expression patterns are to members of the training set.

SUPPORT VECTOR MACHINE (SVM). A support vector machine (Brown *et al.*, 2000) is a supervised classification method that bisects data into two distinct groups: in class and out of class. SVM uses a subset of data that is known to fall into the class of interest as examples of the class.

K-NEAREST NEIGHBOR CLASSIFIER (KNNC). KNNC (Theilhaber *et al.*, 2002) partitions data into *k* distinct classes, where *k* is a supplied number of partitions. Like SVM, KNNC uses a subset of data to use as examples of each class being partitioned.

Data Visualizations and Component Analysis

This broad category includes algorithms that attempt to simplify the interpretation of the main features of data by presenting a view of data that provides a means to consider high level structure of data.

PRINCIPAL COMPONENTS ANALYSIS. PCA (Raychaudhuri *et al.*, 2000; Downey, 2006) extracts the features in the data set that are most representative and best ''explain'' the nature of the variation in data. These features, known as principal components, are used to map data into 2D and

3D visualizations that can sometimes provide an intuitive view of the main aspects of variation in the data set. A related method, correspondence analysis (Fellenberg *et al.*, 2001; Culhane *et al.*, 2002), maps both genes and samples onto the same set of axes, revealing associations between genes and experiments.

GENE EXPRESSION TERRAIN MAPS (TRN). Gene expression terrain maps (Kim *et al.*, 2001) build a 3D topological terrain view where gene or sample clusters appear as peaks in the terrain. The algorithm first maps data into a two-dimensional grid such that elements that are similar are close together. The third dimension giving rise to the peaks is related to the density of the elements under the peak. This means that if many elements are similar to each other, they will appear as a tall sharp peak over a small region of the map. By using appropriate metrics, one can use TRN to get an overview of the data set and can estimate a rough idea about the number of major clusters in data.

GENE DISTANCE MATRIX. The gene distance matrix displays a 2D heat map representation of the similarity matrix. This matrix displays the distance (inverse of similarity) between any two elements (genes or samples) in the data set. When the matrix is sorted by cluster membership based on a prior clustering result, the matrix can qualitatively indicate how distinct two clusters are in terms of the expression patterns of the member. When used to assess sample distances, where the matrix is relatively small, one can interrogate the actual similarity between any pair of expression profiles.

Biological Theme Discovery

After obtaining a list of genes, an important task is to determine whether the genes have a common or connected biological role within the system being studied (Whetzel *et al.*, 2006).

EXPRESSION ANALYSIS SYSTEMATIC EXPLORER (EASE). To assist in the discovery of prevalent biological roles, MeV has an implementation of the EASE algorithm (Hosack *et al.*, 2003) for finding overrepresented biological themes in gene lists. This module compares the prevalence of a biological theme within the cluster to the prevalence of the biological theme in the population of genes from which the cluster was extracted. One must first have all of the array probes assigned various classes based on a categorical classification system, such as assignment of gene ontology terms (Ashburner, 2000). After selecting a set of genes that are ''significant'' in an analysis based on a statistical or other objective test, EASE compares representation of the various classes within the significant set to the representation on the entire array using Fisher's exact test to identify overrepresented categories and assign a likelihood score (p value) to each group. For example, if only 10% of the genes on the array represent energy metabolism, but 60% of the genes deemed significant are involved in

energy metabolism, it is likely that this selection did not occur by chance and that energy metabolism may be mechanistically involved in the process being studied.

Interface Orientation and Selected Features

The interface of MeV is organized into four main sections (Fig. 12). The *main menu bar* (A) contains the main menus for file loading and output, data transformations and analysis, display options and utility functions, as well as other key tasks. The *algorithm tool bar* (B) organizes the algorithm module buttons into the rough algorithm categories described earlier. An abbreviated module name and graphic on each analysis button clearly indicates the analysis. The *result navigation tree* (C) is used to organize and navigate analysis results. Clicking on a node in the tree will open a viewer associated with the labeled node in the *viewer display panel* (D) to the right of the tree. The navigation tree also contains the cluster manager, script manager, and the analysis history log.

MeV has many features to help researchers extract significant information from data and clustering results. This section describes some of the most basic functions and capabilities in the order that they would be encountered during analysis.

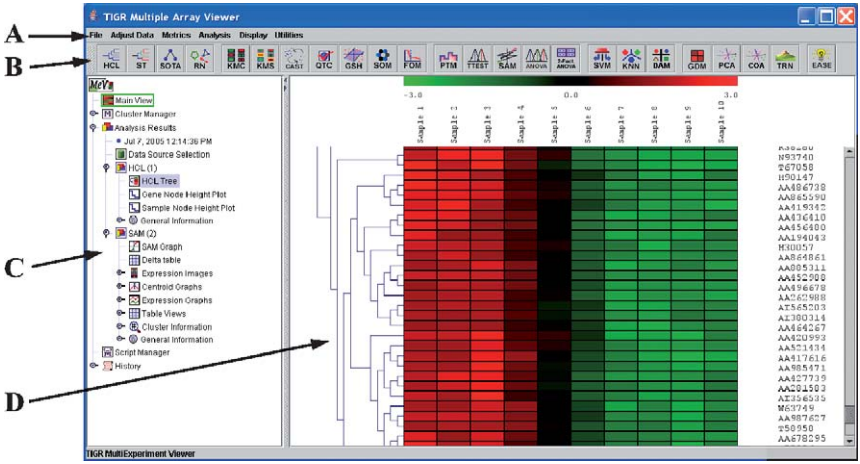

FIG. 12. Graphical interface of MeV: (A) main menu bar, (B) algorithm toolbar, (C) result navigation tree, and (D) viewer panel. A hierarchical tree viewer is shown in the viewer panel.

File Loading/Data Filtering/Data Transformations

MeV supports the loading of six expression file formats, including Affymetrix, GenePix (.gpr), Agilent, and the TM4 suite's .mev format. A variety of *Data filters* can be applied to the loaded data to remove data of low quality, genes (rows) with few valid data measurements, or genes that show little change over the loaded experiments.

Data transformations can also be performed from the Adjust Data menu. These transformations include *log transformations* of expression values and *mean centering*, where each gene expression vector is shifted such that the mean of the values in each vector is zero.

Cluster Viewers

Nodes that represent clustering results are appended to the result tree as they are created. Clusters can be viewed in the viewer display panel by clicking on these nodes in the result tree. In addition to many algorithm-specific viewers, MeV provides four basic *cluster viewers* (Fig. 13) to view the expression and

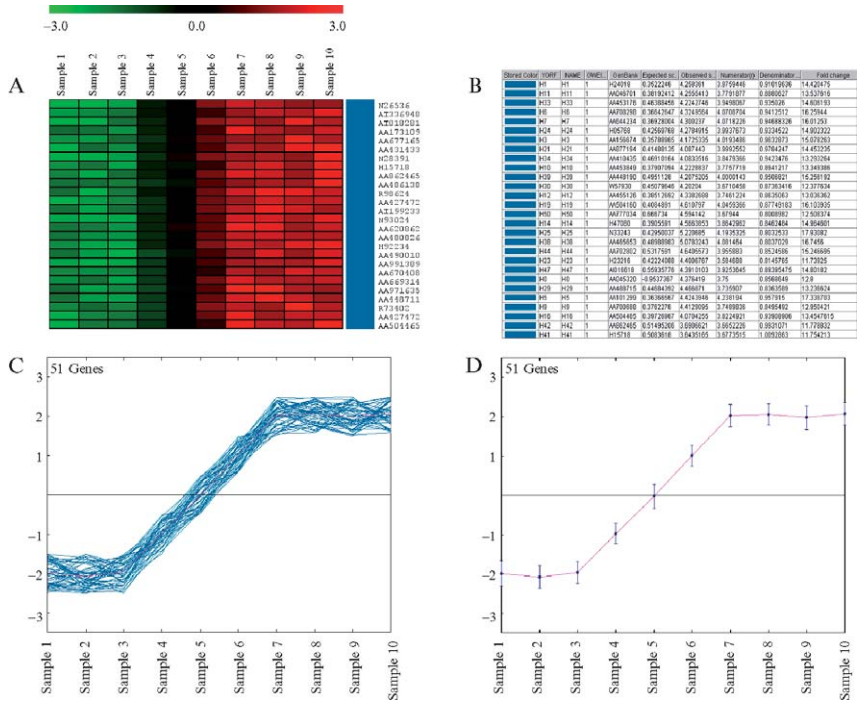

FIG. 13. Cluster viewer examples, (A) Expression image, (B) cluster table viewer, (C) expression graph, and (D) centroid graph.

membership of each cluster. *Expression Image*s display an expression matrix that corresponds to the subset of genes within the viewed cluster in which the expression level of each gene (row) across several experiments (columns) is displayed as a color that reflects the level of expression. The *Cluster Table Viewer* displays all gene annotation relating to the genes in the cluster and supports sorting on annotation, searches, and many other useful features. *Expression Graphs* display a graph showing the expression of each gene in the cluster over the set of loaded samples, whereas *Centroid Graphs* only show the cluster's mean expression pattern with error bars ($\pm$ 1 SD).

Cluster File Output/Cluster Storage/Cluster Operations

Once formed, clusters can be output to file in a tab-delimited text format that contains all expression and annotation data for the genes in the cluster. This format can be viewed as a spreadsheet and can be loaded easily into MeV to further visualize and mine that subset of data. Clusters can also be stored in MeV's *Cluster Manager,* which is a repository of selected clusters that can be viewed via the Cluster Manager node in the result tree. User-defined attributes such as a cluster label and description can be stored with the cluster as well as an assigned color that can be used to track the location of the cluster members during analysis. The assigned color propagates through all viewers to provide a qualitative measure of cluster overlap between analysis methods or runs. The cluster manager table provides many useful options, but the most useful are *cluster set operations,* such as cluster unions, intersections, and exclusive OR . These operations allow one to combine clusters of interest or to find genes common to two or more clusters.

Analysis Branching

A common task during analysis is to use an algorithm to reduce data to a set of interesting genes and then to extract this data subset for further analysis. We term this basic function where data are split off and analyzed as *analysis branching*. MeV provides three ways to perform analysis branching: (1) save the cluster as a file and then load it into a new MeV session as described, (2) use a feature of cluster viewers to automatically launch a new session that contains only the genes (or samples) in the cluster, and (3) right click on the cluster node in the result tree and select a check box to set that cluster as the primary data source for further analysis.

Analysis Scripting

The graphical nature of MeV lends itself to direct interaction, and it is often required that algorithms be applied several times to hone in on appropriate parameter values. An alternative to the interactive mode of MeV is the scripting mode. MeV provides graphical tools for script building, representation, and execution (Fig. 14). Once constructed, the XML

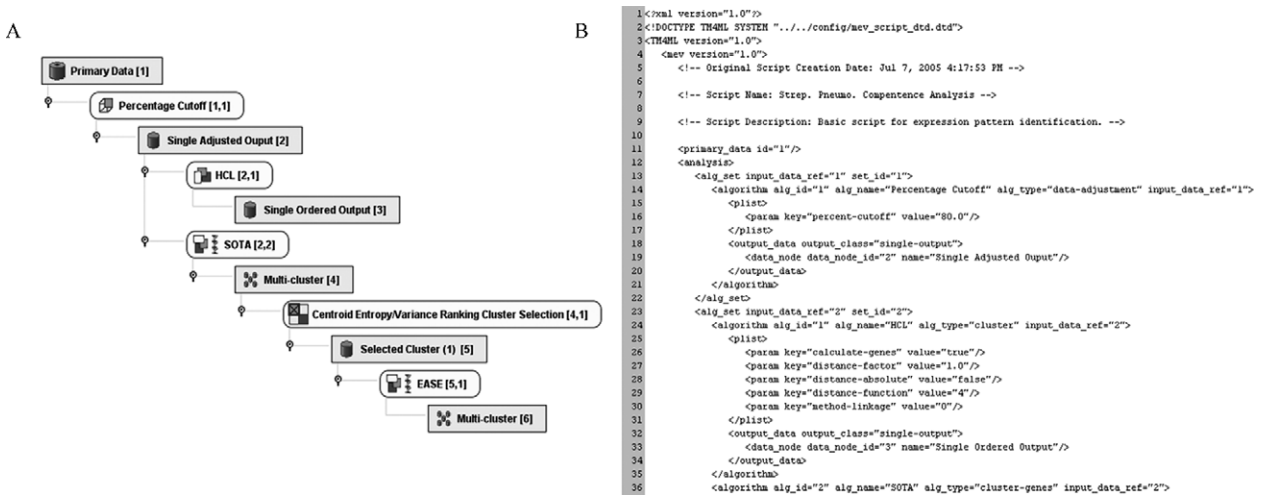

FIG. 14. Script viewers of MeV. (A) Graphical script tree viewer and (B) corresponding script XML viewer (script section).

analysis script can be saved and shared with collaborators to define analysis pipelines that reveal features of interest.

History Log

All analysis operations, from file loading, data filtering, algorithm runs, cluster storage, and file output, are recorded in a history log that describes the operation and attaches a time stamp. This serves as a detailed account of the analysis.

Sample Analysis Walk-Through

This section presents a sample Midas and MeV analysis that takes data through filtering and normalization, clustering and statistical analysis, and on to biological role analysis. To take full advantage of this walk-through it is best to download the applications and the sample data set so that one can follow along. Data for this analysis walk-through can be downloaded from this ftp site: ftp://ftp.tigr.org/pub/software/Microarray/MeV/MIE_data/. Each section indicates the proper files to use to illustrate the example. Midas and MeV can be downloaded from http://www.tm4.org/midas.html and http://www.tm4.org/mev.html.

Study Overview

This study investigates gene expression differences during ovalbumin induction of asthmatic responses. The study compared expression differences in mouse strains that are high or low responders to the stimuli in order to find genes that correlate to susceptibility or resistance. This example considers a low responder strain (CASTEi denoted as 'C' in the sample description) and a moderate responder strain (BALB\C denoted as 'B' in the sample description). For each strain there are biological duplicates for three time points: 24, 48, and 72 h. Each exposure time point had a corresponding vehicle control. The emphasis of this exercise is on the process of analysis rather than making specific claims about the nature of the findings.

Normalization Using Midas

This step filters low-quality spots using Spotfinder-generated flags, normalizes using block level Lowess and standard deviation regularization, and finally applies a flip-dye consistency filter. Because the same normalization process is repeated for each flip-dye pair (24 pairs), we will demonstrate the process on only one pair of raw files from the study as an example. The two files are contained in the sample data zip file and are labeled File_A_Sample Cy5_RefCy3.mev (file 'A') and File_B_SampleCy3_RefCy5.mev (file 'B').

These files contain the same sample and reference material but with the dye labels swapped.

Define the Analysis Pipeline

Open Midas by double clicking midas.bat in the midas directory. The analysis will proceed as follows.

a. Read two sample .mev format files as a flip-dye pair.
b. Execute Lowess (LocFit) normalization.
c. Execute standard deviation regularization (SD).
d. Perform the flip-dye consistency filter and file merge.
e. Write result files.

Select the analysis buttons in the Midas interface to construct the pipeline by referring to Fig. 15 as a guide to help identify the buttons for each step of the process. If a button is hit in error, one can clear the graphical pipeline and start again by clicking the left-most button in the tool bar (Fig. 15).

Modify the Parameters

Once the analysis pipeline matches the one in Fig. 15, you are ready to enter and modify the analysis parameters. Click on the first icon in the

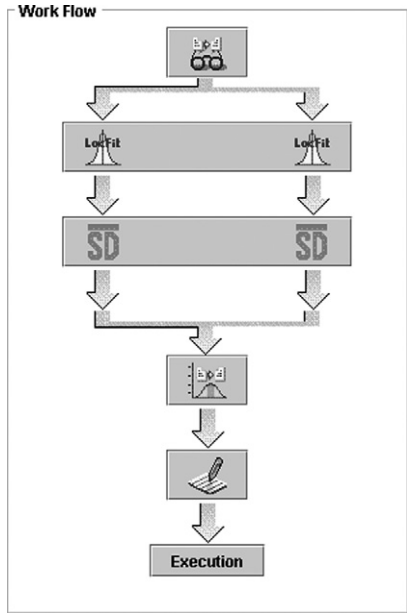

FIG. 15. The graphical scripting interface in Midas with the sample analysis pipeline indicates the order of processing operations.

pipeline that controls file loading. The parameter panel in the upper right will reflect available parameters. Select the input files by clicking on the empty field (first table cell) in the parameter panel. Use the file selection dialog to navigate to the analysis files and click on file 'A' and then ctrl-click on file 'B' to select the pair. Select the down arrow button to place the pair of files in the selection area and hit the OK button. Note that multiple file pairs can be analyzed by adding multiple pairs to the selection area. Select the check boxes to use the flags to filter low-quality data. Each of these selections will prompt a request for a flag column identifier. Just accept the default flag column header names. Review the parameters for the other parts of the pipeline by clicking on each of the remaining icons. Accept the default parameters for the other sections of the pipeline.

Select Output Reports

Select the *Reports* menu from the main menu bar and check the *Create PDF Reports* option. Now Midas is set to output a text-based result summary as well as a customizable pdf format analysis summary. The summary will contain input parameters, diagnostic plots, and numerical data related to the output such as the number of retained spots after filtering. Just before the analysis starts a dialog will be presented to customize the PDF report. For this example keep all graphs. When processing many files it is best to limit the file output to the key plots for each analysis stage, as the PDF creation requires a large amount of memory.

Execute the Analysis Pipeline

Select the *Execute* button to trigger execution. The final step is to select a project folder for output and to specify the project file name to store the pipeline and parameters. The progress of the analysis will be indicated in the analysis log at the bottom of the interface. Once the analysis is complete the diagnostic plots can be reviewed to assess the impact of the procedures.

Assessing the Results: The Investigation Panel

Open the *Investigation* panel by clicking on the tab just below the button panel to use Midas to view diagnostic plots. Use the file tree on the left side to navigate to the folders that contain the results. A right click on any plot will open a menu to allow you to view or plot the output file. The folder labeled *raw* contains the plots of data in its initial state. Plots of the same type can be overlaid to view the effect of normalization by first plotting raw data and then plotting normalized data. Some plots to try are histograms (.his) and R-I plots (.prc) in raw and post Lowess, box plots (.box) before and after standard deviation regularization, and in the flip dye folder you can view the flip dye diagnostic plots (.rrc) before and after filtering (Fig. 16).

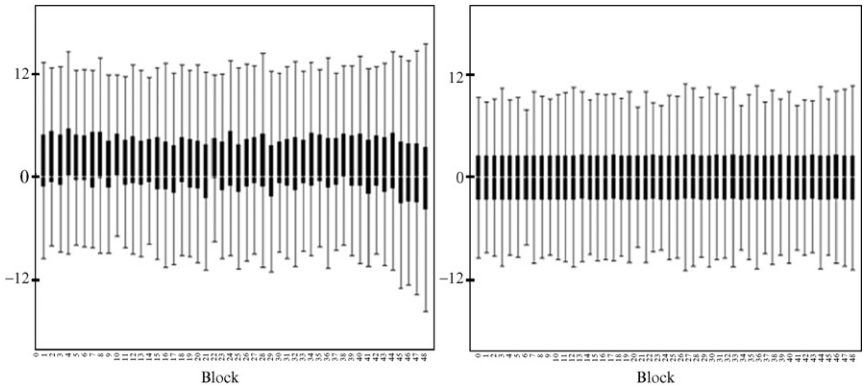

FIG. 16. Box plots of raw data (left) and data after Lowess normalization and standard deviation regularization (right). Note the centering effect of Lowess on block level mean log ratios. The nearly equal span of the middle quartiles of each block reflects the variance regularization following the SD regularization step.

Statistical Analysis and Clustering Using MeV

Now that data have been normalized to remove systematic bias and filtered to remove spots that are not expressed consistently we can use MeV to perform statistical analysis, clustering, and functional analysis. Data for this section started as raw mev files and were normalized and filtered as described earlier. The processed files for both strains were loaded into MeV in an order according to strain, exposure (control or experimental), and time point. The resulting expression matrix was saved to a single file to help streamline data loading for this example. The data file is in the sample data zip file and is labeled *CastEi_Balbc_combined_TDMS.txt*.

Launching MeV, File Loading, and Adjusting the Display

Double click on *tmev.bat* to launch MeV. The multiple array viewer can be resized to full screen by clicking on the maximize button in the upper right corner of the window. Select *Load Data* from the *File* menu of the multiple array viewer. The top part of the file loader interface will have a drop-down menu that is used to select the input file format. Select the second menu option labeled *Tab Delimited, Multiple Sample Files (TDMS)*. Use the file navigation tree on the left to navigate to sample data and select *CastEi_Balbc_combined_TDMS.txt* from the available files window. Selecting the file will present a portion of the file in the expression table preview panel on the right. Click on the first expression value in the upper-left position of the expression values. For this file the value happens to be NaN, as this value is missing or was filtered out. Selection of this table

cell informs the loader that rows above and to the left are sample and gene annotation. Click *Load* to load the data file.

The initial main view of the expression matrix will include sample names that correspond to the original mev files. From the *Display* menu select *Sample/Column Labels* and then *Select Sample Label* to *Label by Sample Description*. The sample annotation now contains strain ID (C or B), condition (control or experimental), time, and replicate ID. To improve the appearance of the expression matrix, modify the gradient scale limits by using the *Set Color Scale Limits* option from the *Display* menu. Set the lower limit to –2.0 and the upper limit to 2.0.

Filtering out Missing Data

It is common to have genes within loaded data that have few valid intensity measurements over the loaded samples. These rows with a lot of missing data appear mostly gray in the expression matrix image. To filter these genes out, open the *Adjust Data* menu and open the *Data Filters* menu and select the *Percentage Cutoff Filter* option. Enter 85.0 in the input dialog to keep only genes for which greater than 85% of the samples have values. A data filter result node will be placed on the result tree to report the number of genes that remain and to provide a view of the conserved rows. The log of the history node will also report the filtering result. Note that 27,648 rows were loaded and after applying the filter 20,048 rows remain for further analysis.

Statistical Analysis

Because there are two strains and two conditions, a 2×2 design two-factor ANOVA can be applied if we treat all time points as being in one group. Hit the two-factor ANOVA analysis button to open the dialog. In the first dialog label factor A as strain and factor B as condition and enter two levels for each factor, as there are two strains and two conditions, control and experimental. Advance to the next dialog to make group assignments. Designate strain membership in the upper left panel by selecting group 1 for all 'C' strain samples and group 2 for all 'B' strain samples. Designate condition by placing all controls in group 1 and all experimental samples in group 2 in the group selection panel on the right. Set the critical value of p to 0.001. Near the bottom of the dialog select the check box to build HCL trees on significant genes and hit the OK button.

An HCL initialization dialog will come up to select parameters for HCL. Deselect the option to make sample trees so that samples are not reordered. Select the *Pearson Correlation* as the distance metric and hit OK.

Interaction significant genes from two-factor ANOVA in the result tree are those that show differences in response to exposure that are dependent on strain. In this example this mostly consists of genes that changed in the moderate responder strain B but not in the low responder

strain C. One can view the various results by clicking on the viewer nodes in the result tree.

Dissecting Significant Genes

To further explore the interaction significant genes, click on the HCL viewer in the ANOVA result for the interaction significant cluster. Right click in the HCL cluster viewer and select *Gene Tree Properties*. Slide the slider bar to the right until the number of terminal nodes is about two or three. Select the check box labeled *Create Cluster Viewers* and hit OK. This will create clusters that correspond to subtrees with the full HCL viewer. The viewer nodes will be appended under the HCL viewer node on the result tree. Figure 17 shows the cluster centroid viewers that correspond to the two dominant patterns in the interaction significant cluster where genes were upregulated or downregulated in only the moderate responder strain B (Fig. 17).

Storing Clusters and Cluster Operations

A right click-activated menu provides a *Store Cluster* option in most cluster viewers that allows one to store clusters of interest to the cluster manager. Open a viewer other than the HCL viewer that displays all significant interaction genes, right click, and select *Store Cluster*. The cluster can be assigned attributes such as a label and a description. Selection of a cluster

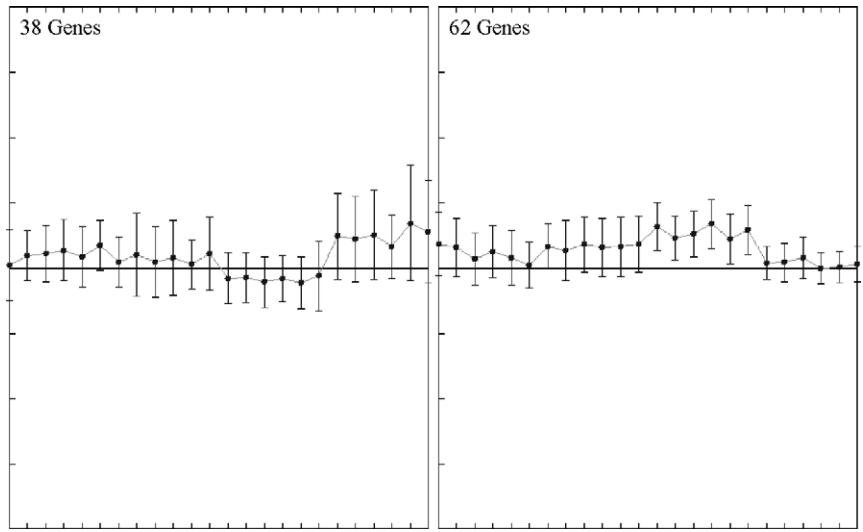

FIG. 17. Centroid graphs showing genes with a significant interaction effect. (Left) Mean profile for 38 genes that were overexpressed in the high responder strain. (Right) Mean profile for the 62 genes that were underexpressed in the responder strain. Error bars are ±1 SD. (Two-factor ANOVA results, interaction significant genes, $p < 0.001$.)

color is required and can be used to track the genes during analysis. Stored clusters can be viewed in the cluster manager node's gene cluster table above the main analysis node in the result tree. If multiple clusters exist from various results, one can use the cluster operations in the cluster manager to perform cluster operations such as cluster unions, intersections, or exclusive OR.

Exploring Biological Themes

The EASE module can be used to investigate the biological roles within a cluster of interest. All clusters stored in the cluster manager are candidates for EASE analysis. The data directory of MeV has an EASE file system to support the analysis of this data set. Select the EASE button from the right end of the analysis tool bar or from the analysis menu. The center portion of the dialog has three tabbed panels. The first panel is used is to designate a population of genes and a cluster for analysis. Select the *Population from Current Viewer* option to define the population. Select the cluster to analyze by selecting a row in the cluster table. On the second tab check that *tc#* is selected as the gene identifier. In the bottom portion of the panel select the button to add annotation/ontology linking files and use ctrl-click to select the three GO files and the KEGG pathways file. Accept the defaults for the statistical parameters panel by hitting OK. The resulting table will list all biological roles that were identified for the cluster, and the roles will be ordered by the provided *p* value for each role. The GO hierarchical viewer will show themes in a hierarchy of specificity. MeV's manual, slide presentation in the *documentation/presentations* folder, and the EASE reference (Hosack *et al.*, 2003) will describe the parameter selections, theory basics, and the statistical details behind EASE analysis.

Further Analysis

The purpose of this section was to provide a basic sample analysis. Various other tests can be run on this data set to extract other genes of interest. The power of any analysis tool comes with the understanding of the available analysis modules and features and how they can be used to extract a variety of findings relevant to the study.

References

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., and Sherlock, G. (2000). Gene ontology: Tool for the unification of biology. *Nature Genet.* **25,** 25–29.

Ayroles, J. F., and Gibson, G. (2006). Analysis of variance of microarray data. *Methods Enzymol.* **411,** 214–233.

Barrett, T., and Edgar, R. (2006). Gene Expression Omnibus (GEO): Microarray data storage, submission, retrieval and analysis. *Methods Enzymol.* **411,** 352–369.

Ben-Dor, A., Shamir, R., and Yakhini, Z. (1999). Clustering gene expression patterns. *J. Comput. Biol.* **6,** 281–297.

Brazma, A., Hingamp, P., Quackenbush, J., Sherlock, G., Spellman, P., Stoeckert, C., Aach, J., Ansorge, W., Ball, C. A., Causton, H. C., Gaasterland, T., Glenisson, P., Holstege, F. C., Kim, I. F., Markowitz, V., Matese, J. C., Parkinson, H., Robinson, A., Sarkans, U., Schulze-Kremer, S., Stewart, J., Taylor, R., Vilo, J., and Vingron, M. (2001). Minimum information about a microarray experiment (miame)-toward standards for microarray data. *Nature Genet.* **29,** 365–371.

Brazma, A., Kapushesky, M., Parkinson, H., Sarkins, U., and Shojatalab, M. (2006). Data storage and analysis in ArrayExpress. *Methods Enzymol.* **411,** 370–386.

Brazma, A., Parkinson, H., Sarkans, U., Shojatalab, M., Vilo, J., Abeygunawardena, N., Holloway, E., Kapushesky, M., Kemmeren, P., Lara, G. G., Oezcimen, A., Rocca-Serra, P., and Sansone, S. A. (2003). Arrayexpress: A public repository for microarray gene expression data at the EBI. *Nucleic Acids Res.* **31,** 68–71.

Brown, M. P., Grundy, W.N, Lin, D., Cristianini, N., Sugnet, C. W., Furey, T. S., Ares, M., Jr., and Haussler, D. (2000). Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proc. Natl. Acad. Sci. USA* **97,** 262–267.

Chen, Y., Dougherty, E. R., and Bittner, M. L. (1997). Ratio-based decisions and the quantitative analysis of cDNA microarray images. *J. Biomed. Optics* **2,** 364–374.

Chu, G., Narasimhan, B., Tibshirani, R., and Tusher, V. (2002). SAM ''Significance Analysis of Microarrays.'' Users Guide and Technical Document.http://www-stat.stanford.edu/~tibs/SAM/.

Culhane, A. C., Perriere, G., Considine, C., Cotter, T. G., and Higgins, D. G. (2002). Between-group analysis of microarray data. *Bioinformatics* **18**(12), 1600–1608.

Dalma-Weiszhauz, D. D., Warrington, J., Tanimoto, E. Y., and Miyada, C. G. (2006). The Affymetrix Gene Chip® platform: An overview. *Methods Enzymol.* **410,** 3–28.

Dopazo, J., and Carazo, J. M. (1997). Phylogenetic reconstruction using an unsupervised growing neural network that adopts the topology of a phylogenetic tree. *J. Mol. Evol.* **44,** 226–233.

Downey, T. (2006). Analysis of a multifactor microarray study using Partek Genomics Solution. *Methods Enzymol.* **411,** 256–270.

Dudoit, S., Shaffer, J. P., and Boldrick, J. C. (2003). Multiple hypothesis testing in microarray experiments. *Stat. Sci.* **18,** 71–103.

Dudoit, S., Yang, Y. H., Callow, M. J., and Speed, T. (2000). Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. Technical Report 2000, Statistics Department, University of California, Berkeley.

Eads, B., Cash, A., Bogart, K., Costello, J., and Andrews, J. (2006). Troubleshooting microarray hybridizations. *Methods Enzymol.* **411,** 34–49.

Edgar, R., Domrachev, M., and Lash, A. E. (2002). Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Res.* **30**(1), 207–210.

Eisen, M. B., Spellman, P. T., Brown, P. O., and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA* **95,** 14863–14868.

Fellenberg, K., Hauser, N. C., Brors, B., Neutzner, A., Hoheisel, J. D., and Vingron, M. (2001). Correspondence analysis applied to microarray data. *Proc. Natl. Acad. Sci. USA* **98**(19), 10781–10786.

Finkelstein, D.B, Gollub, J., Ewing, R., Sterky, F., Somerville, S., and Cherry, J. M. (2000). Iterative linear regression by sector: Renormalization of cDNA microarray data and cluster analysis weighted by cross homology. In CAMDA. http://afgc.stanford.edu/afgc_html/site2Stat.htm.

Gaidoukevitch, Y. C., Ryan, K. W., and Sequera, D. E. (2002). Method, system and product for analyzing image of an array to create an image of a grid overlay, U.S. patent 6,498,863.
George, R. A. (2006). The printing process: Tips on tips. *Methods Enzymol.* **410,** 121–135.
Jeremy Gollub, J., and Gavin Sherlock, G. (2006). Clustering microarray data. *Methods Enzymol.* **411,** 194–213.
Hastie, T., Tibshirani, R., Eisen, M. B., Alizadeh, A., Levy, R., Staudt, L., Chan, W. C., Botstein, D., and Brown, P. (2000). ''Gene shaving'' as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biol.* **1**(2), RESEARCH0003.
Herrero, J., Valencia, A., and Dopazo, J. (2001). A hierarchical unsupervised growing neural network for clustering gene expression patterns. *Bioinformatics* **17**(2), 126–136.
Heyer, L. J., Kruglyak, S., and Yooseph, S. (1999). Exploring expression data: Identification and analysis of co-expressed genes. *Genome Res.* **9,** 1106–1115.
Hosack, D. A., Dennis, G., Jr., Sherman, B. T., Lane, H. C., and Lempicki, R. A. (2003). Identifying biological themes within lists of genes with EASE. *Genome Biol.* **4,** R70–R78.
Johnston, R., Wang, B., Nuttall, R., Doctolero, M., Edwards, P., Lu, J., Vainer, M., Yue, H., Wang, X., Minor, J., Chan, C., Lash, A., Goralski, T., Parisi, M., Oliver, B., and Eastman, S. (2004). FlyGEM, a full transcriptome array platform for the Drosophila community. *Genome Biol.* **5** research0019.1-0019.11.
Keppel, G., and Zedeck, S. (1989). ''Data Analysis for Research Designs.'' Freeman, New York.
Kim, S. K., Lund, J., Kiraly, M., Duke, K., Jiang, M., Stuart, J. M., Eizinger, A., Wylie, B. N., and Davidson, G. S. (2001). A gene expression map for. *Caenorhabditis elegans. Science* **293,** 2087–2092.
Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biol. Cybernet.* **43,** 59–69.
Korn, E. L., Troendle, J. F., McShane, L. M., and Simon, R. (2001). Controlling the number of false discoveries: Application to high-dimensional genomic data. Technical report 003, Biometric Research Branch, National Cancer Institute. http://linus.nci.nih.gov/~brb/TechReport.htm.
Korn, E. L., Troendle, J. F., McShane, L. M., and Simon, R. (2004). Controlling the number of false discoveries: Application to high-dimensional genomic data. *J. Stat. Plan. Inference* **124,** 379–398.
Liao, P.-S., Chen, T.-S., and Chung, P.-C. (2001). A fast algorithm for multilevel thresholding. *J. Inform. Sci. Eng.* **17,** 713–727.
Lipshutz, R. J., Morris, D., Chee, M., Hubbell, E., Kozal, M. J., Shah, N., Shen, N., Yang, R., and Fodor, S. P. (1995). Using oligonucleotide probe arrays to access genetic diversity. *Biotechniques* **19**(3), 442–447.
Manly, B. F. J. (1997). ''Randomization, Bootstrap and Monte Carlo Methods in Biology,'' 2nd Ed. Chapman and Hall/CRC, FL.
Minor, J. M. (2006). Microarray quality control. *Methods Enzymol.* **411,** 233–255.
Neal, S. J., and Westwood, T. (2006). Optimizing experiment and analysis parameters for spotted microarrays. *Methods Enzymol.* **410,** 203–221.
Otsu, N. (1979). A treshold selection method from gray-level histogram. IEEE Transactions on System Man Cybernetics, v. SMC-9, No. 1, 62–66.
Pan, W. (2002). A comparative review of statistical methods for discovering differentially expressed genes in replicated microarray experiments. *Bioinformatics* **18,** 546–554.
Pavlidis, P., and Noble, W. S. (2001). Analysis of strain and regional variation in gene expression in mouse brain. *Genome Biol.* **2,** research0042.1-0042.15.
Quackenbush, J. (2002). Microarray data normalization and transformation. *Nature Genet.* **32**(Suppl.), 496–501.
Raychaudhuri, S., Stuart, J. M., and Altman, R. B. (2000). Principal components analysis to summarize microarray experiments: Application to sporulation time series. Pacific

Symposium on Biocomputing 2000 Honolulu, Hawaii 452–463. Available at http://smi-web.stanford.edu/pubs/SMI_Abstracts/SMI-1999-0804.html.

Reimers, M., and Carey, V. J. (2006). Bioconductor: An open source framework for bioinformatics and computational biology. *Methods Enzymol.* **411,** 119–134.

Royce, T. E., Rozowsky, J. S., Luscombe, N. M., Emanuelsson, O., Yu, H., Zhu, X., Snyder, M., and Gerstein, M. B. (2006). Extrapolating traditional DNA microarray statistics to tiling and protein microarray technologies. *Methods Enzymol.* **411,** 282–311.

Schena, M., Shalon, D., Davis, R. W., and Brown, P. O. (1995). Quantitative monitoring of gene expression patterns with a complimentary DNA microarray. *Science* **270**(5235), 467–470.

Sharov, V., Kwong, K. Y., Frank, B., Chen, E., Hasseman, J., Gaspard, R., Yu, Y., Yang, I., and Quackenbush, J. (2004). The limits of log-ratios. *BMC Biotechnol.* **4,** 3.

Soukas, A., Cohen, P., Socci, N. D., and Friedman, J. M. (2000). Leptin-specific patterns of gene expression in white adipose tissue. *Genes Dev.* **14,** 963–980.

Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E. S., and Golub, T. R. (1999). Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proc. Natl. Acad. Sci. USA* **96,** 2907–2912.

Theilhaber, J., Connolly, T., Roman-Roman, S., Bushnell, S., Jackson, S., Call, K., Garcia, T., and Baron, R. (2002). Finding genes in the C2Cl2 osteogenic pathway by K-nearest-neighbor classification of expression data. *Genome Res.* **12,** 165–176.

Timlin, J. A. (2006). Scanning microarrays: Current methods and future directions. *Methods Enzymol.* **411,** 79–98.

Troein, C., Vallon-Christersso, J., and Saal, L. H. (2006). An introduction to BioArray Software Environment. *Methods Enzymol.* **411,** 99–119.

Tusher, V. G., Tibshirani, R., and Chu, G. (2001). Significance analysis of microarrays applied to the ionizing radiation response. *Proc. Natl. Acad. Sci. USA* **98,** 5116–5121.

Weinstein, J. N., Myers, T. G., O'Connor, P. M., Friend, S. H., Fornace, A. J., Jr., Kohn, K. W., Fojo, T., Bates, S. E., Rubinstein, L. V., Anderson, N. L., Buolamwini, J. K., van Osdol, W. W., Monks, A. P., Scudiero, D. A., Sausville, E. A., Zaharevitz, D. W., Bunow, B., Viswanadhan, V. N., Johnson, G. S., Wittes, R. E., and Paull, K. D. (1997). An information-intensive approach to the molecular pharmacology of cancer. *Science* **275,** 343–349.

Welch, B. L. (1947). The generalization of 'students' problem when several different population variances are involved. *Biometrika* **34,** 28–35.

Whetzel, P. L., Parkinson, H., and Stoeckert, C. J. (2006). Using ontologies to annotate microarray experiments. *Methods Enzymol.* **411,** 325–339.

Yang, I. V., Chen, E., Hasseman, J. P., Liang, W., Frank, B. C., Wang, S., Sharov, V., Saeed, A. I., White, J., Li, J., Lee, N. H., Yeatman, T. J., and Quackenbush, J. (2002a). Within the fold: Assessing differential expression measures and reproducibility in microarray assays. *Genome Biol.* **3,** research0062.1-0062.12.

Yang, Y.-H., Buckley, M. J., Dudoit, S., and Speed, T. P. (2002b). Comparison of methods for image analysis on cDNA microarray data. *J. Comput. Graph. Stat.* **11**(No. 1), 108–136.

Yang, Y.-H., Buckley, M. J., and Speed, T. P. (2001). Analysis of cDNA microarray images. *Brief. Bioinform.* **2**(No. 4), 341–349.

Yang, Y. H., Dudroit, S., Luu, P., Lin, D. M., Peng, V., Ngal, J., and Speed, T. P. (2002c). Normalization for cDNA microarray data: A robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Res.* **30,** e15.

Yeung, K. Y., Haynor, D. R., and Ruzzo, W. L. (2001). Validating clustering for gene expression data. *Bioinformatics* **17,** 309–318.

Zar, J. H. (1999). ''Biostatistical Analysis,'' 4th Ed. Prentice Hall, New Jersey.