GenePattern Tutorial

December 6, 2005

BROAD INSTITUTE

Table of Contents

GenePattern 2.0 Tutorial

In this tutorial you will learn how to use the features of GenePattern, including:

- How to use GenePattern to do microarray analysis
- How to create reusable pipelines that capture your own analyses
- How to extend GenePattern to work with other tools and data
- How to call GenePattern modules from a programming language
- How to manage jobs on a GenePattern server
- How to change the GenePattern server configuration

This tutorial is written specifically for use with GenePattern 2.0. You will need to be connected to the internet to complete some of the exercises.

About GenePattern

GenePattern is a software package designed to address several requirements for integrative genomics analysis: an intuitive interface for novice users, capability for reproducible research, accessibility to programmers and computational biologists, simple addition of user-created analysis tasks, and the ability to deploy on a wide range of computing hardware. These design objectives are described below.

Accessibility to novice users

Intuitive graphical interface for "novice" user

Users of analysis tools comprise a wide range of individuals. There are biologists, researchers and clinicians with little or modest computational backgrounds who need flexibility to run tools and look at results without writing programs themselves or finding help from a computational colleague. They should be able to interact with a system that encompasses their needs and which can be easily enhanced to meet ever-changing requirements.

A main distinguishing feature of GenePattern is the inclusion of multiple user environments including a friendly user interface. The "friendly" interface provides prepackaged analysis modules. These include both individual algorithms and more complex methodologies encapsulated as pipelines. Modules and pipelines are available to the user via pull-down menu task lists, and datasets are accessible through a standard file browser.
Learn more about the Graphical Environment

Reproducible Research

Pipeline environment

Typical data analysis involves a complex multi-step sequence of computation and data access across a multitude of tools, platforms, and data sets. Joining these steps together in one-off scripts can be tiresome and the end product may not be accessible for use by colleagues.

The GenePattern pipeline builder provides an easy to use, form-based method for combining data processing, analytic, and visualization modules together into methodologies. Through its pipeline environment, GenePattern provides the means to create and distribute an entire computational analysis methodology in a unified and executable script thus enabling a form of *in silico* reproducible research.

Life Science IDentifiers

An important aspect of reproducible research is the use of Life Science IDentifiers (LSIDs) to uniquely identify specific versions of tasks and pipelines. Whenever a new task or pipeline is created in GenePattern it is assigned a unique LSID. Every time that the task or pipeline definition is subsequently modified, the version portion of the LSID is incremented. LSIDs enable integration of disparate data sources and analysis methods while allowing researchers to pick and choose the tools that best meet their needs. By unambiguously defining a particular dataset or tool, they allow researchers to develop and share working models and publish papers which others can test and extend.

Learn more about the Pipeline Environment

Accessibility to programmers and computational biologists

Access to analysis modules from a programming language

The GenePattern Programming Language Environment supports computationally sophisticated users by providing access to GenePattern from R, Java, and MATLAB. This environment permits arbitrary scripting, access to GenePattern modules via function calls, and development of new methodologies combining modules in arbitrarily complex combinations all from within a researcher's favorite programming environment.
Learn more about the Programming Language Environment

Flexibility & ease of integration of new tasks

Self-service model for integration of new methodologies

GenePattern provides a rapid, language-agnostic deployment and integration mechanism for new modules that requires very little software engineering. With GenePattern, new methods, written in the author's favorite programming language, can be shared, tested, and used by an entire lab or research community rapidly and effectively within the GenePattern environment. New methods can be developed locally, and the architecture also supports the rapid, low-overhead integration of algorithms and tools developed externally and downloaded or accessed from the web. The advantage of the GenePattern approach is that users can quickly and seamlessly create, evaluate, and adopt new methods.
Learn more about the Task Integrator

Local or distributed computing

GenePattern is designed so that it can run standalone on a small machine such as a laptop, or can be separated into its client and server components to take advantage of a more powerful compute server. A single installer will work for both local or distributed configurations. After installing your GenePattern server, you will want to configure GenePattern for your site.
Learn more about GenePattern Server Administration

If you are running in a distributed environment, you may also want to secure your GenePattern server.
Learn more about Securing Your GenePattern Server

Getting Started with GenePattern

This tutorial assumes you have done a complete install of GenePattern, including the server, the modules, the graphical client, and the datasets. If you haven't installed all of these components, please go back and follow the installation instructions. If your GenePattern Server is running, you may revisit your installation summary here or type `http://localhost:8080/gp/installFrame.jsp` into your browser.

Launching GenePattern

There are three components to GenePattern:

- the server, which is the GenePattern engine
- the Graphical Client (also called the Java Client), which is the primary GenePattern interface for most users
- the Web Client (also called the GenePattern home page or GenePattern server home page), which is primarily used for administrative tasks, such as adding and deleting analysis modules; the Web Client also provides access to the GenePattern tasks accessible from the Graphical Client

To run GenePattern, you must launch the server first. You can then launch the Graphical Client and/or the Web client.

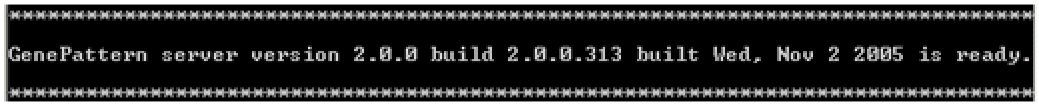
Launching the Server

To launch the GenePattern server:

- Double-click the **`Start GenePattern Server`** icon, shown below. It is located where you chose to place it during the server installation.

`StartGenePatternServer`

- While the server is launching, you will see a console window displaying messages. When you see the "server ready" message below, the server is running.



```
*****************************************************************************
GenePattern server version 2.0.0 build 2.0.0.313 built Wed, Nov 2 2005 is ready.
*****************************************************************************
```

! Mac OS X users will not see this window. On Mac OS X, the server icon will appear and bounce in the Dock when the server is launched. When the icon stops bouncing the server is ready. If you wish to read the messages, they can be seen in the Console.

- You are now running the GenePattern Server.

!
1. You must launch the server before launching the Graphical or Web Clients.
2. The server takes several seconds to start up. Clients will not be able to connect to it during this time.

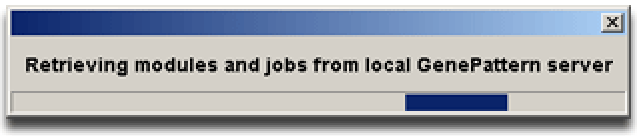
Launching the GenePattern Graphical Client

To launch the GenePattern Graphical Client:

- Double-click the **`GenePattern Client`** icon, shown below. It is located where you chose to place it during the client installation.

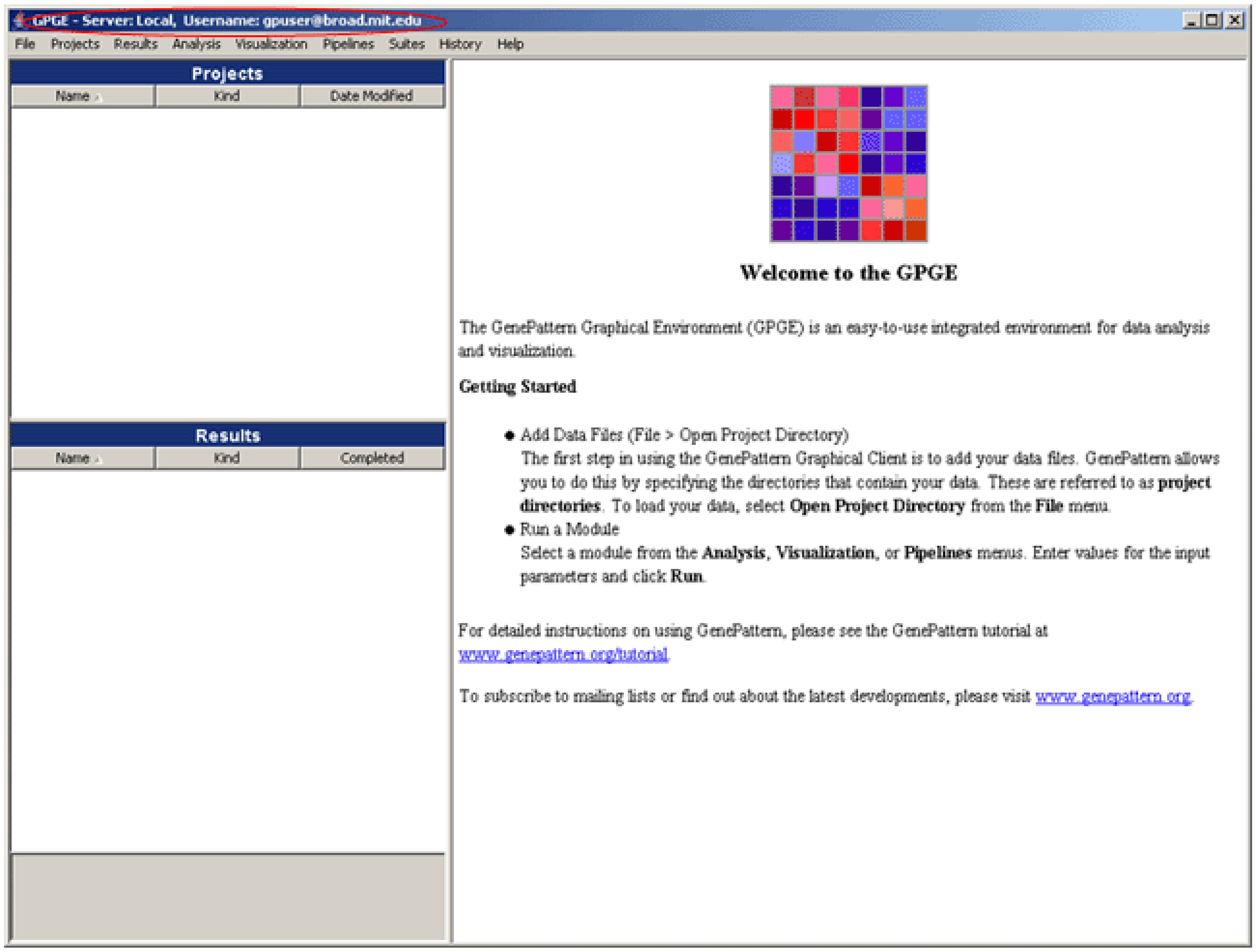
`GenePatternClient`

- While the client is launching you will see a window indicating that the client is connecting to the server and loading your installed modules.



Retrieving modules and jobs from local GenePattern server

When the client is connected it will ask you to log in. Log in with your email address. Once you are logged in, all of the loaded analysis tasks, visualizers, and pipelines are available. You can see which server and which email address you are using by looking at the top of your client window.


GPGE - Server: Local, Username: gpuser@broad.mit.edu
File Projects Results Analysis Visualization Pipelines Suites History Help
Projects
Name Kind Date Modified
Results
Name Kind Completed

Welcome to the GPGE

The GenePattern Graphical Environment (GPGE) is an easy-to-use integrated environment for data analysis and visualization.

Getting Started

- Add Data Files (File > Open Project Directory)
  The first step in using the GenePattern Graphical Client is to add your data files. GenePattern allows you to do this by specifying the directories that contain your data. These are referred to as **project directories**. To load your data, select **Open Project Directory** from the **File** menu.
- Run a Module
  Select a module from the **Analysis**, **Visualization**, or **Pipelines** menus. Enter values for the input parameters and click **Run**.

For detailed instructions on using GenePattern, please see the GenePattern tutorial at www.genepattern.org/tutorial.

To subscribe to mailing lists or find out about the latest developments, please visit www.genepattern.org.

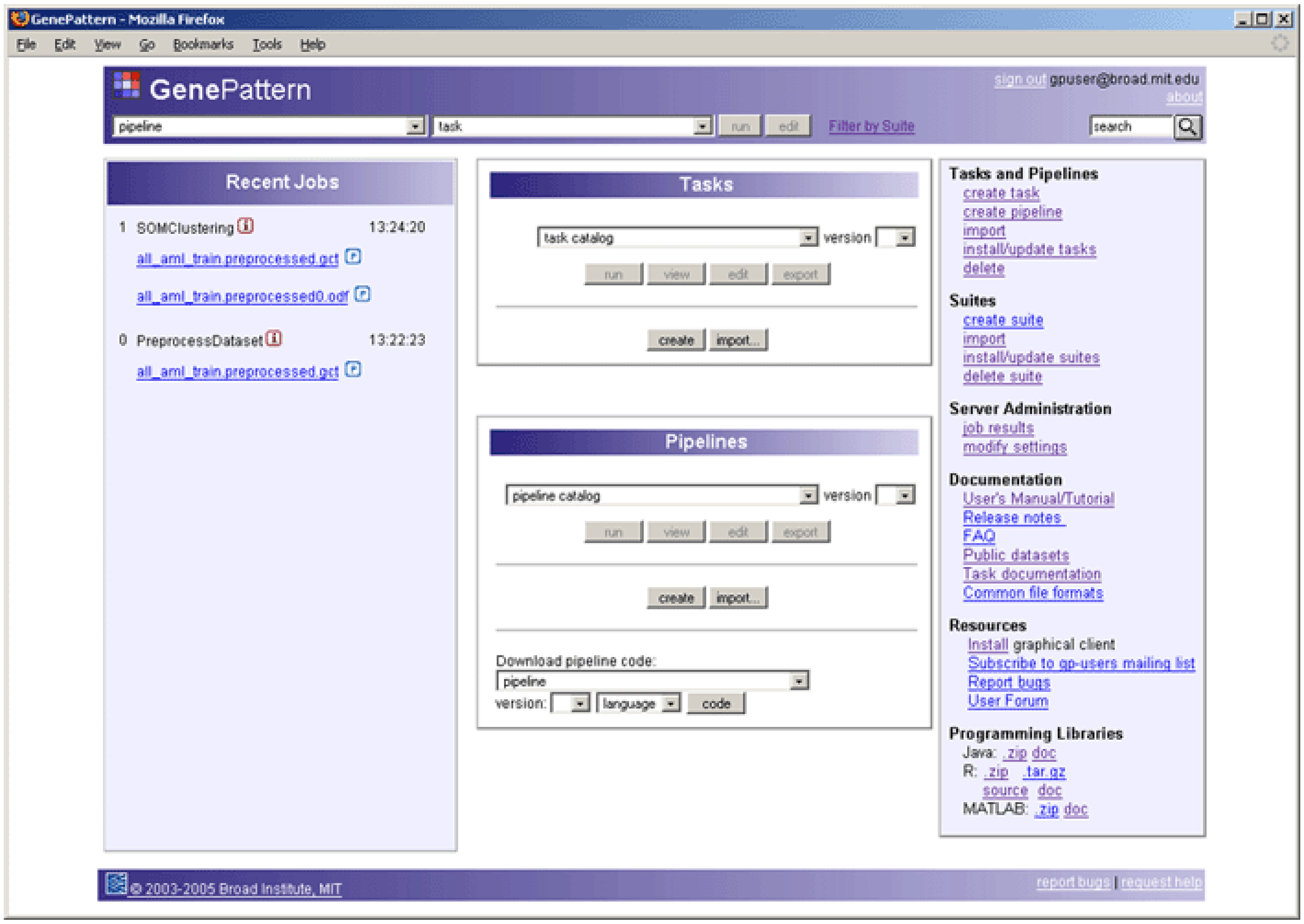
Launching the GenePattern Web Client

To launch the GenePattern Web Client:

1. Double-click on the **`GenePatternHome.html`** link:

   `GenePatternHome.html`

   Your browser will open and ask you to login. Once you've logged in with your email address you will see the GenePattern server home page:



You can also view this page by entering the URL of your GenePattern server into a Web browser. If you are running GenePattern in standalone mode, the URL will be `http://127.0.0.1:8080/gp`.

Shutting Down GenePattern

When you shut down GenePattern, shut down the clients and then the server. If a client remains running after the server has been shut down, you will see errors.

To shut down the GenePattern clients and server:

1. Exit from the GenePattern Graphical Client, if it is running, by choosing the `File->Quit` menu.
2. Exit from the GenePattern Web Client, if it is running, by clicking the `sign out` link in the top right corner of the page.
3. Exit from the server by double-clicking the `Stop GenePattern Server` icon. Alternatively, if you are using Windows, you can close the console window.

`StopGenePatternServer`

Windows users will see the GenePattern console window close. Mac users will see the GenePattern server icon disappear from the Dock. On Linux machines, the GenePattern server will quit silently.

GenePattern Graphical Environment

The GenePattern Graphical Environment (GPGE) is an easy-to-use integrated environment for data analysis and visualization. In this section, you will learn how to view data, launch analyses, and visualize results using the GenePattern Graphical Client.

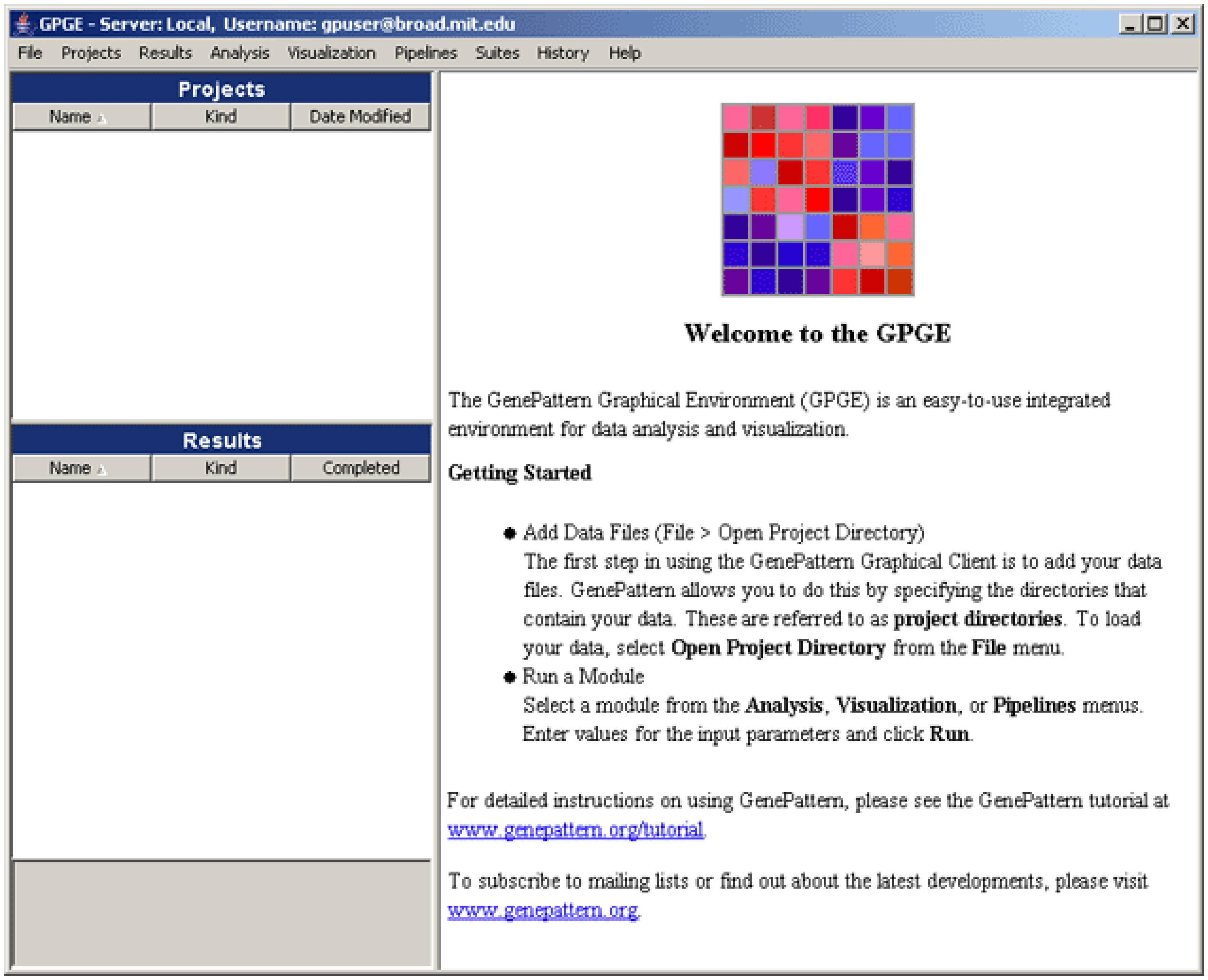
NOTES

!

- To run the exercises in this section, the GenePattern server and Graphical Client must be running. If you have not launched them, you can follow the directions for launching the GenePattern server and Graphical Client in Getting Started.
- The exercises require the GenePattern tutorial datasets. If you did not download them during installation, you can download them here or from the datasets page, http://www.broad.mit.edu/cancer/software/genepattern/datasets. You will need to use a compression utility such as WinZip (Windows) or StuffIt Expander (Mac) to extract the datasets.

Viewing Data in GenePattern

After you launch the GenePattern Client, you will see the Object Browser, shown below:


GPGE - Server: Local, Username: gpuser@broad.mit.edu
File Projects Results Analysis Visualization Pipelines Suites History Help
Projects
Name
Kind
Date Modified
Results
Name
Kind
Completed

Welcome to the GPGE

The GenePattern Graphical Environment (GPGE) is an easy-to-use integrated environment for data analysis and visualization.

Getting Started

- Add Data Files (File > Open Project Directory)
  The first step in using the GenePattern Graphical Client is to add your data files. GenePattern allows you to do this by specifying the directories that contain your data. These are referred to as **project directories**. To load your data, select **Open Project Directory** from the **File** menu.
- Run a Module
  Select a module from the **Analysis**, **Visualization**, or **Pipelines** menus. Enter values for the input parameters and click **Run**.

For detailed instructions on using GenePattern, please see the GenePattern tutorial at www.genepattern.org/tutorial.

To subscribe to mailing lists or find out about the latest developments, please visit www.genepattern.org.

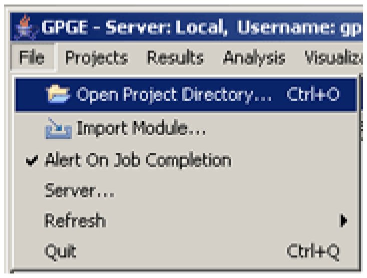
This is the central control panel for the GenePattern graphical environment. This window allows you to load data, execute algorithms, launch visualizers, and create pipelines.

Using Files in GenePattern

The first step in using the GenePattern Graphical Client is to add your data files to the environment. GenePattern allows you to do this by specifying the directories that contain your data. These are referred to as **project directories**. In the following exercises, you will learn how to add, remove, and view the contents of project directories.

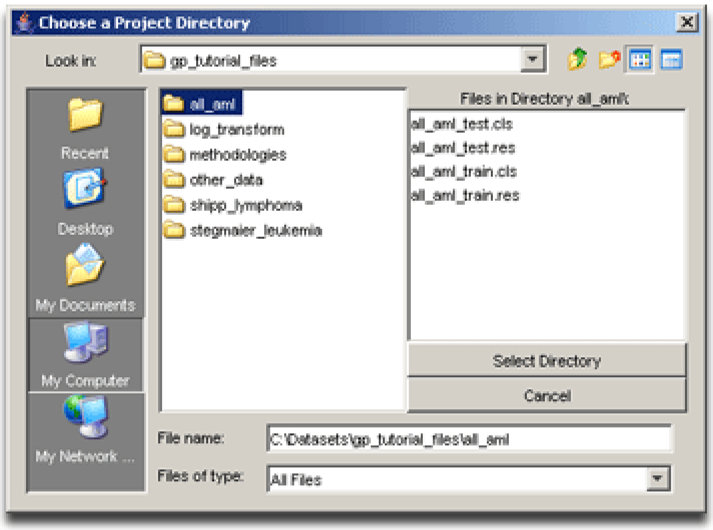
Exercise 1a: Adding a project directory

Select `File->Open Project Directory...`

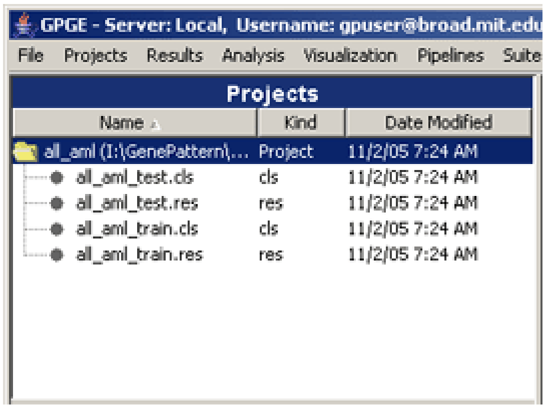

GPGE - Server: Local, Username: gp
File Projects Results Analysis Visualiz
Open Project Directory... Ctrl+O
Import Module...
✓ Alert On Job Completion
Server...
Refresh
Quit Ctrl+Q

Navigate to the **`gp_tutorial_files`** folder you downloaded, which contains the all_aml datasets. Note that the left side of the window contains all of the subdirectories you can choose, and the right side contains a listing of the files in the currently selected directory.

- Click ONCE on **`all_aml`**.
- Click on **`Select Directory`**.



You will see that the directory has been added as a project node. Double-click on the **`all_aml`** node to see files within that directory sorted alphabetically by name. You may also sort by kind as well as by date.



Name	Kind	Date Modified
all_aml (I:\GenePattern\...	Project	11/2/05 7:24 AM
all_aml_test.cls	cls	11/2/05 7:24 AM
all_aml_test.res	res	11/2/05 7:24 AM
all_aml_train.cls	cls	11/2/05 7:24 AM
all_aml_train.res	res	11/2/05 7:24 AM

NOTES

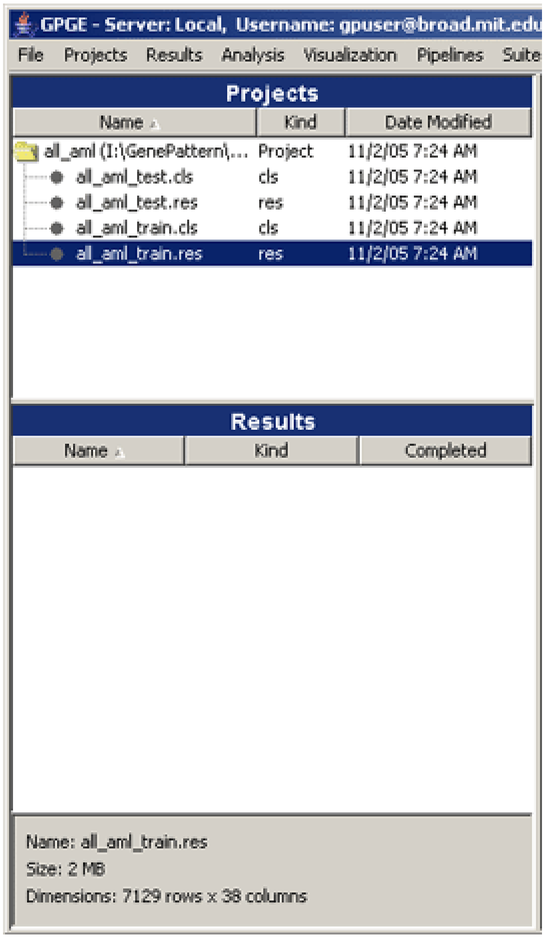
- The next time you start up the GPGE, the **`all_aml`** directory (and any other directories you have added) will be loaded.
- If you add the same directory twice, you will see a message that the directory has already been added.

Exercise 1b: Using the Object Browser

Projects - This section includes all of your data files in the local folder(s) you added in the previous exercise.

Note the field next to the name where you can see the full path to the folder you have loaded.

File Information Window - When a file is selected, this window displays vital statistics about files you are using.



Name	Kind	Date Modified
all_aml (I:\GenePattern\...	Project	11/2/05 7:24 AM
all_aml_test.cls	cls	11/2/05 7:24 AM
all_aml_test.res	res	11/2/05 7:24 AM
all_aml_train.cls	cls	11/2/05 7:24 AM
all_aml_train.res	res	11/2/05 7:24 AM

Name	Kind	Completed

Name: all_aml_train.res
Size: 2 MB
Dimensions: 7129 rows x 38 columns

The fields **`Name, Kind, Date Modified`** in the **`Projects`** pane, and the fields **`Name, Kind, Completed`** in the **`Results`** pane, allow you to sort the contents of the folders based on those properties. When sorting by "kind", GenePattern sorts by file extension. The exception is odf files which have "models" associated with them. GenePattern knows about the following kinds of odf files:

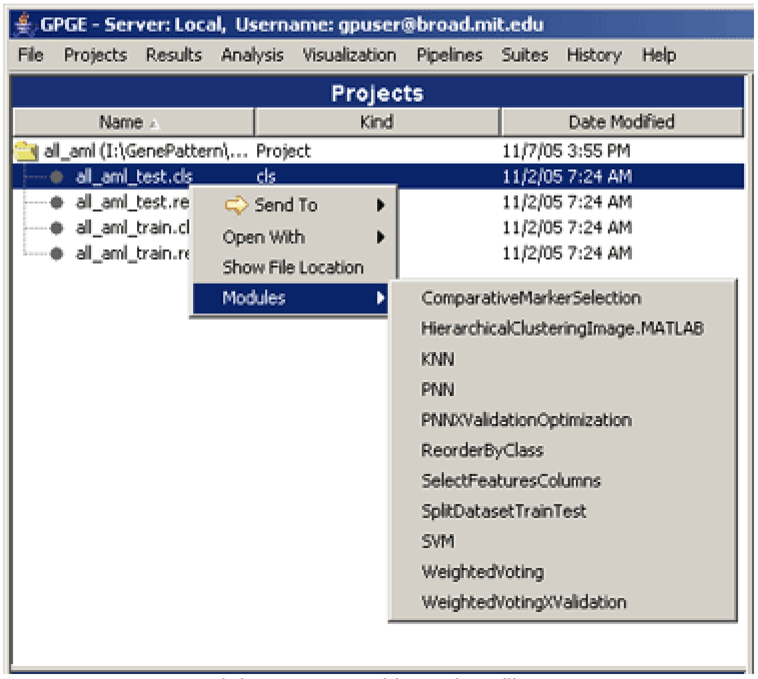
- **`Gene List`** - lists of genes or markers and associated descriptions and significance values resulting from marker selection algorithms
- **`PredictionResults`** - results of a prediction algorithm
- **`SOMCluster`** - results of SOM clustering
- **`ComparativeMarkerSelection`** - results of Comparative MarkerSelection

Files that are sorted based on their extension include the following kinds:

- **`cls`** - Class vector files contain labels for samples in a dataset. For example, a class file might define which samples are normal and which are tumor, or which correspond with patients who survived or didn't survive 5 years after treatment.
- **`res, gct`** - Expression dataset files; each dataset is a combination of chip scans that has been put together so they can be analyzed as a group.
- **`HTML`** - Web documents.
- **`tiff, jpg, png, or gif`** - Graphic image files created by a visualization tool.

- For descriptions of the file formats supported by GenePattern, see GenePattern File Formats.

File types are used by GenePattern to make suggestions for modules which accept a particular type. You can right-click (Mac users Option-click) a file and choose **`Modules`**. (Alternatively, click the file and select **`Projects->Modules`**.) This will give you a list of modules which can be used for that file. If you select a module, it appears in the module pane with that file in the appropriate file box.


GPGE - Server: Local, Username: gpuser@broad.mit.edu
File Projects Results Analysis Visualization Pipelines Suites History Help
Projects
Name Kind Date Modified
all_aml (I:\GenePattern\,... Project 11/7/05 3:55 PM
all_aml_test.cls cls 11/2/05 7:24 AM
all_aml_test.re 11/2/05 7:24 AM
all_aml_train.cl 11/2/05 7:24 AM
all_aml_train.re 11/2/05 7:24 AM
Send To
Open With
Show File Location
Modules
ComparativeMarkerSelection
HierarchicalClusteringImage.MATLAB
KNN
PNN
PNNXValidationOptimization
ReorderByClass
SelectFeaturesColumns
SplitDatasetTrainTest
SVM
WeightedVoting
WeightedVotingXValidation
Modules suggested based on file type

It is important to note that if a module does not appear here it may be because it has not yet been annotated, not because it can't be used. If you are interested in a particular module you can still access it manually from the **`Analysis`** menu.

Viewing information about a file

There are three ways to view information about a file:

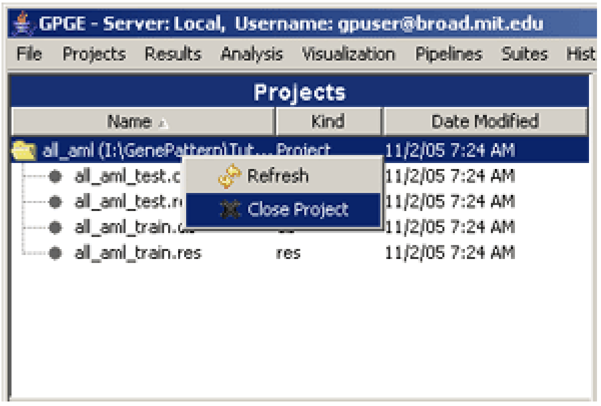
1. *File Information Window*. This window (see figure above) summarizes information about a file, such as what type of data this file contains, its size, and other relevant parameters specific to that data type. To view information about a file in this window, single-click on its filename in the Object Browser.
2. *Text Viewer*. The Text Viewer displays the raw contents of a file. If you double-click on a file in the Object Browser, you will see its contents appear in a window. You can cut and paste the contents of this window. However, you can't modify a file using the Text Viewer.
3. *Data-specific browser*. If the file you double-click is a type for which your browser has an application enabled, it will open in that viewer.

Exercise 1c: Removing a project directory

To remove a project directory from the Object Browser:

- Right-click (Mac users Option-click) the project directory that you want to remove.
- Select **`Close Project`**.

(Alternatively, click the project directory and select **`Projects->Close Project`**.) The directory has now been closed and will not be reloaded the next time you launch GenePattern.

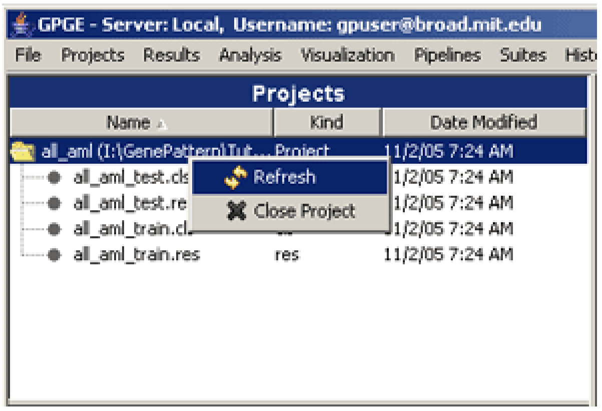

GPGE - Server: Local, Username: gpuser@broad.mit.edu
File Projects Results Analysis Visualization Pipelines Suites Hist
Projects
Name Kind Date Modified
all_aml (I:\GenePattern\Tut... Project 11/2/05 7:24 AM
all_aml_test.c 11/2/05 7:24 AM
all_aml_test.r 11/2/05 7:24 AM
all_aml_train.c 11/2/05 7:24 AM
all_aml_train.res res 11/2/05 7:24 AM
Refresh
Close Project

Exercise 1d: Refreshing a project directory

The GenePattern client periodically reads the contents of your project directories and updates the Object Browser to reflect any changes in them. However, if you have changed the files in a project directory and you want them to appear in the Object Browser, you can manually refresh the directory by doing the following:

- Right-click (Mac users Option-click) the project directory that you want to refresh.
- Select **`Refresh`**. The changes you have made will be reflected in the Object Browser.

(Alternatively, click the project directory and select **`Projects->Refresh`**.)


GPGE - Server: Local, Username: gpuser@broad.mit.edu
File Projects Results Analysis Visualization Pipelines Suites Hist
Projects
Name
Kind
Date Modified
all_aml (I:\GenePattern\Tut... Project 11/2/05 7:24 AM
all_aml_test.cls
all_aml_test.re
all_aml_train.cl
all_aml_train.res res 11/2/05 7:24 AM
Refresh
Close Project

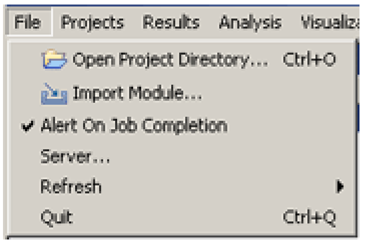
The GenePattern Menu Bar

The following exercises will demonstrate the other functions available in the GenePattern Graphical Environment.

`File`-> **`Open Project Dir`** - Add a project directory to the Object Browser. This function is demonstrated in Exercise 1a.

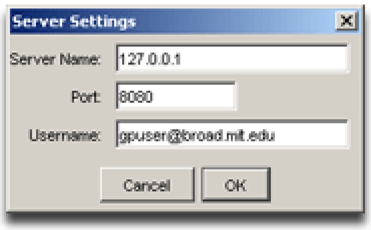
`File->Import Module` - Imports an analysis module, a visualization module, or a pipeline module from a zip file.

`File->Alert On Job Completion` - If this line is checked a window will pop up to announce that your analysis task(s) have completed. This feature can be toggled on and off.


File Projects Results Analysis Visualiz
Open Project Directory... Ctrl+O
Import Module...
✓ Alert On Job Completion
Server...
Refresh
Quit Ctrl+Q

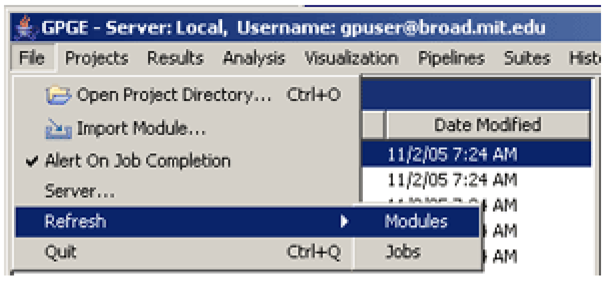
`File->Server` - Allows you to view or change the server on which your jobs are running. If you want to change the server you are accessing, you can change it here by doing the following:

- Select **`File->Server`**.
- Enter the URL and port of of the GenePattern server to which you want to connect.
- Click **`OK`**.
- Continue working. You do not need to restart your client.



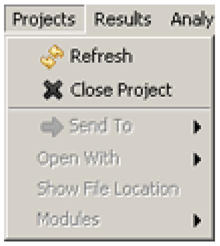
Server Name: 127.0.0.1
Port: 8080
Username: gpuser@broad.mit.edu

`File->Refresh` - Updates the GenePattern Graphical Environment to reflect changes that have been made on the GenePattern server. If modules have been imported or removed, refresh Modules; if jobs have been deleted, refresh Jobs.


GPGE - Server: Local, Username: gpuser@broad.mit.edu
File Projects Results Analysis Visualization Pipelines Suites Hist
Open Project Directory... Ctrl+O
Import Module...
✓ Alert On Job Completion
Server...
Refresh
Quit Ctrl+Q
Modules
Jobs
Date Modified
11/2/05 7:24 AM
11/2/05 7:24 AM

`File->Quit` - Quits the program.

`Projects->Refresh` will refresh a selected project directory. An alternative method was demonstrated in Exercise 1d.



`Projects->Close Project` will remove a selected project directory from the GenePattern Object Browser. An alternative method was demonstrated in Exercise 1c.

The remainder of the menu will become active once your work in GenePattern has progressed further. We will detail these functions then.

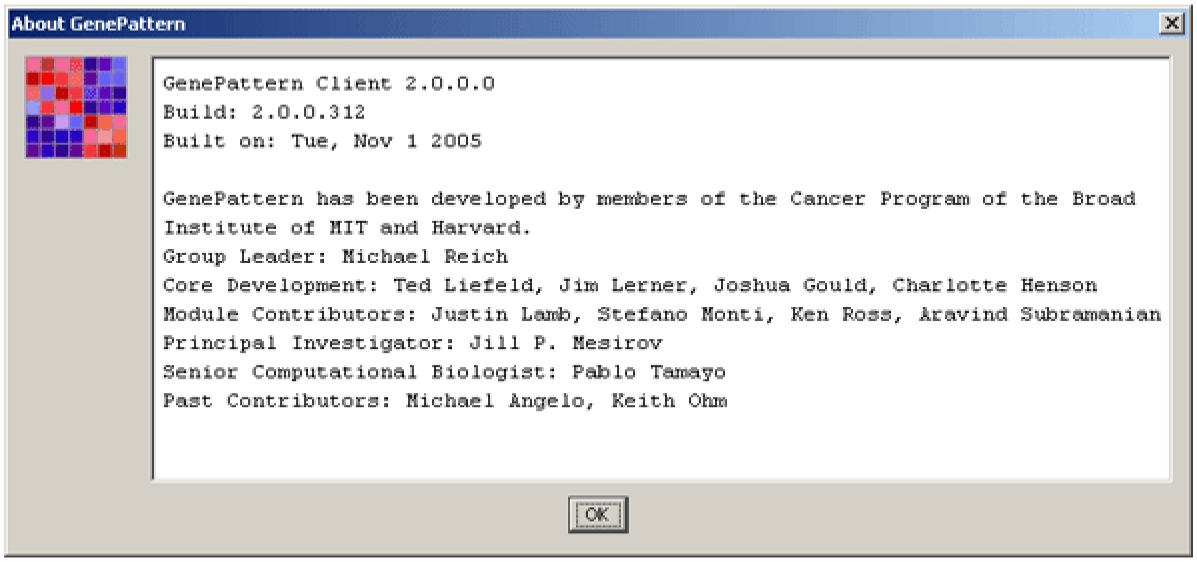
The **`Results`** menu will become active once your work in GenePattern has progressed further. You will be able to delete individual jobs, or all your GenePattern jobs here. You will be able to terminate individual jobs which are running.

The **`Analysis`**, **`Visualization`**, and **`Pipelines`** menus are the most important part of the graphical environment. This is where you will do analysis and visualization, covered in detail in later exercises.

The **`Suites`** menu allows you to group modules into packages of modules that have similar functionality, such as clustering or proteomics. You can then work with the suite, for example, you can install all modules in the "clustering" suite or view only those modules in the "proteomics" suite. Suites are covered in detail in later exercises.

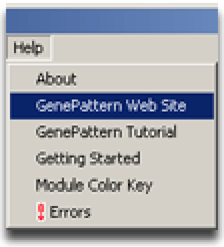
The **`History`** menu lists the last 10 jobs you have run. When you select a job from this list you will reload it as it was when you launched it. **`History->View All`** will bring up a window of all modules you have run. From this window you can relaunch jobs that were launched further back in your history, or purge jobs to delete them from your server. If you select all of the jobs you can purge all jobs here.

`Help->About GenePattern`. This contains basic authorship text and the build number of this version of the GPGE - useful in bug reports.



```
GenePattern Client 2.0.0.0
Build: 2.0.0.312
Built on: Tue, Nov 1 2005

GenePattern has been developed by members of the Cancer Program of the Broad
Institute of MIT and Harvard.
Group Leader: Michael Reich
Core Development: Ted Liefeld, Jim Lerner, Joshua Gould, Charlotte Henson
Module Contributors: Justin Lamb, Stefano Monti, Ken Ross, Aravind Subramanian
Principal Investigator: Jill P. Mesirov
Senior Computational Biologist: Pablo Tamayo
Past Contributors: Michael Angelo, Keith Ohm
```

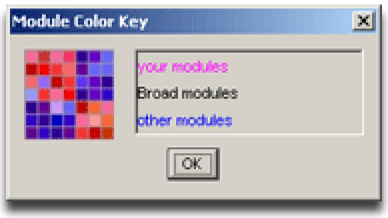


`Help->GenePattern Web Site` will take you to the public GenePattern web site. You must be connected to the internet to use this feature.

`Help->GenePattern Tutorial` will take you to the online latest version of this document. You must be connected to the internet to use this feature.

`Help->GenePattern Server` will open the GenePattern Web Client.

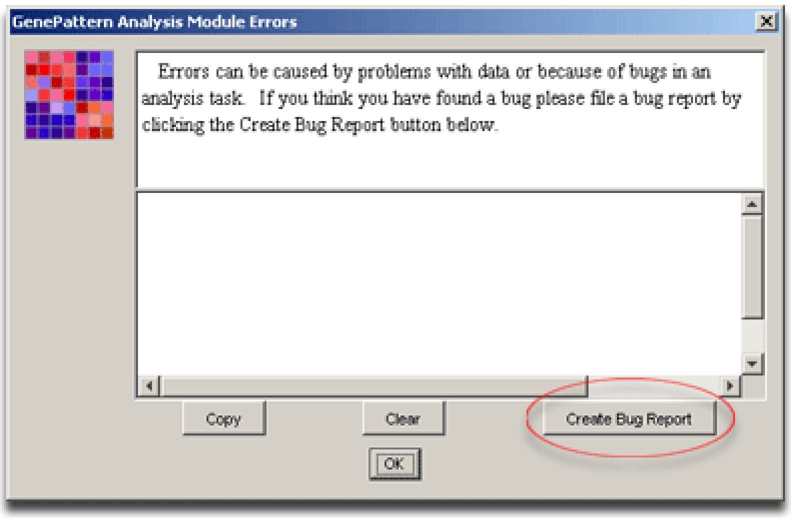
`Help->Getting Started` will display the welcome message in your graphical environment.



`Help->Module Color Key`. GenePattern modules are listed by name, with color codes which indicate who authored the module. This element will bring up a cheat-sheet to remind you what the colors mean.

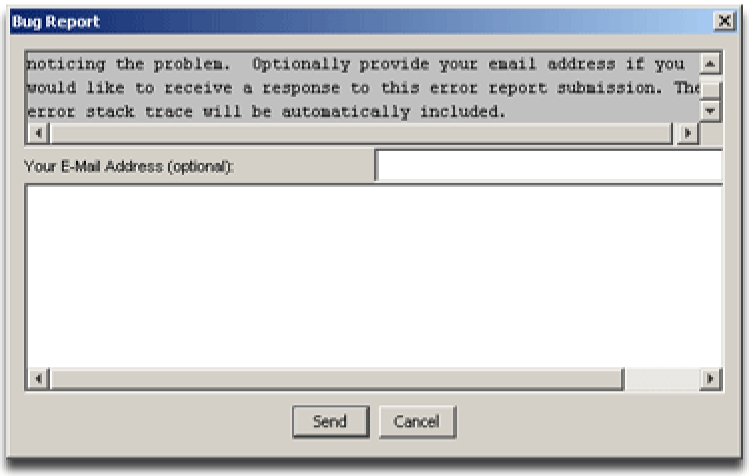
`Help->Errors` will allow you to send a bug report to the GenePattern Help Desk.

- Select the **`Help->Errors`** menu item.
- In the GenePattern Analysis Module Errors window, click **`Create Bug Report`**.



Errors can be caused by problems with data or because of bugs in an analysis task. If you think you have found a bug please file a bug report by clicking the Create Bug Report button below.

- You will see a dialog box prompting you for the details of the bug.
- Enter as much information as possible, including the text of any error messages.
- Click **`Send`** to send the report to the GenePattern help desk.



```
noticing the problem.  Optionally provide your email address if you
would like to receive a response to this error report submission. The
error stack trace will be automatically included.
```

Your E-Mail Address (optional):

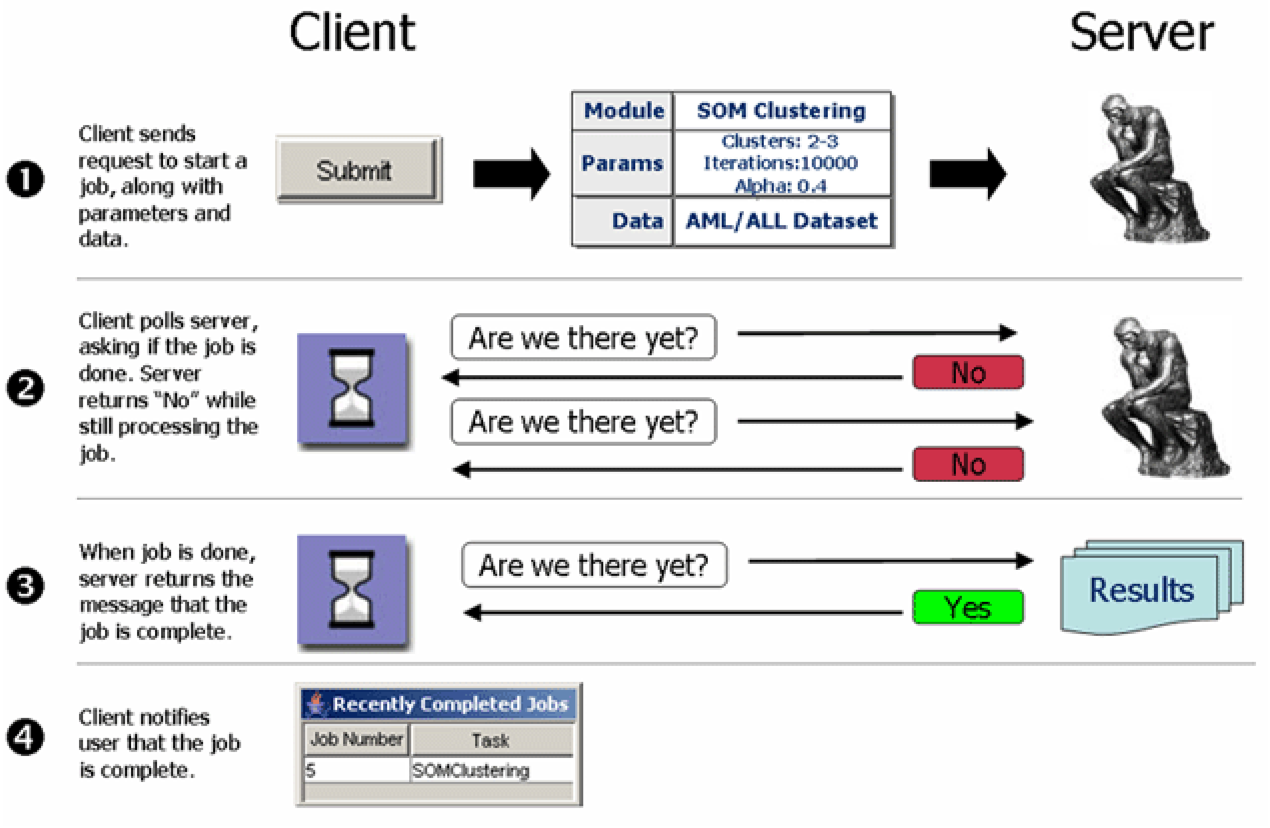
! You must be connected to the Internet to use this feature.

Data Analysis with the GenePattern Graphical Environment

The GenePattern client/server architecture is designed for flexibility and ease of use. There are several important points about how GenePattern runs analyses:

- All modules - algorithms, visualizers, and pipelines - are stored on the server.
- When you launch the client it gets the list of available modules from the server.
- To use a module, select it, fill in necessary parameters, and click **`Run`**.
- The request is sent to the server, where the algorithm runs on the data you have given it.
- While the server is doing the analysis the client will poll the server periodically to see if the module is done yet.
- When the module is complete, the server returns "yes", indicating that the results are available.

A figure illustrating client/server interaction is shown below:

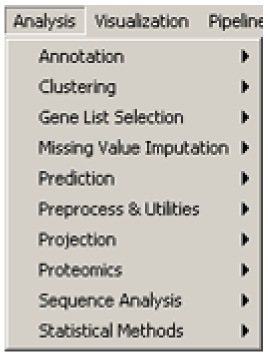

Client
Server
1
Client sends request to start a job, along with parameters and data.
Submit
Module SOM Clustering
Params Clusters: 2-3 Iterations:10000 Alpha: 0.4
Data AML/ALL Dataset
2
Client polls server, asking if the job is done. Server returns "No" while still processing the job.
Are we there yet?
No
Are we there yet?
No
3
When job is done, server returns the message that the job is complete.
Are we there yet?
Yes
Results
4
Client notifies user that the job is complete.
Recently Completed Jobs
Job Number Task
5 SOMClustering

Anatomy of an Analysis

The result files are stored on the server; however, server storage is temporary and result files are deleted after they have been on the server for a certain length of time (the default is one week). To save your result files permanently, you must download them. You will learn how to do this in Exercise 2c.

Analysis Modules

Getting Started:
Click on **`Analysis`**. You will see a list of task categories, which include:



- **Annotation** - Retrieve additional data about microarray identifiers.
- **Clustering** - Unsupervised learning. These algorithms find inherent patterns in data with no prior knowledge of the structure of the data.
- **GeneListSelection** - Algorithms that return lists of genes. These include marker selection algorithms such as finding the nearest neighbors of a gene or expression profile.
- **Missing Value Imputation** - Algorithms for filling in data missing in a dataset.
- **Prediction** - Supervised Learning. These algorithms learn to distinguish between classes of data by being presented with examples from each class. They use these examples to build models that can predict, given data of an unknown class, to which class data belongs.
- **Preprocess and Utilities** - Basic operations such as filtering, creating a subset of genes or samples, converting probe sets from one chip type to another.
- **Projection** - Algorithms to project data into a space with a differing (usually lower) number of dimensions.
- **Proteomics** - Algorithms for proteomics analysis that are run on the set of input spectra, including quality assessment, normalization, peak detection, and peak matching.
- **Sequence Analysis** - modules intended for use on sequence data.
- **Statistical Methods** - statistical tests such as measures of fit, correlation, and significance.

Running an Analysis

In this section, you learn how to submit an analysis job.

Exercise 2a: Submitting an analysis job

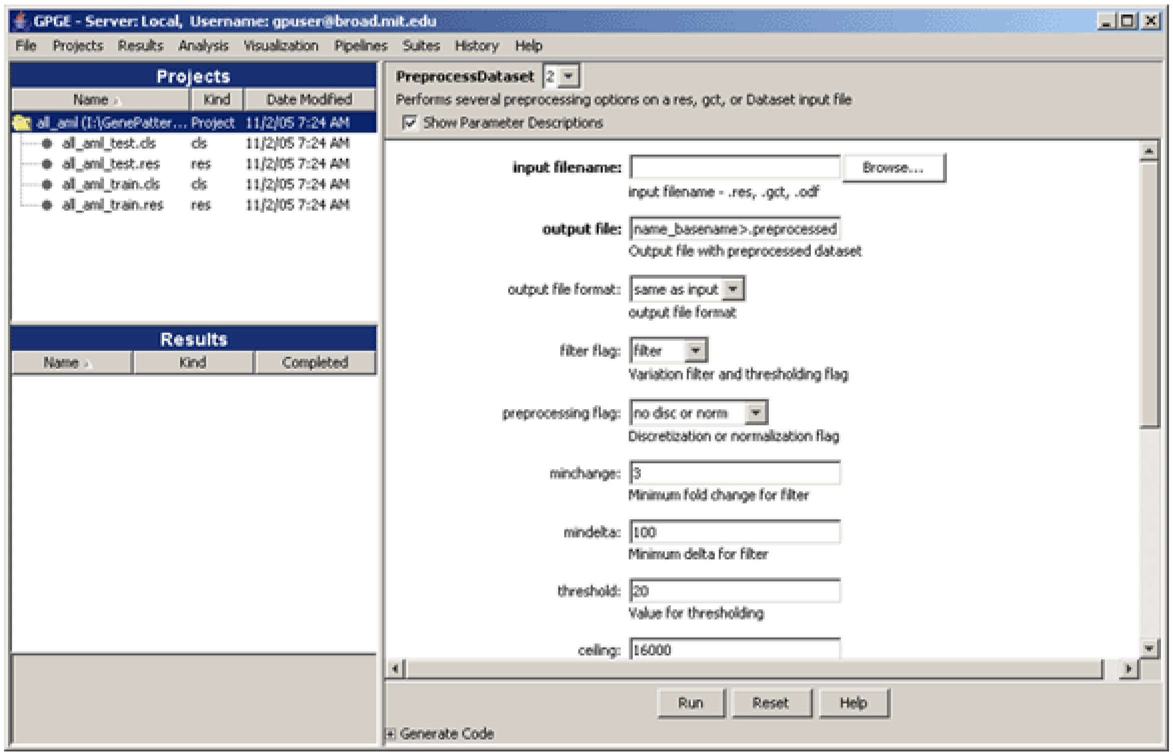
The data we will use in the exercise is from Golub and Slonim *et al.*, which used clustering and prediction algorithms to find genes that distinguish between two subtypes of leukemia, ALL and AML.

We will use the following analysis steps as an exercise:

1. Preprocess expression data containing ALL and AML samples
2. Run a SOM clustering algorithm on the resulting dataset
3. View the results in a SOM Cluster viewer
4. Run a marker selection algorithm to find genes that best distinguish between ALL and AML

Analysis Step 1. Preprocess the expression dataset

- Make sure you have added the directory with the **`all_aml`** data.
- Choose **`Analysis->Preprocess & Utilities->PreprocessDataset`**.



Name	Kind	Date Modified
all_aml (I:\GenePatter...	Project	11/2/05 7:24 AM
all_aml_test.cls	cls	11/2/05 7:24 AM
all_aml_test.res	res	11/2/05 7:24 AM
all_aml_train.cls	cls	11/2/05 7:24 AM
all_aml_train.res	res	11/2/05 7:24 AM

Name	Kind	Completed

PreprocessDataset 2
Performs several preprocessing options on a res, gct, or Dataset input file
☑ Show Parameter Descriptions

input filename:
input filename - .res, .gct, .odf

output file: name_basename>.preprocessed
Output file with preprocessed dataset

output file format: same as input
output file format

filter flag: filter
Variation filter and thresholding flag

preprocessing flag: no disc or norm
Discretization or normalization flag

minchange: 3
Minimum fold change for filter

mindelta: 100
Minimum delta for filter

threshold: 20
Value for thresholding

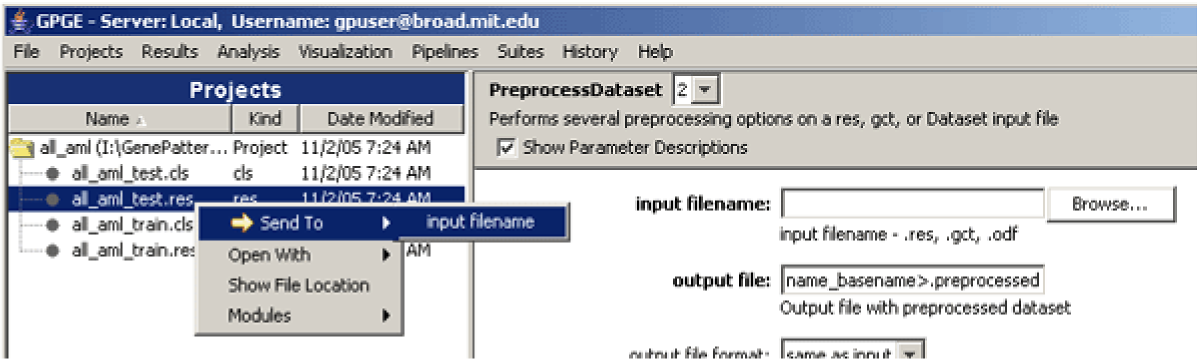
ceiling: 16000

Notice the layout of the module. This is the format for all modules:

- Each line starts with the name of the parameter.
- The box is where you fill in the parameter values.
- Parameters can be free text, multiple choice from a drop-down menu, or files.
- Below each parameter is a more complete description of that parameter. These descriptions may be hidden by unchecking the **`Show Parameter Descriptions`** box in the upper left corner.
- Required parameters are listed in **bold**. Other parameters are optional.
- At the bottom of the module pane are three buttons. **`Run`** will launch the module. **`Reset`** will restore the parameters to their default values. **`Help`** will display the documentation for that module.

To perform the analysis, do the following:

- From your **all_aml** project directory, right-click (Mac users Option-click) **all_aml_train.res** and select **Send To** and then select "input filename" to send the file to the input filename box.


GPGE - Server: Local, Username: gpuser@broad.mit.edu
File Projects Results Analysis Visualization Pipelines Suites History Help
Projects
Name Kind Date Modified
all_aml (I:\GenePatter... Project 11/2/05 7:24 AM
all_aml_test.cls cls 11/2/05 7:24 AM
all_aml_test.res res 11/2/05 7:24 AM
all_aml_train.cls
all_aml_train.res
Send To
input filename
Open With
Show File Location
Modules
PreprocessDataset 2
Performs several preprocessing options on a res, gct, or Dataset input file
Show Parameter Descriptions
input filename:
Browse...
input filename - .res, .gct, .odf
output file:
name_basename>.preprocessed
Output file with preprocessed dataset
output file format:
same as input

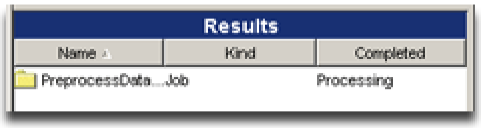
The **input filename** parameter now displays the filename. (Alternatively, you can click the file and select **Projects->Send To**, or click and drag the file to the **input filename** parameter box. Using **Send To** has the benefit of presenting you with only those input boxes that accept the filetype you have selected.)

- The **output file** parameter contains a default name for the result file that will be created. This is <input.filename_basename>.preprocessed, meaning that the output file will have the same name as the input file, with the text ".preprocessed" appended to it. If you want a different name for the output file, you can enter it here.
- For the **output file format** parameter, select "GCT".
- For the **preprocessing flag**, choose "row normalization".
- To start the analysis, click **Run**.

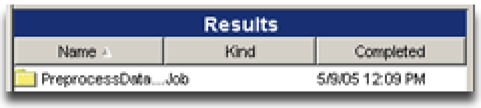
Checking Job Status

Once you submit a job, there are two ways you can know its status:

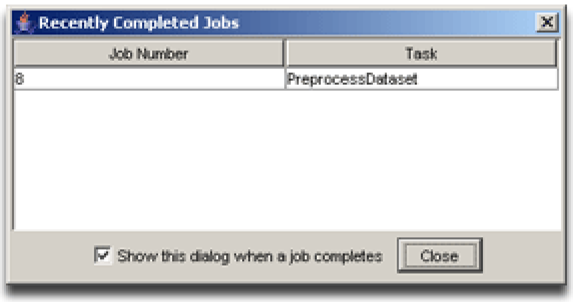
- Within the **Results** pane on the lower left hand side, you will see messages informing you that your job is **Processing**.



When the task has completed the **Completed** column lists the date and time that the task ended.



- Alternatively, once your job is complete, you will see the dialog box shown below. This window will accumulate job results until you close it. This is useful when you have several jobs running simultaneously or when you want a log of recent jobs.

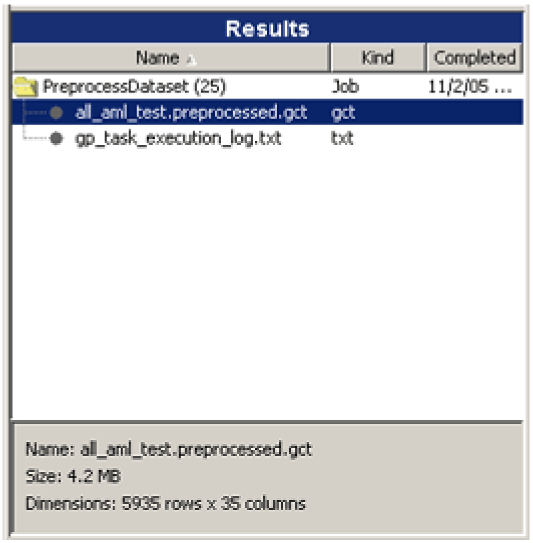


- If you do not want this window to appear after a job completes, uncheck **`Alert On Job Completion`** under your **`File`** menu.
- To bring it back again, select it again. The check mark will toggle on and off.

Analysis Results

The files created by the analysis are stored on the server and listed in the **`Results`** pane. To view them or use them in further analysis, do the following:

- In the **`Results`** pane, double-click on the **`PreprocessDataset`** node. You will see a list of all the results from your **`PreprocessDataset`** task.



Name	Kind	Completed
PreprocessDataset (25)	Job	11/2/05 ...
all_aml_test.preprocessed.gct	gct	
gp_task_execution_log.txt	txt	

Name: all_aml_test.preprocessed.gct
Size: 4.2 MB
Dimensions: 5935 rows x 35 columns

You'll notice that in addition to the output file you created with the **`PreprocessDataset`** task, you also have a "txt" file called **`gp_task_execution_log.txt`**. This file contains all of the parameter values you would need to repeat the task you just completed. This file is useful for reproducing research results.

In addition to execution log files, you will often see other text files in your analysis results; for example, **`stdout`** and **`stderr`**.

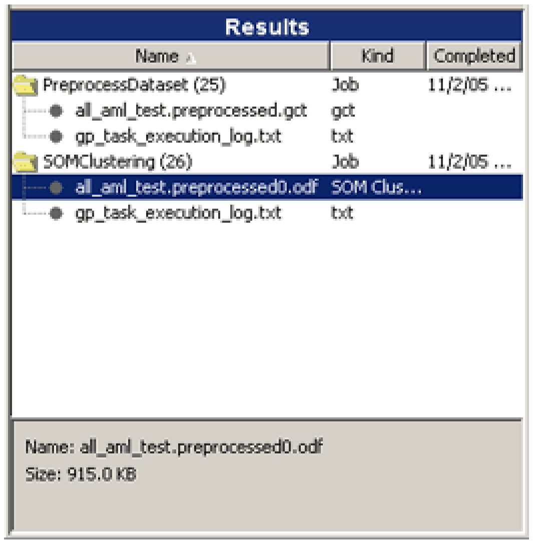
- Files named **`stdout`** contain "standard output" messages; that is, comments generated as the analysis module runs.
- Files named **`stderr`** contain "standard error" messages; that is, information about errors (if any) that occurred during the analysis.

Exercise 2b: Using Results in Further Analyses

Analysis Step 2: SOM Clustering

Now that we have filtered the dataset we will analyze it using the *Self-Organizing Maps (SOM)* algorithm:

- Select **`Analysis->Clustering->SOMClustering`**.
- For the **`cluster range`** parameter, enter **`2`**. In this example, we know that there are 2 classes in the data, ALL and AML, so entering this value will show how well the SOM algorithm reproduces the distinction.
- If you haven't already, open the **`PreprocessDataset`** node.
- Click and drag **`all_aml_train.preprocessed.gct`** to the **`dataset filename`** box.
- Change **`cluster by`** from rows to columns. This step will allow us to cluster samples rather than genes in our SOM cluster algorithm.
- Click **`Run`**.
- When the job finished dialog box appears, the job is complete.
- Open the **`SOMClustering`** node. The SOM result file will be there, as shown in the image below.



Name	Kind	Completed
PreprocessDataset (25)	Job	11/2/05 ...
all_aml_test.preprocessed.gct	gct	
gp_task_execution_log.txt	txt	
SOMClustering (26)	Job	11/2/05 ...
all_aml_test.preprocessed0.odf	SOM Clus...	
gp_task_execution_log.txt	txt	

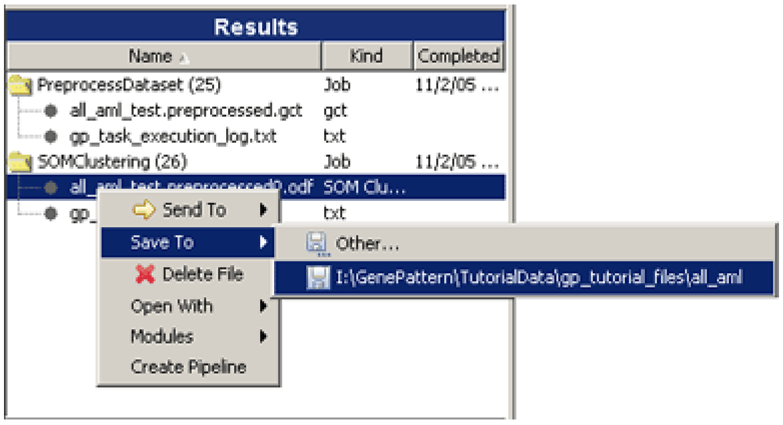
Name: all_aml_test.preprocessed0.odf
Size: 915.0 KB

Exercise 2c: Saving Result Files Locally

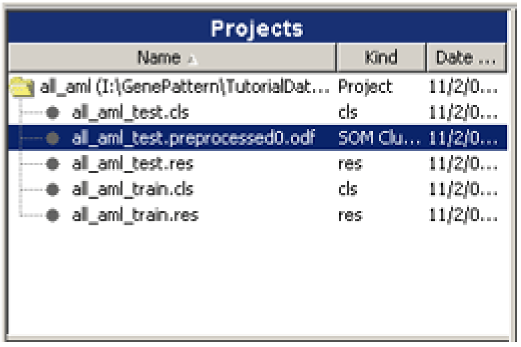
GenePattern modules create result files that are stored on the server. However, the server storage is temporary and result files are deleted after they have been on the server for a certain length of time (the default is one week). To save result files permanently:

- Open the **`SOMClustering`** node.
- Right-click (Mac users Option-click) the cluster result file.
- Select **`Save To`** and then the desired location.
- You will see the available options, as shown below, prompting you to select a location where the file will be saved.
- Save the file in your desired location, in this case, the **`all_aml`** directory.

(Alternatively, click the result file and select **`Results->Save to`**.)



The file will appear in your project directory node.



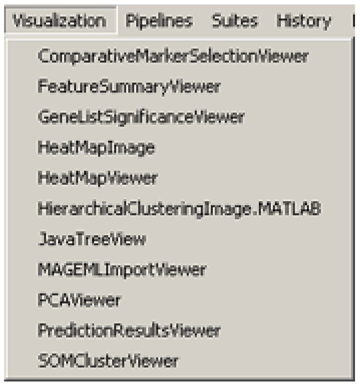
Name	Kind	Date ...
all_aml (I:\GenePattern\TutorialDat...	Project	11/2/0...
all_aml_test.cls	cls	11/2/0...
all_aml_test.preprocessed0.odf	SOM Clu...	11/2/0...
all_aml_test.res	res	11/2/0...
all_aml_train.cls	cls	11/2/0...
all_aml_train.res	res	11/2/0...

Viewing Analysis Results in the GenePattern Graphical Environment

In GenePattern, the process of viewing data is identical to the process of analyzing data. The `Visualization` menu displays the modules that allow you to view and manipulate your data and to create graphics that can be viewed in other applications.

There are two types of visualizer:

- **Viewer** - a module that lets you view your data graphically and to interactively manipulate the view. By convention these are named with "Viewer" in the name.
- **Image creator** - a module that creates static graphics for view in other applications. These include modules to generate GIF, JPEG or other graphic files, PDF files, etc. By convention these are named with "Image" in the name.



Following are brief descriptions of the visualizers currently available. As new visualization tools are released, they are added to the Visualization menu; therefore, your Visualization menu may include additional tools.

- **ComparativeMarkerSelectionViewer** - Displays the results from ComparativeMarkerSelection and permits filtering and extraction of Feature Lists.
- **FeatureSummaryViewer** - Displays in table format, features used to build models in cross-validation.
- **GeneListSignificanceViewer** - Displays the results of marker selection algorithms such as ClassNeighbors and GeneNeighbors in the format of a gene lis t with associated significance values.
- **HeatMap Image** - Creates a graphic of a heat map in one of several formats. This module is useful when you want to create a graphic without having to do it manually using the HeatMapViewer UI, for example as a step in a pipeline.
- **HeatMapViewer** - Displays a dataset in a heat map format, graphically showing the up- and down-regulation of genes.
- **HierarchicalClusterViewer** - Displays results of hierarchical clustering in a 2-dimensional biclustered-tree format.
- **JavaTreeView** - Displays a hierarchical clustering viewer that reads in Eisen's cdt, atr, and gtr files.
- **MAGEMLImportViewer** - Takes a zip file containing MAGE-ML files and imports the data into GenePattern. MAGE-ML is a standard for the representation of microarray expression data.
- **PCAViewer** - Displays a 3D representation of PCA analysis results. This viewer is adapted from the PCA viewer in TIGR's Multi-experiment Viewer package.
- **PredictionResultsViewer** - Displays results of prediction (supervised learning) algorithms (eg. KNN, Weighted Voting), such as true versus predicted class, confidence level, etc.
- **SOMClusterViewer** - Displays results and permits navigation of SOM clusters.

Exercise 2d: Viewing Clustering Results

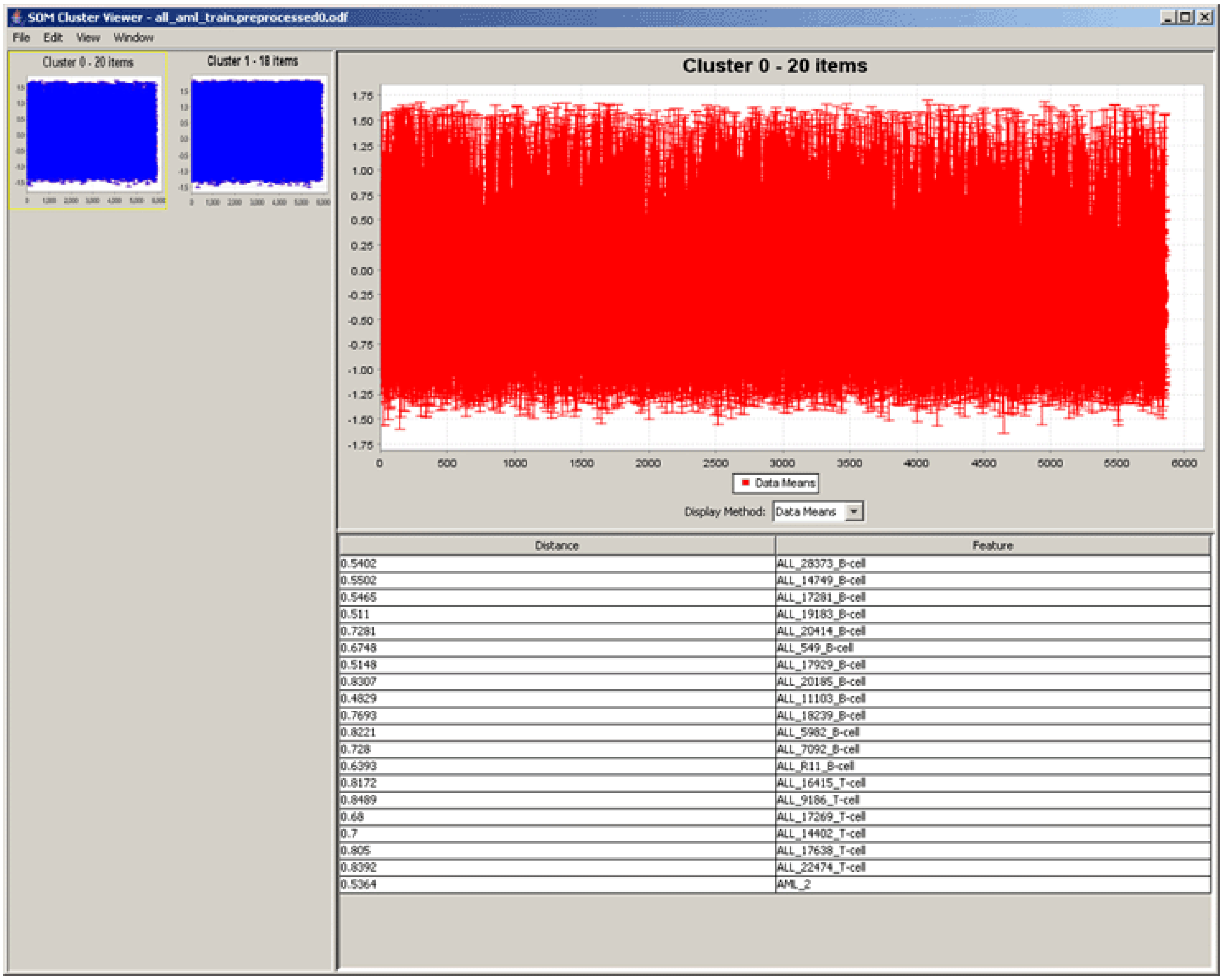
Analysis Step 3. Viewing SOM clustering results

To view the result file you just created:

- Select `Visualization->SOMClusterViewer`.
- Click and drag the file created in the `SOMCluster` node to the "som cluster filename" box.
- Click `Run`.

You will see the *SOM Cluster Viewer*, shown below, displaying the two clusters from the SOM analysis.

- Click on a thumbnail cluster profile on the left pane to view the contents of each cluster.



Distance	Feature
0.5402	ALL_28373_B-cell
0.5502	ALL_14749_B-cell
0.5465	ALL_17281_B-cell
0.511	ALL_19183_B-cell
0.7281	ALL_20414_B-cell
0.6748	ALL_549_B-cell
0.5148	ALL_17929_B-cell
0.8307	ALL_20185_B-cell
0.4829	ALL_11103_B-cell
0.7693	ALL_18239_B-cell
0.8221	ALL_5982_B-cell
0.728	ALL_7092_B-cell
0.6393	ALL_R11_B-cell
0.8172	ALL_16415_T-cell
0.8489	ALL_9186_T-cell
0.68	ALL_17269_T-cell
0.7	ALL_14402_T-cell
0.805	ALL_17638_T-cell
0.8392	ALL_22474_T-cell
0.5364	AML_2

Exercise 2e: Marker Selection in GenePattern

Now we will continue the analysis to use this data to find the genes that best distinguish AML from ALL samples. To do this, we use the *ComparativeMarkerSelection* module.

Analysis Step 4: Marker Selection

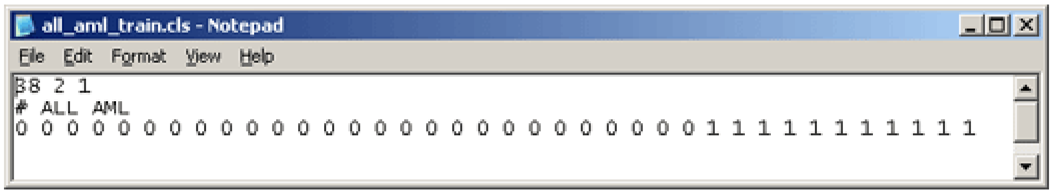
Class Vectors

Marker selection algorithms and prediction algorithms need to know to which class each sample in a dataset belongs. For example, to find the genes that best distinguish between ALL and AML, the algorithm needs to know which samples are ALL and which are AML. In GenePattern, this label information is contained in `Class Vector` (.cls) files.

To view the format of the class vector file for the ALL/AML dataset:

- Select the `all_aml` project directory.
- Double click on `all_aml_train.cls`.

You will see the following screen:



```
38 2 1
# ALL AML
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
```

- The first line indicates that there are 38 samples, representing 2 classes (the "1" is non-functional).
- The second (optional) line names the classes, starting with 0. In this file, class 0 is named ALL, and class 1 is named AML.
- The third line contains the class of each sample. In this file, the first 27 samples are class 0 (ALL) and the next 11 are class 1 (AML).

More information on class vector files is available in the File Formats section.

This section of the tutorial will depart slightly from the published research. We have released a new marker analysis algorithm which is superior to the version used in 1999. The original analysis may still be reproduced using the all.aml.methodology pipeline.

To run the marker analysis algorithm:

- Select **`Analysis->Gene List Selection->ComparativeMarkerSelection`**.

  This algorithm finds the genes whose expression differs most between two classes. In this case the two classes are AML and ALL. Difference is determined by a statistical measure such as a signal to noise (S2N) or t-test.

- Right-click (Mac users Option-click) **`all_aml_train.res`** and **`Send To`** the "input filename" box.
- Right-click (Mac users Option-click) **`all_aml_train.cls`** and **`Send To`** the "cls filename" box.
- Click **`Run`**.

To select the top 50 up regulated and the top 50 down regulated genes:

- Select **`Analysis->Gene List Selection->ExtractComparitiveMarkerResults`**.
- Right-click (Mac users Option-click) the **`all_aml_train_comp.marker.odf`** file and **`Send To`** to the "comparative marker selection filename" box.
- Right-click (Mac users Option-click) the original dataset **`all_aml_train.res`** from the project directory and **`Send To`** to the "dataset filename" box.
- In the **`number of neighbors`** box, enter "50".
- Click on **`Run`**.

Exercise 2f: Visualizing Data

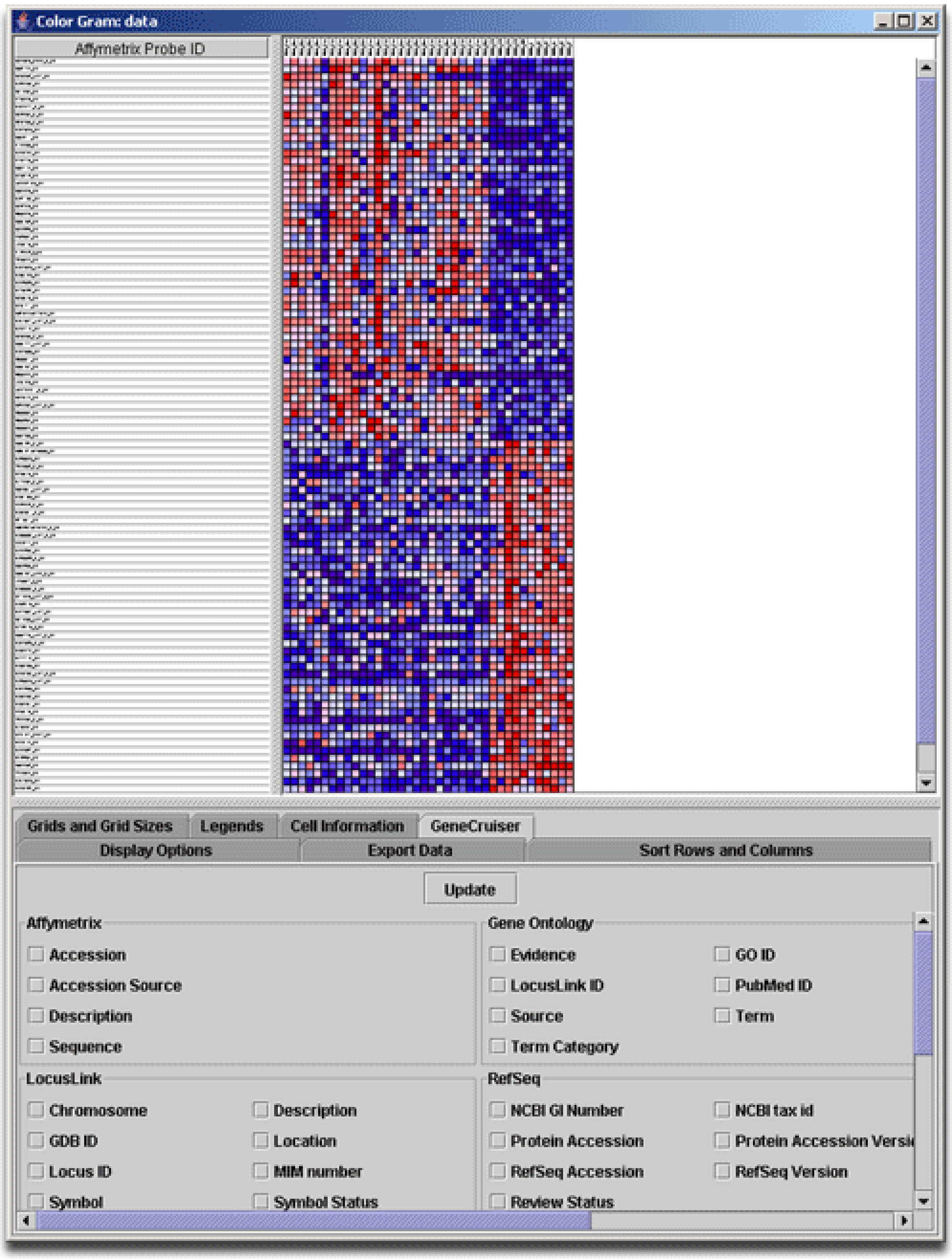
Heat Map Viewer

When the analysis is complete, you will see the **`ExtractComparativeMarkerResults`** node "Completed" date appear in the **`Results`** pane. An intuitive way to visualize these results is in a heat map format, also called a colorgram.

To visualize the ComparativeMarkerSelection results:

- Open the **`ExtractComparativeMarkerResults`** node in the **`Results`** pane.
- Select the **`Visualization->HeatMap Viewer`** menu item.
- Right-click (Mac users Option-click) **`all_aml_train.comp.marker.filt.res`** and **`Send To`** to the **`filename`** input box.
- Click **`Run`**.

You will see the following screen:



This visualizer has many features, but the most important ones to notice are:

- Upregulated values are in red, downregulated in blue.
- Samples are listed along the top row, genes are down the left side.
- The **`Cell Information`** tab displays the annotation of the gene currently under the cursor.
- The **`Grids and Grid Sizes`** tab allows you to enlarge and reduce the grid size.

As for the data itself, you can see that there are quite a few genes which by visual inspection look as though they do a very good job of separating AML from ALL.

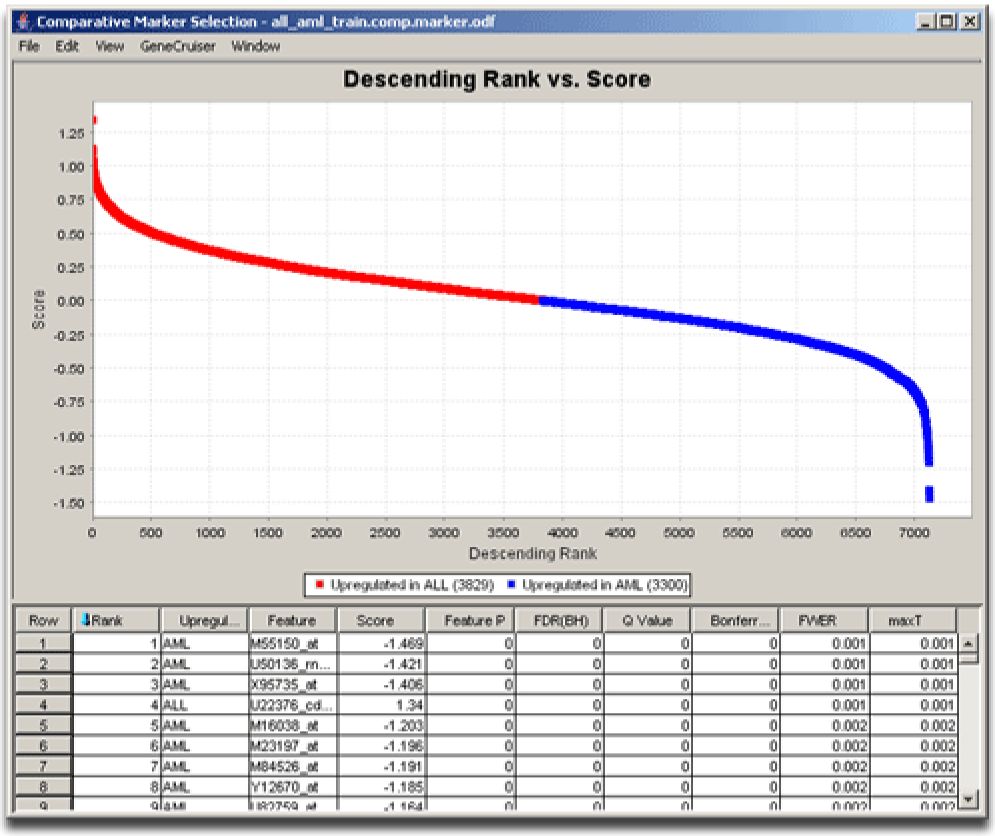
Comparative Marker Selection Viewer

In addition to creating a dataset, **`ComparativeMarkerSelection`** also creates a table of the significance values for the genes that it finds.

To view the genes and their significance values:

- Select the **`Visualization->ComparativeMarkerSelectionViewer`** menu item.
- Open the **`ComparativeMarkerSelection`** node.
- Right-click (Mac users Option-click) **`all_aml_train.comp.marker.odf`** and **`Send To`** the **`input filename`** box.
- Click **`Run`**.

You will see the screen shown below:



Row	Rank	Upregul...	Feature	Score	Feature P	FDR(BH)	Q Value	Bonferr...	FWER	maxT
1	1	AML	M55150_at	-1.469	0	0	0	0	0.001	0.001
2	2	AML	U50136_rn...	-1.421	0	0	0	0	0.001	0.001
3	3	AML	X95735_at	-1.406	0	0	0	0	0.001	0.001
4	4	ALL	U22376_cd...	1.34	0	0	0	0	0.001	0.001
5	5	AML	M16038_at	-1.203	0	0	0	0	0.002	0.002
6	6	AML	M23197_at	-1.196	0	0	0	0	0.002	0.002
7	7	AML	M84526_at	-1.191	0	0	0	0	0.002	0.002
8	8	AML	Y12670_at	-1.185	0	0	0	0	0.002	0.002
9	9	AML	U82759_at	-1.164	0	0	0	0	0.002	0.002

This robust visualizer includes interactive histogram plots of the feature specific p-value, FWER, FDR (BH), Q value, and maxT p-value. There is an interactive plot of the descending test statistic rank versus the test statistic value included, which is useful for visualizing the number of features that have a positive and negative test statistic value in each class. Pair-wise comparison of different significance measures can also be plotted, so as to help the user assess the relative stringency of the hypothesis-rejection criteria selected. You can change filtering criteria in the viewer, then view features that pass the filtering and create new datasets and features lists from these filtered features. All the plots are dynamically updated to include only those filtered features.

Annotation of features is provided by two mechanisms. Affymetrix probe ids can be interactively annotated from genomic databases. You can also enter your own annotations of features and view these annotations with color-coding. Lastly, you can visually inspect the profiles of each feature across each sample. All of these options are outlined in the documentation for the visualizer. Remember that the documentation is always accessible from the **`Help`** button located at the bottom of the module pane.

Exercise 2g: Creating a JPEG Image of a Colorgram/Adding Files From Outside the Object Browser

Visualizers allow you to manipulate data graphically, but you may want to create a graphic image non-interactively, for instance as a step in a pipeline. To do this, you can use an Image Creator module. In this example, you will create a JPEG graphic of the heat map for the data you created in the ComparativeMarkerSelection analysis.

Adding input files from outside the GenePattern environment

As part of this exercise, you will also learn how you can add input files to an analysis without having to click and drag from the Object Browser. The GenePattern client also allows you to click and drag a file from any folder or URL to a file input box.

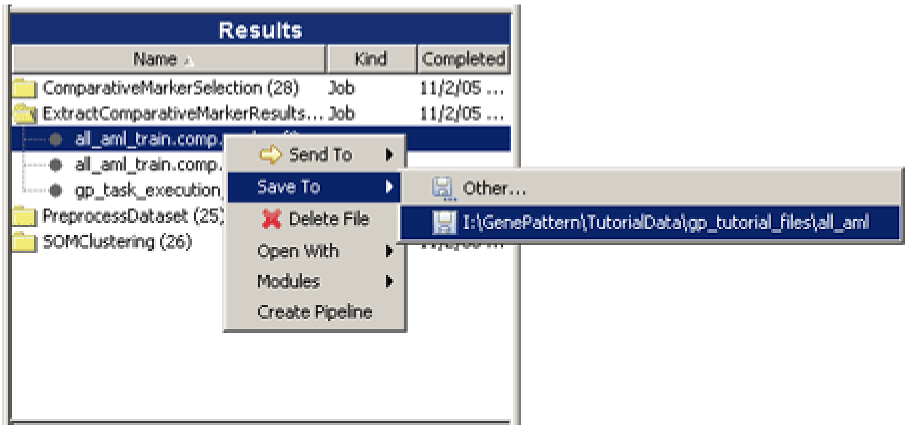
To demonstrate this functionality, you will do the following in this exercise:

1. Save the filtered ComparativeMarkerSelection dataset result file locally.
2. Open the HeatMapImage module.

3. Run the HeatMapImage module using the file you saved to your local folder.

1. Save the filtered ComparativeMarkerSelection dataset result file locally

- Open the **`ExtractComparativeMarkerResults`** node.
- Right-click (Mac users Option-click) **`all_aml_train.comp.marker.filt.res`**.
- Choose **`Save To`** and then the path to your datasets.

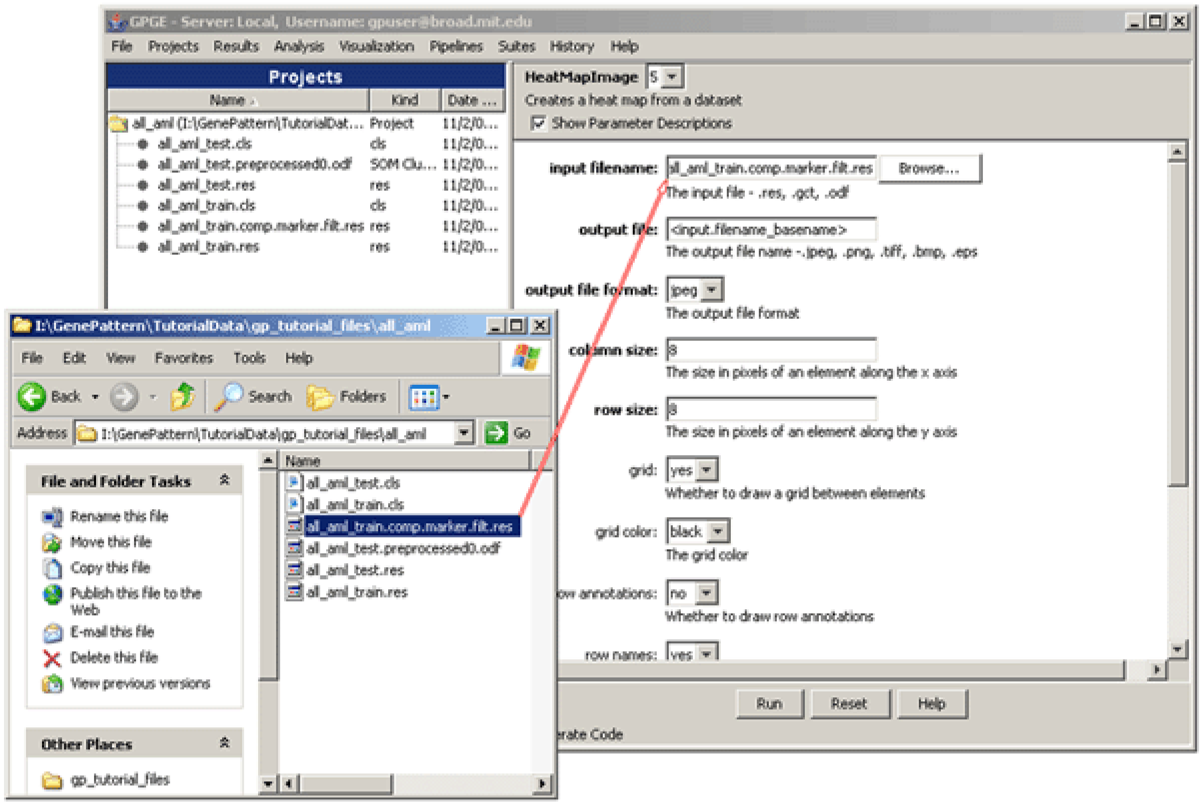


2. Open the HeatMapImage module

- Select the **`Visualization->HeatMapImage`** menu item.

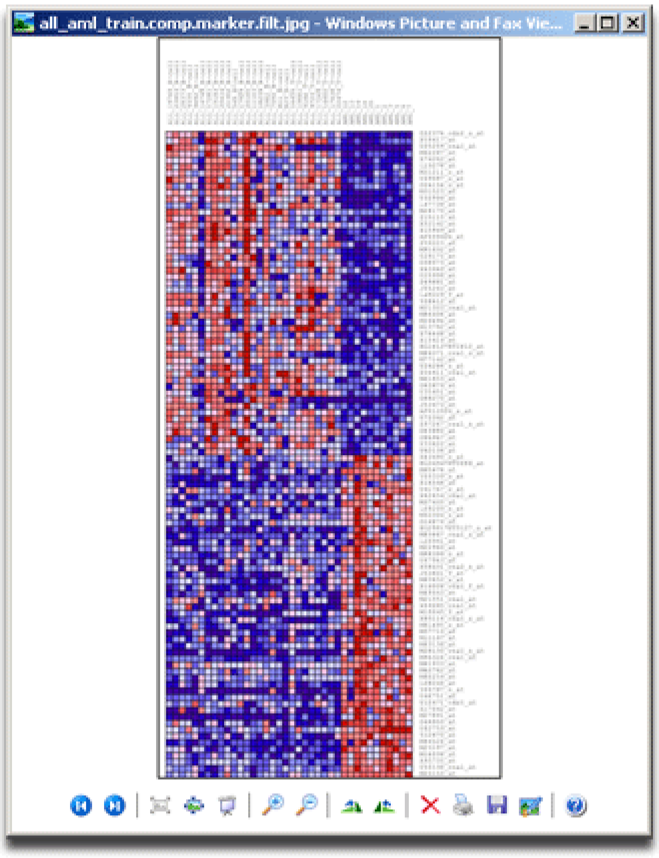
3. Run the HeatMapImage module using the file you saved to your local folder

- Find and open the folder on your hard drive that contains the **`all_aml`** datasets.
- Arrange your windows so you can see both the **`all_aml_train.comp.marker.filt.res`** file icon there and the "file input" box in the HeatMapImage module (see image below).
- Click and drag the file from the all_aml window onto the input box.
- Click **`Run`**.


GPGE - Server: Local, Username: gpuser@broad.mit.edu
File
Projects
Results
Analysis
Visualization
Pipelines
Suites
History
Help
Projects
Name
Kind
Date ...
all_aml (I:\GenePattern\TutorialDat...
Project
11/2/0...
all_aml_test.cls
cls
all_aml_test.preprocessed0.odf
SOM Clu...
all_aml_test.res
res
all_aml_train.cls
all_aml_train.comp.marker.filt.res
all_aml_train.res
HeatMapImage
Creates a heat map from a dataset
Show Parameter Descriptions
input filename:
all_aml_train.comp.marker.filt.res
Browse...
The input file - .res, .gct, .odf
output file:
<input.filename_basename>
The output file name - .jpeg, .png, .tiff, .bmp, .eps
output file format:
jpeg
The output file format
column size:
8
The size in pixels of an element along the x axis
row size:
8
The size in pixels of an element along the y axis
grid:
yes
Whether to draw a grid between elements
grid color:
black
The grid color
ow annotations:
no
Whether to draw row annotations
row names:
yes
Run
Reset
Help
rate Code
I:\GenePattern\TutorialData\gp_tutorial_files\all_aml
File
Edit
View
Favorites
Tools
Help
Back
Search
Folders
Address
I:\GenePattern\TutorialData\gp_tutorial_files\all_aml
Go
File and Folder Tasks
Rename this file
Move this file
Copy this file
Publish this file to the Web
E-mail this file
Delete this file
View previous versions
Other Places
gp_tutorial_files
Name
all_aml_test.cls
all_aml_train.cls
all_aml_train.comp.marker.filt.res
all_aml_test.preprocessed0.odf
all_aml_test.res
all_aml_train.res

Dragging a file from a window to the GenePattern Graphical Client

- When the job completes, open the `HeatMapImage` node.
- Right-click (Mac users Option-click) to save the file to your local `all_aml` project directory.
- Notice that the file has been added to your `all_aml` window.
- Double click on this file. You will see it open in the JPEG image browser your system currently has configured.


all_aml_train.comp.marker.filt.jpg - Windows Picture and Fax Vie...

Prediction with GenePattern

Prediction algorithms

Prediction (supervised learning) algorithms "learn" how to distinguish between members of different classes (for example, tumor versus normal) by "training" themselves on a set of data where the classifications are already known. Using these examples, prediction algorithms create a model, which is then used to predict which class an unknown sample belongs to. (By unknown, we mean unknown to the prediction algorithm. The researcher may know the correct class, and in fact, must know the correct class in order to determine the algorithm's accuracy.)

Testing prediction accuracy: Train/test and Cross-validation

There are two basic approaches to testing the accuracy of a prediction algorithm: train/test and cross validation. Train/test uses one dataset to train a prediction algorithm and another to test it. Cross-validation is a refinement of the train/test approach, separating a single dataset into a number of pieces and running one train/test once for each piece, in which the algorithm is trained on all samples not in that piece, and the samples in the piece are used as a test set. The error rates over all iterations are averaged, providing a more accurate picture of the prediction accuracy of the algorithm.

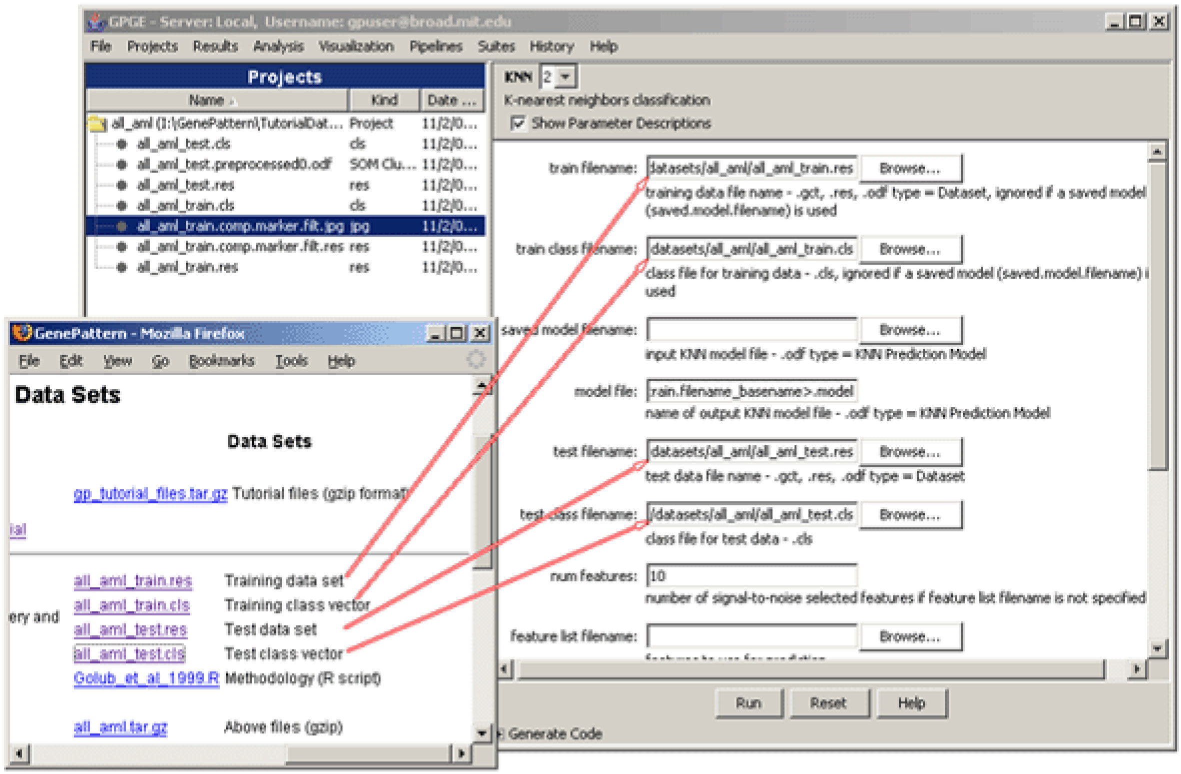
The GenePattern modules that offer cross-validation (KNN, PNN and Weighted Voting), use **leave-one-out** cross-validation, in which, for every sample, a model is created on all data except that sample, and the model is then used to predict the class of that sample.

Exercise 2h: Prediction with GenePattern: test/train/Adding files from URLs

This exercise will demonstrate how to run prediction (supervised learning) analyses with GenePattern. You will do a K-nearest neighbors (KNN) analysis on the AML/ALL dataset using a train set and a test set. (Because preprocessing is a frequent part of KNN, it is included in the algorithm.) You will then view the prediction results. This exercise will also demonstrate how to add files from a Web browser into the GenePattern Graphical Environment.

Entering Data into GenePattern from a URL

- Select the **`Analysis->Prediction->KNN`** menu item.
- Open a web browser and load the GenePattern datasets page, http://www.broad.mit.edu/cancer/software/genepattern/datasets.
- Click and drag the **`all_aml_train.res`** link to the **`train filename`** box.
- Click and drag the **`all_aml_train.cls`** link to the **`train class filename`** box.
- Click and drag the **`all_aml_test.res`** link to the **`test filename`** box.
- Click and drag the **`all_aml_test.cls`** link to the **`test class filename`** box.
- Click **`Run`**.


GPGE - Server: Local, Username: gpuser@broad.mit.edu
File Projects Results Analysis Visualization Pipelines Suites History Help
Projects
GenePattern - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Browse...
Run
Reset
Help
Generate Code

Name	Kind	Date ...
all_aml (I:\GenePattern\TutorialDat...	Project	11/2/0...
all_aml_test.cls	cls	11/2/0...
all_aml_test.preprocessed0.odf	SOM Clu...	11/2/0...
all_aml_test.res	res	11/2/0...
all_aml_train.cls	cls	11/2/0...
all_aml_train.comp.marker.filt.jpg	jpg	11/2/0...
all_aml_train.comp.marker.filt.res	res	11/2/0...
all_aml_train.res	res	11/2/0...

KNN 2

K-nearest neighbors classification

[✓] Show Parameter Descriptions

train filename: datasets/all_aml/all_aml_train.res
training data file name - .gct, .res, .odf type = Dataset, ignored if a saved model (saved.model.filename) is used

train class filename: datasets/all_aml/all_aml_train.cls
class file for training data - .cls, ignored if a saved model (saved.model.filename) i used

saved model filename:
input KNN model file - .odf type = KNN Prediction Model

model file: rain.filename_basename>.model
name of output KNN model file - .odf type = KNN Prediction Model

test filename: datasets/all_aml/all_aml_test.res
test data file name - .gct, .res, .odf type = Dataset

test class filename: /datasets/all_aml/all_aml_test.cls
class file for test data - .cls

num features: 10
number of signal-to-noise selected features if feature list filename is not specified

feature list filename:

Data Sets

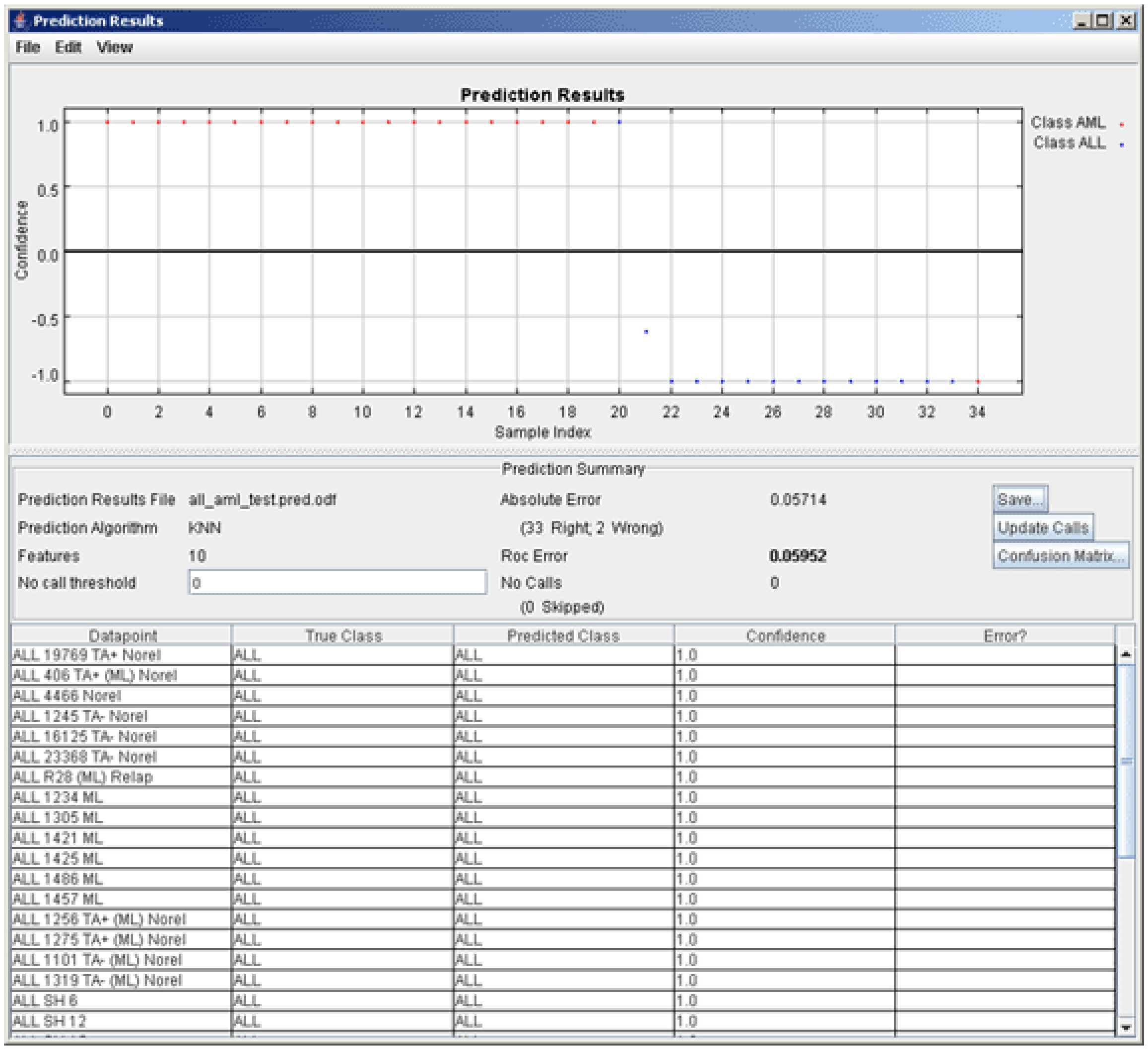
Data Sets

gp_tutorial_files.tar.gz	Tutorial files (gzip format)
all_aml_train.res	Training data set
all_aml_train.cls	Training class vector
all_aml_test.res	Test data set
all_aml_test.cls	Test class vector
Golub_et_al_1999.R	Methodology (R script)
all_aml.tar.gz	Above files (gzip)

Entering files into the GenePattern Graphical Environment from a browser

- When the job finishes, open the **`KNN`** node.
- To view the results, select the **`Visualization->PredictionResultsViewer`** menu item.
- Right-click (Mac users Option-click) **`all_aml_test.pred.odf`** and **`Send To`** to the "prediction results filename" input box.
- Click **`Run`**.

You will see the visualizer shown below, which allows you to view the results of the prediction algorithm:



Prediction Summary

Prediction Results File	all_aml_test.pred.odf	Absolute Error	0.05714
		(33 Right; 2 Wrong)	
Prediction Algorithm	KNN		
Features	10	Roc Error	**0.05952**
No call threshold	0	No Calls	0
		(0 Skipped)	

Datapoint	True Class	Predicted Class	Confidence	Error?
ALL 19769 TA+ Norel	ALL	ALL	1.0	
ALL 406 TA+ (ML) Norel	ALL	ALL	1.0	
ALL 4466 Norel	ALL	ALL	1.0	
ALL 1245 TA- Norel	ALL	ALL	1.0	
ALL 16125 TA- Norel	ALL	ALL	1.0	
ALL 23368 TA- Norel	ALL	ALL	1.0	
ALL R28 (ML) Relap	ALL	ALL	1.0	
ALL 1234 ML	ALL	ALL	1.0	
ALL 1305 ML	ALL	ALL	1.0	
ALL 1421 ML	ALL	ALL	1.0	
ALL 1425 ML	ALL	ALL	1.0	
ALL 1486 ML	ALL	ALL	1.0	
ALL 1457 ML	ALL	ALL	1.0	
ALL 1256 TA+ (ML) Norel	ALL	ALL	1.0	
ALL 1275 TA+ (ML) Norel	ALL	ALL	1.0	
ALL 1101 TA- (ML) Norel	ALL	ALL	1.0	
ALL 1319 TA- (ML) Norel	ALL	ALL	1.0	
ALL SH 6	ALL	ALL	1.0	
ALL SH 12	ALL	ALL	1.0	

This visualizer shows the following information for each sample:

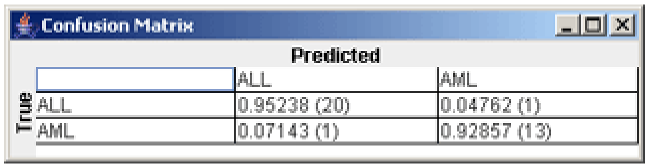
- **Datapoint** - The name of the sample whose class is being predicted
- **Predicted Class** - The class that the algorithm predicted for this sample
- **True Class** - The true class of the sample
- **Confidence** - The confidence level with which the algorithm has predicted this class
- **Error** - Marked if the prediction is incorrect

Confidence Levels

This viewer also allows you to specify a minimum confidence level, under which a sample will not be counted in the error rate of the predictor. To change this threshold level, enter a value between 0 and 1 in the **`No call threshold`** input box and click **`Update Calls`**. You will see an **`NC`** in the **`Error`** column where a sample is marked as no-call, and the **`ROC Error`** value will change accordingly.

Confusion Matrix

If you want to see where the algorithm tends to misclassify samples, click the **`Confusion Matrix`** button. You will see the following screen:



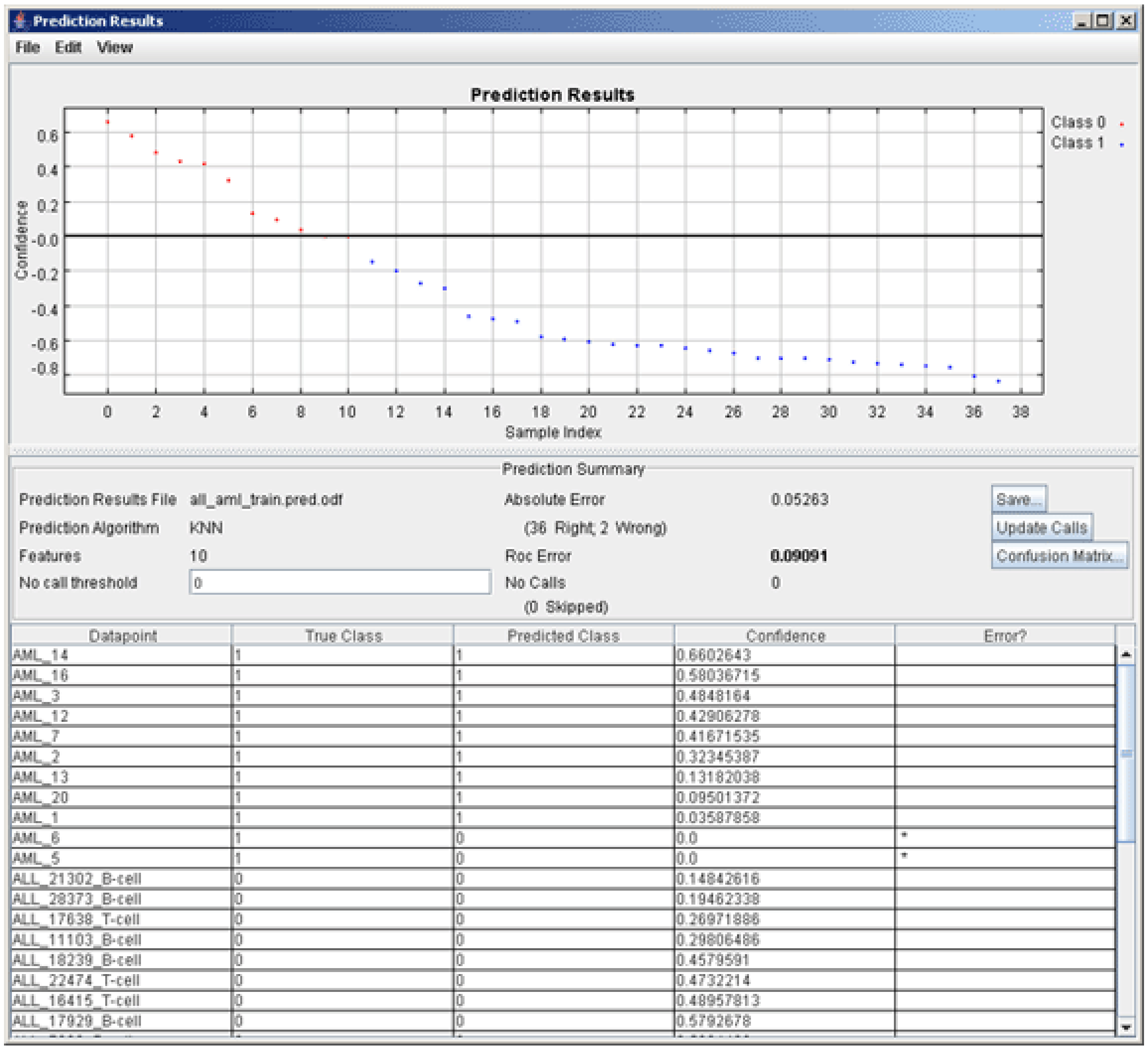
		Predicted	
		ALL	AML
True	ALL	0.95238 (20)	0.04762 (1)
	AML	0.07143 (1)	0.92857 (13)

members that were given this prediction (each row totals to 100%). The second, in parentheses, is the actual number of samples with the given true and predicted classes. For example, in the cell corresponding to a predicted class of ALL and a true class of AML, there is 1 sample, representing 7% of class ALL samples which were misclassified as class AML.

Exercise 2i: Supervised Learning with GenePattern: cross-validation

This exercise will compare the train/test approach used in the previous exercise with cross-validation. You will run the same KNN algorithm, but will use cross-validation rather than train/test to determine its accuracy.

- Select the **`Analysis->Prediction->KNNXValidation`** menu item.
- In the all_aml project directory, right-click (Mac users Option-click) **`all_aml_train.res`** and **`Send To`** the "data filename" input box.
- In the all_aml project directory, right-click (Mac users Option-click) **`all_aml_train.cls`** and **`Send To`** the "class filename" input box.
- Click **`Run`**.
- When the job is done, open the **`KNNXValidation`** node.
- Select the **`Visualization->PredictionResultsViewer`** menu item.
- Right-click (Mac users Option-click) and send the **`all_aml_train.pred.odf`** file to the "prediction results filename" input box.
- Click **`Run`**.



Prediction Summary

Prediction Results File	all_aml_train.pred.odf	Absolute Error	0.05263
Prediction Algorithm	KNN	(36 Right; 2 Wrong)	
Features	10	Roc Error	**0.09091**
No call threshold	0	No Calls	0
		(0 Skipped)	

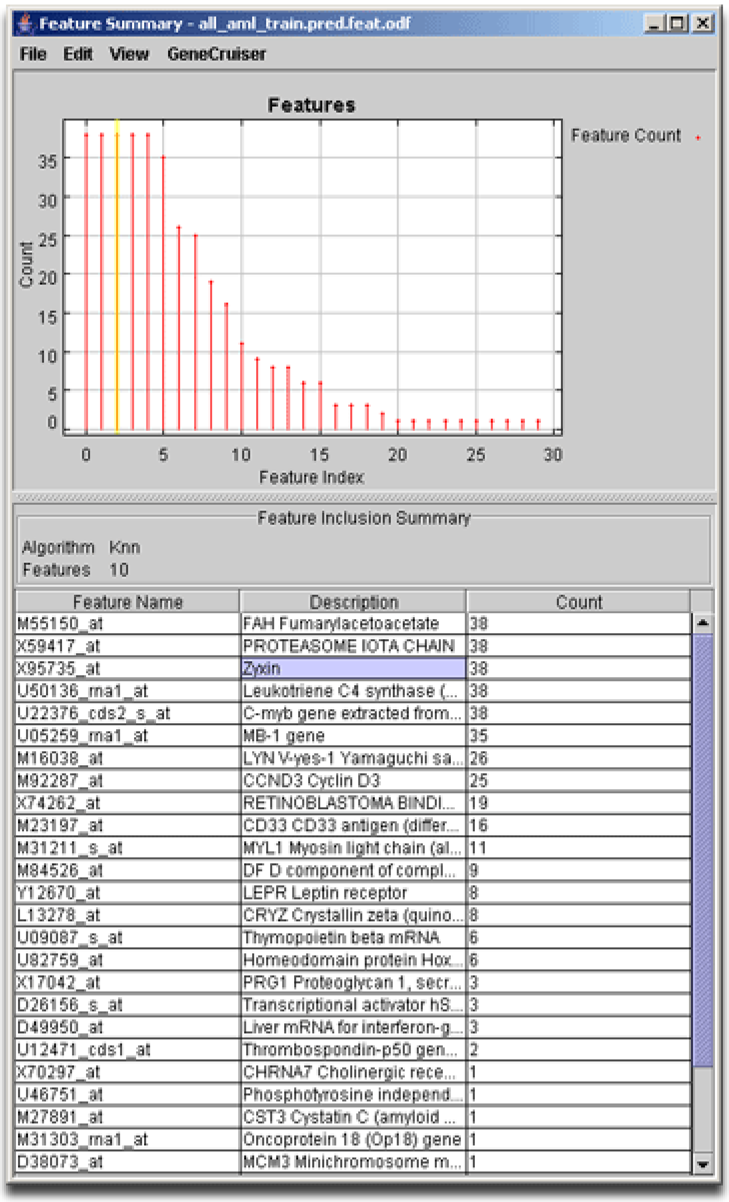
Datapoint	True Class	Predicted Class	Confidence	Error?
AML_14	1	1	0.6602643	
AML_16	1	1	0.58036715	
AML_3	1	1	0.4848164	
AML_12	1	1	0.42906278	
AML_7	1	1	0.41671535	
AML_2	1	1	0.32345387	
AML_13	1	1	0.13182038	
AML_20	1	1	0.09501372	
AML_1	1	1	0.03587858	
AML_6	1	0	0.0	*
AML_5	1	0	0.0	*
ALL_21302_B-cell	0	0	0.14842616	
ALL_28373_B-cell	0	0	0.19462338	
ALL_17638_T-cell	0	0	0.26971886	
ALL_11103_B-cell	0	0	0.29806486	
ALL_18239_B-cell	0	0	0.4579591	
ALL_22474_T-cell	0	0	0.4732214	
ALL_16415_T-cell	0	0	0.48957813	
ALL_17929_B-cell	0	0	0.5792678	

Viewing Model Features: Feature Summary Viewer

When you create a prediction model, you specify how many features of the original dataset are to be used in the model. In the exercise you have been doing, the number of features you enter in the **`Num Features`** input box of the **`KNNXValidation`** module corresponds to the number of genes that are used to predict whether a sample is ALL or AML.

When you run a cross-validation loop, a new set of features is chosen each time the algorithm is trained. To view a summary of the features that were used in cross-validation and how often they appear in a model, you can use the FeatureSummaryViewer:

- Select the **`Visualization->FeatureSummaryViewer`** menu item.
- Open the **`KNNXValidation`** node.
- Click and drag the **`all_aml_train.pred.feat.odf`** file to the "feature filename" input box.
- Click **`Run`**.


Feature Summary - all_aml_train.pred.feat.odf
File Edit View GeneCruiser
Features
Feature Count
Count
Feature Index
0 5 10 15 20 25 30 35
0 5 10 15 20 25 30
Feature Inclusion Summary

Algorithm Knn
Features 10

Feature Name	Description	Count
M55150_at	FAH Fumarylacetoacetate	38
X59417_at	PROTEASOME IOTA CHAIN	38
X95735_at	Zyxin	38
U50136_rna1_at	Leukotriene C4 synthase (...	38
U22376_cds2_s_at	C-myb gene extracted from...	38
U05259_rna1_at	MB-1 gene	35
M16038_at	LYN V-yes-1 Yamaguchi sa...	26
M92287_at	CCND3 Cyclin D3	25
X74262_at	RETINOBLASTOMA BINDI...	19
M23197_at	CD33 CD33 antigen (differ...	16
M31211_s_at	MYL1 Myosin light chain (al...	11
M84526_at	DF D component of compl...	9
Y12670_at	LEPR Leptin receptor	8
L13278_at	CRYZ Crystallin zeta (quino...	8
U09087_s_at	Thymopoietin beta mRNA	6
U82759_at	Homeodomain protein Hox...	6
X17042_at	PRG1 Proteoglycan 1, secr...	3
D26156_s_at	Transcriptional activator hS...	3
D49950_at	Liver mRNA for interferon-g...	3
U12471_cds1_at	Thrombospondin-p50 gen...	2
X70297_at	CHRNA7 Cholinergic rece...	1
U46751_at	Phosphotyrosine independ...	1
M27891_at	CST3 Cystatin C (amyloid ...	1
M31303_rna1_at	Oncoprotein 18 (Op18) gene	1
D38073_at	MCM3 Minichromosome m...	1

Feature Summary Viewer

The Feature Summary Viewer displays information about the model that was used in a prediction algorithm:

- **Model Type** - This indicates the algorithm that generated this model
- **Num Features** - The number of features selected in each iteration of the cross-validation loop

The table shows the features that were chosen in all runs of the cross-validation loop, along with the number of times they were chosen:

- **Feature Name** - The name of the feature.

- **Description** - A description of the feature
- **Count** - The number of iterations of the cross-validation loop in which this feature was included in the model. In the example of **`all_aml_train.res`**, there are 38 samples, so in leave-one-out cross validation, this number can be at most 38. Higher values indicate greater stability for this feature as a predictor.

Additional Analysis Features

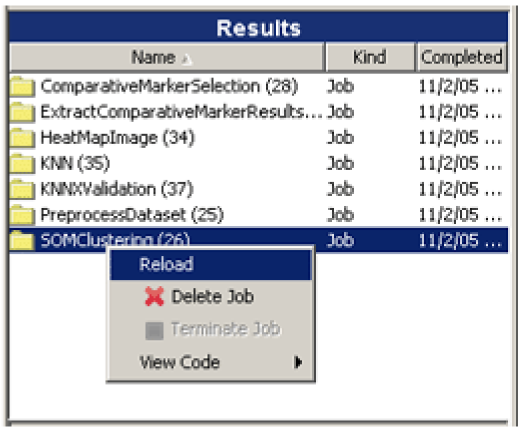
The GenePattern Graphical Client contains several features that allow you to work more effectively. These include replaying and removing previous analyses, and running batch analyses.

Exercise 2j: Re-running previous analyses

You may find that you need to re-run a previous analysis with small changes, or to recall the parameters that you used in a certain analysis. GenePattern Graphical Client allows you to do this by reloading the parameters of any analysis job you have run.

In this exercise, you will reload the parameters you used in the SOM analysis, change the number of clusters, and re-run the analysis.

- Right-click (Mac users Option-click) the **`SOM Clustering`** node.
- You will see a pop-up menu prompting you to **`Reload, Delete Job`** or **`View Code`**, as shown below.
- Select **`Reload`**. (Alternatively, click the node and select **`Results->Reload`**.)
- You will see the **`SOMClustering`** analysis window, with the parameters you entered in the SOMClustering exercise.
- In the **`cluster range`** input box, enter 5.
- Click **`Run`**. The SOMClustering algorithm will run again, with the same input files and parameters you used in the previous analysis but with 5 clusters this time.



Results

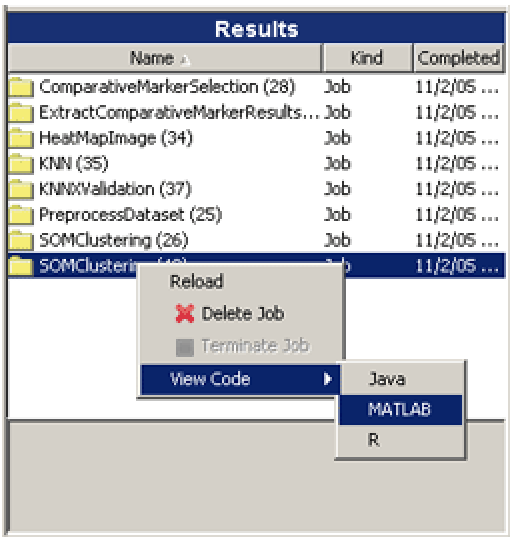
Name	Kind	Completed
ComparativeMarkerSelection (28)	Job	11/2/05 ...
ExtractComparativeMarkerResults...	Job	11/2/05 ...
HeatMapImage (34)	Job	11/2/05 ...
KNN (35)	Job	11/2/05 ...
KNNXValidation (37)	Job	11/2/05 ...
PreprocessDataset (25)	Job	11/2/05 ...
SOMClustering (26)	Job	11/2/05 ...

Reload a job that was run earlier

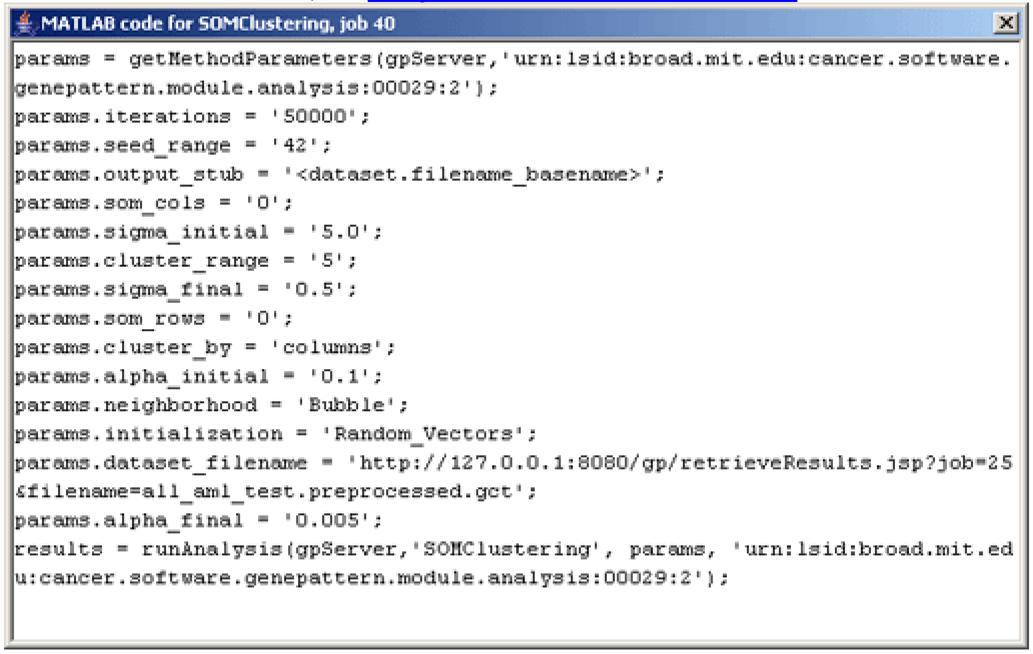
Alternatively, you can re-run previous tasks from the **`History`** menu: select the **`History`** menu, which lists previously run tasks, and then select the task to re-run.

In some cases you may be interested in copying the code used to produce this analysis result.

- Right-click (Mac users Option-click) the **`SOM Clustering`** node.
- You will see a pop-up menu prompting you to **`Reload, Delete Job`** or **`View Code`**.
- Select **`View Code`**. (Alternatively, click the node and select **`Results->View Code`**.)
- You will see an option for Java, MATLAB, and R.


Results
Name
Kind
Completed
ComparativeMarkerSelection (28) Job 11/2/05 ...
ExtractComparativeMarkerResults... Job 11/2/05 ...
HeatMapImage (34) Job 11/2/05 ...
KNN (35) Job 11/2/05 ...
KNNXValidation (37) Job 11/2/05 ...
PreprocessDataset (25) Job 11/2/05 ...
SOMClustering (26) Job 11/2/05 ...
SOMClusteri... Job 11/2/05 ...
Reload
Delete Job
Terminate Job
View Code
Java
MATLAB
R

If you choose MATLAB, for example, you will see the following window with the code for the SOM Clustering job. For more information about how MATLAB code works with GenePattern, see Using MATLAB Modules in GenePattern.



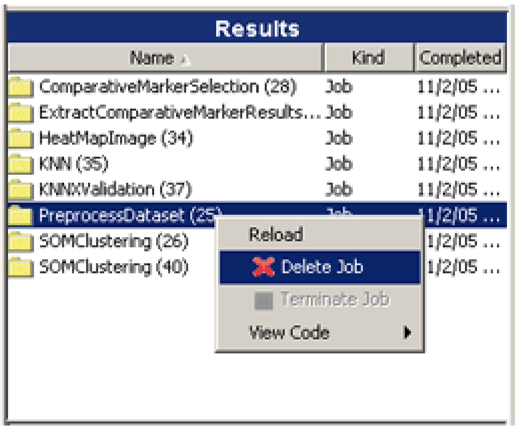
```
params = getMethodParameters(gpServer,'urn:lsid:broad.mit.edu:cancer.software.
genepattern.module.analysis:00029:2');
params.iterations = '50000';
params.seed_range = '42';
params.output_stub = '<dataset.filename_basename>';
params.som_cols = '0';
params.sigma_initial = '5.0';
params.cluster_range = '5';
params.sigma_final = '0.5';
params.som_rows = '0';
params.cluster_by = 'columns';
params.alpha_initial = '0.1';
params.neighborhood = 'Bubble';
params.initialization = 'Random_Vectors';
params.dataset_filename = 'http://127.0.0.1:8080/gp/retrieveResults.jsp?job=25
&filename=all_aml_test.preprocessed.gct';
params.alpha_final = '0.005';
results = runAnalysis(gpServer,'SOMClustering', params, 'urn:lsid:broad.mit.ed
u:cancer.software.genepattern.module.analysis:00029:2');
```

MATLAB code for the SOM Clustering job run earlier

Exercise 2k: Removing result files

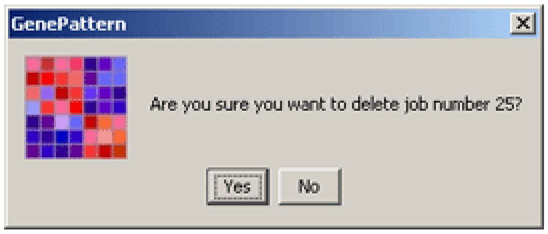
As you run more and more analyses, the Object Browser will contain a growing number of result nodes. If you no longer need results from an analysis, you can remove them from your `Results` pane.

In this exercise, you will remove the results of the `PreprocessDataset` analysis you ran in Exercise 2a.



- Close the **`PreprocessDataset`** node.
- Right-click (Mac users Option-click) the node.
- You will see a pop-up menu prompting you to **`Reload, Delete Job`** or **`View Code`**.
- Select **`Delete Job`**.

(Alternatively, click the node and select **`Results->Delete Job`**.)



- You will see a dialog box asking if you are sure you want to remove this job.
- Click **`Yes`**.

The entire **`PreprocessDataset`** node will be removed.

Alternatively, you can delete analysis results from the **`History`** menu: select **`History->View All`**. In the History window, select the task to delete and then select **`Purge`**.

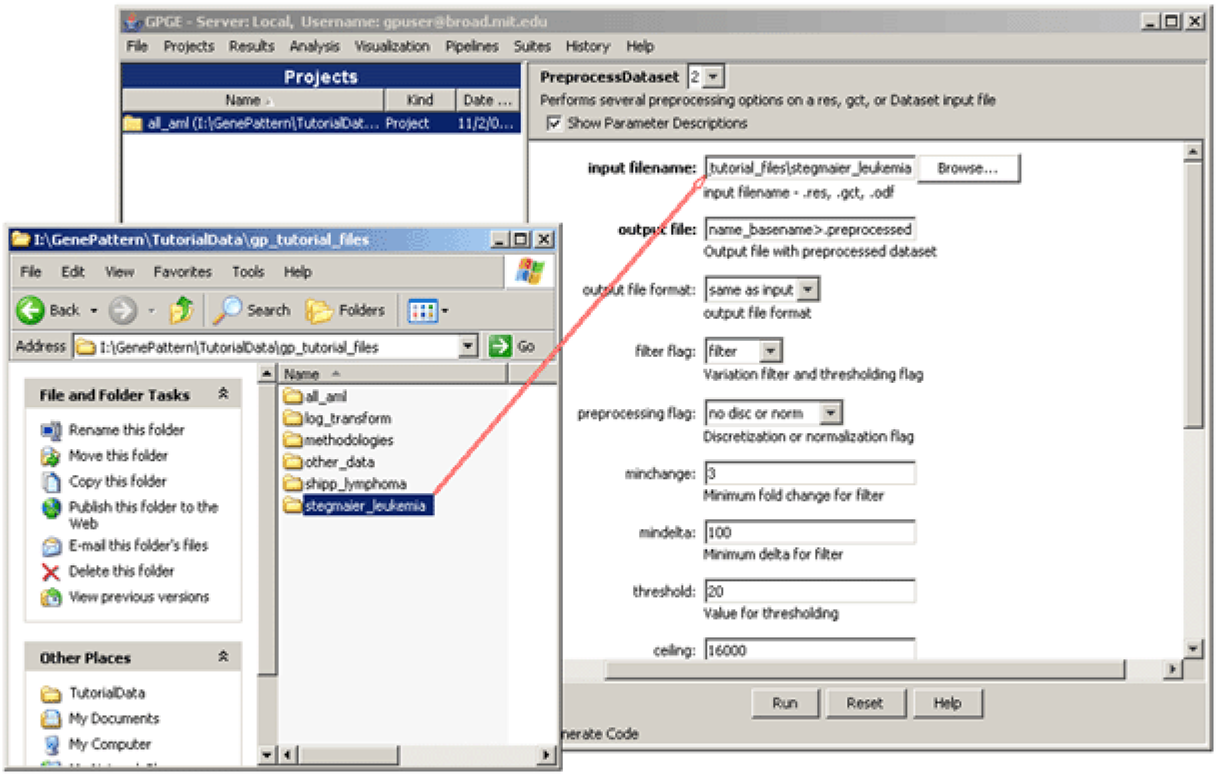
Exercise 2l: Running batch analyses

You can use GenePattern to run an analysis automatically over all of the files in a directory. In this exercise, you will use the batch processing functionality of GenePattern to preprocess the contents of the stegmaier_leukemia dataset in the tutorial_datasets directory.

- Select the **`Analysis->Preprocessing & Utilities->PreprocessDataset`**.
- Find and open the folder on your hard drive that contains the **`tutorial_datasets`** folder.
- Click and drag the **`stegmaier_leukemia`** folder to the **`Input filename`** box.

➤
- When running batch analyses, if a task has multiple file parameters, you can specify a directory name for only one of those parameters.

- Click **`Run`**.


GPGE - Server: Local, Username: gpuser@broad.mit.edu
File Projects Results Analysis Visualization Pipelines Suites History Help
Projects
I:\GenePattern\TutorialData\gp_tutorial_files
File and Folder Tasks
Other Places
Run Reset Help

Name	Kind	Date ...
all_aml (I:\GenePattern\TutorialDat...	Project	11/2/0...

PreprocessDataset 2
Performs several preprocessing options on a res, gct, or Dataset input file
☑ Show Parameter Descriptions

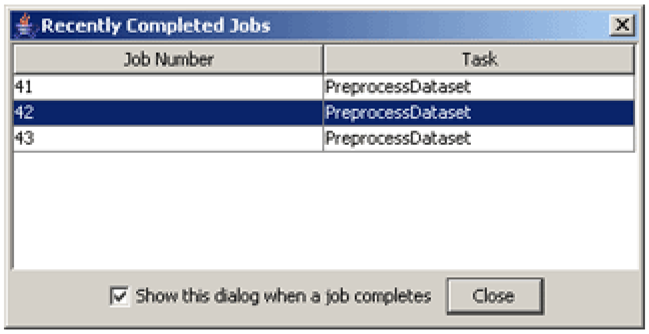
input filename:	tutorial_files\stegmaier_leukemia (input filename - .res, .gct, .odf)
output file:	name_basename>.preprocessed (Output file with preprocessed dataset)
output file format:	same as input (output file format)
filter flag:	filter (Variation filter and thresholding flag)
preprocessing flag:	no disc or norm (Discretization or normalization flag)
minchange:	3 (Minimum fold change for filter)
mindelta:	100 (Minimum delta for filter)
threshold:	20 (Value for thresholding)
ceiling:	16000

Address I:\GenePattern\TutorialData\gp_tutorial_files

all_aml
log_transform
methodologies
other_data
shipp_lymphoma
stegmaier_leukemia

Analyzing the entire contents of a directory

The analysis will be run on each file in the directory. As each job completes, you will see it appear in the `Recently Completed Jobs` window, shown below. If a directory contains files that are in a format that is not appropriate for that algorithm, you will see a message informing you that the file is in the wrong format, and the rest of the files will be processed.


Recently Completed Jobs

Job Number	Task
41	PreprocessDataset
42	PreprocessDataset
43	PreprocessDataset

☑ Show this dialog when a job completes   Close

Results of batch preprocessing analysis

Pipelines

Pipelines allow you to chain modules together into analysis workflows, including the visualization of intermediate and final results. You can use pipelines to run analyses automatically rather than having to repeat each step manually, to run the same analysis over different data sets, or to encapsulate the algorithms and parameters you used for a method so it can be remembered later.

Pipelines are essential to the GenePattern design objective of reproducible research. GenePattern uses Life Science Identifiers (LSIDs) to uniquely identify specific tasks and pipelines. Whenever a new pipeline is created, it is assigned a new LSID. When that pipeline definition is subsequently modified, the version portion of the LSID is incremented. Each task within the pipeline also has a specific LSID. Therefore, all of the input parameter names, values, and semantics are fixed and not subject to change or "improvement". Updating a task or a pipeline automatically updates the version number of its LSID; the original task or pipeline is not changed or deleted. Thus, a researcher may precisely reproduce any previous analysis with a mouse click.

- Typically, you update the latest version of a task or pipeline, which increments its version number. For example, editing version 1 of a pipeline creates version 2 of that pipeline. You may at times need to edit an older version of a task or pipeline. This creates a “point version” of that task or pipeline. For example, if you have versions 1 and 2 of a pipeline, editing version

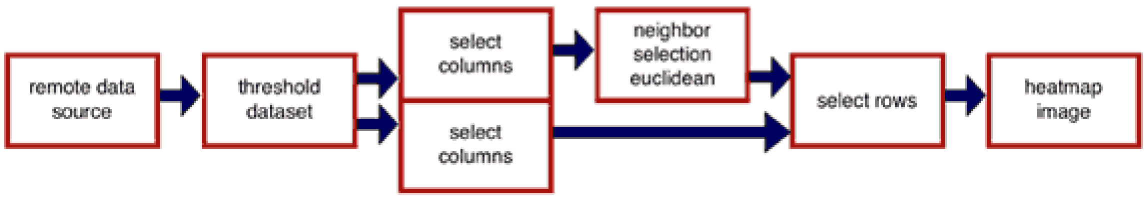
1 of the pipeline creates version 1.1 of that pipeline.

You can create, edit, and run pipelines from either the GenePattern Graphical Client or the GenePattern Web Client. To delete pipelines, you use the GenePattern Web Client. In this section, you learn how to work with pipelines using the GenePattern Graphical Client.

Types of Pipelines

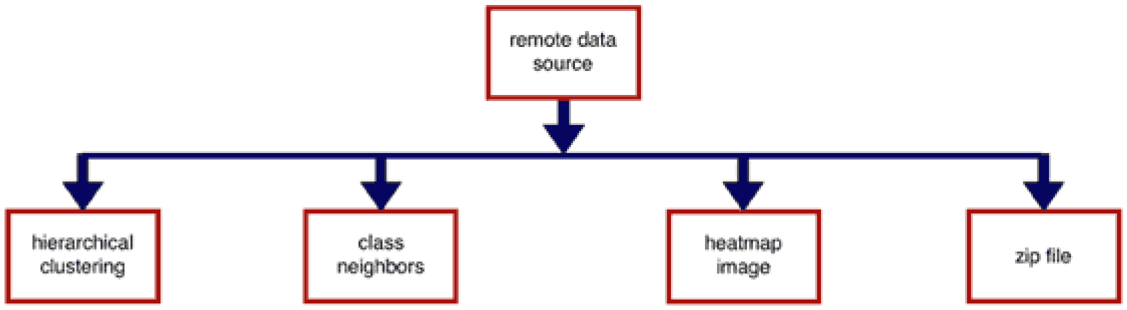
A sequential pipeline starts with a data source and performs various operations, each operation building on the results of the previous one, until a final result is reached.

Sequential Pipeline



A parallel pipeline performs many independent tasks on the same data file.

Parallel Pipeline



Most real-world pipelines have both parallel and serial elements.

Viewing a Pipeline

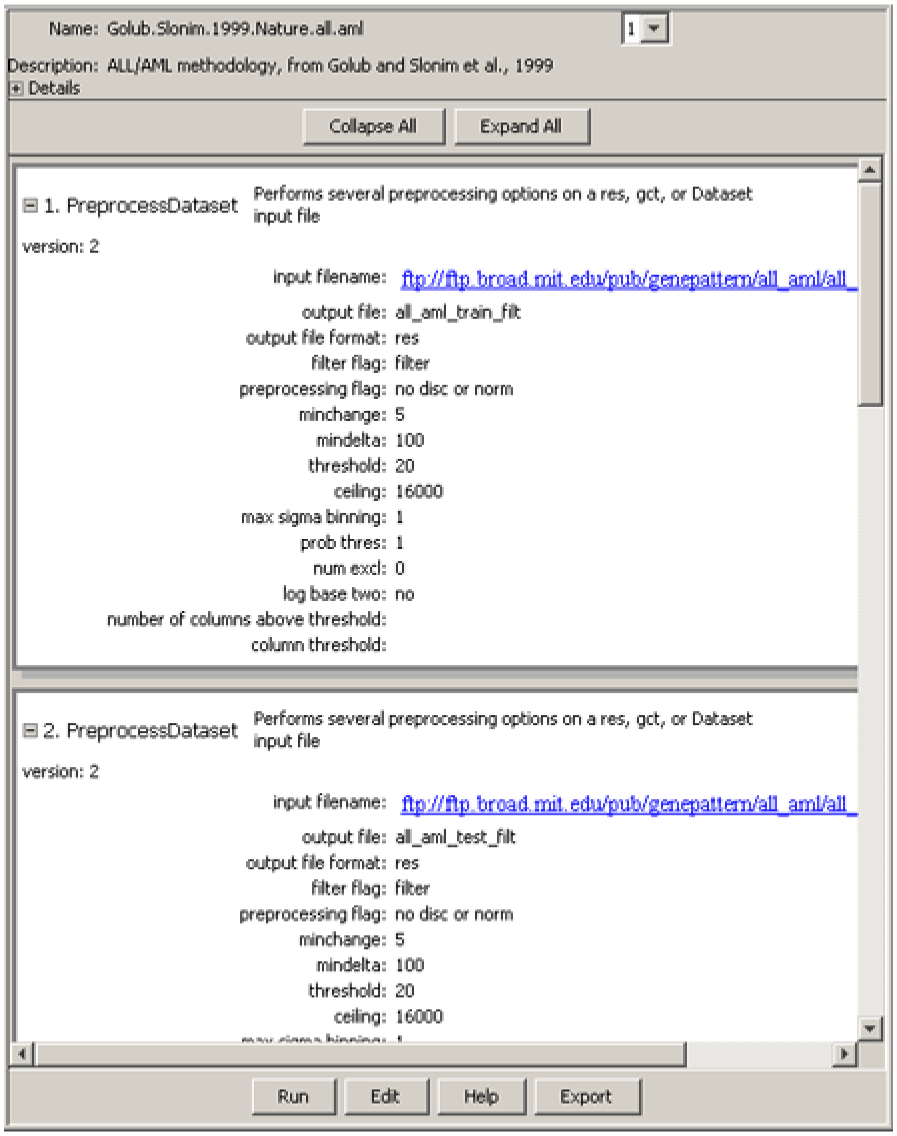
The **`Pipelines`** menu lists the pipelines available to you. In this exercise, you will examine the Golub.Slonim.1999.Nature.all.aml pipeline (Golub and Slonim *et al.*), which uses clustering and prediction algorithms to find genes that distinguish between two subtypes of leukemia, ALL and AML.

Exercise 3a: Viewing a pipeline

To view the Golub.Slonim.1999.Nature.all.aml pipeline:

- Select **`Pipelines-> Golub.Slonim.1999.Nature.all.aml`**.
- Click **`View`** at the bottom of the page. The pipeline definition appears in the window.

  - If any modules used in the pipeline are not installed on the server, the window lists the missing modules. Click the **`Install Missing Modules From Catalog`** button to install the missing modules. After the modules have been installed, click **`Refresh View`**. The window should now look the same as the one shown below.


Name: Golub.Slonim.1999.Nature.all.aml

Description: ALL/AML methodology, from Golub and Slonim et al., 1999

Details

1. PreprocessDataset Performs several preprocessing options on a res, gct, or Dataset input file

version: 2

```
                  input filename: ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_
                     output file: all_aml_train_filt
              output file format: res
                     filter flag: filter
              preprocessing flag: no disc or norm
                       minchange: 5
                        mindelta: 100
                       threshold: 20
                         ceiling: 16000
               max sigma binning: 1
                      prob thres: 1
                        num excl: 0
                    log base two: no
number of columns above threshold:
                column threshold:
```

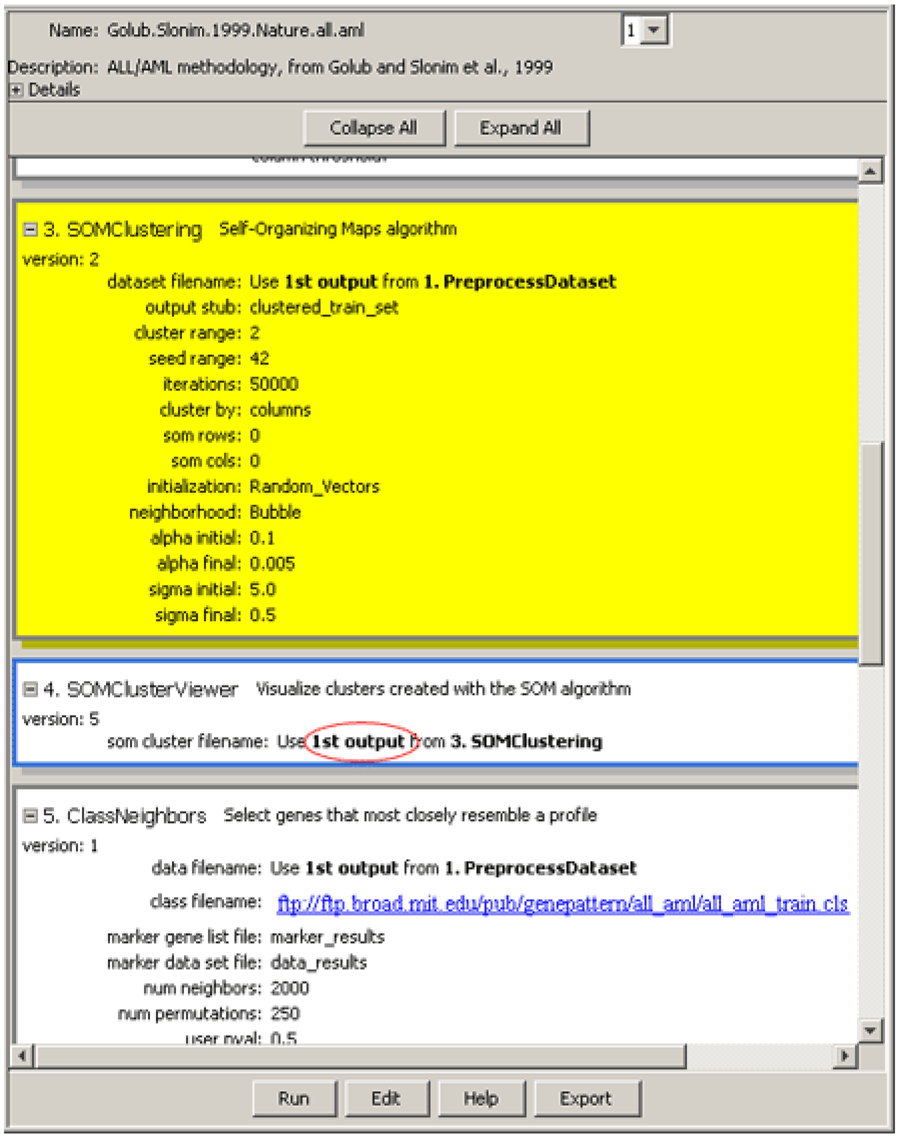
2. PreprocessDataset Performs several preprocessing options on a res, gct, or Dataset input file

version: 2

```
                  input filename: ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_
                     output file: all_aml_test_filt
              output file format: res
                     filter flag: filter
              preprocessing flag: no disc or norm
                       minchange: 5
                        mindelta: 100
                       threshold: 20
                         ceiling: 16000
```

Notice the following elements of the pipeline definition window:

- The first line shows the name of the pipeline and a drop-down menu of versions. By default, you are viewing the most recent version of the pipeline.
- To view author information and documentation for the pipeline, expand the **`Details`** field by clicking the plus (+) sign.
- Below the **`Details`** field is a scrolling window that lists each task in the pipeline, as well as its parameters. You can hide/show parameter details for each task individually (plus/minus icon to the left of each task) or for all tasks at once (**`Collapse All`**/**`Expand All`** buttons at the top of the window).
- The key to connecting tasks in a pipeline is to take the output from one task and use it as the input for another task. For example, in this pipeline, the input for Task 4 (SOMClusterViewer) is the output from Task 3 (SOMClustering). To help you visualize these connections, when you mouse over an input file that comes from another task, the GenePattern Graphical Client highlights the source task. To see this, place your mouse cursor over the words **`1st output`** in Task 4. When you do, Task 3 (SOMClustering) is highlighted in yellow, as shown below this list.
- The buttons at the bottom of the window allow you to run, edit, or export the pipeline to a zip file. **`Help`** displays the documentation for pipeline.


Name: Golub.Slonim.1999.Nature.all.aml

Description: ALL/AML methodology, from Golub and Slonim et al., 1999

3. SOMClustering Self-Organizing Maps algorithm
version: 2

```
 dataset filename: Use 1st output from 1. PreprocessDataset
      output stub: clustered_train_set
    cluster range: 2
       seed range: 42
       iterations: 50000
       cluster by: columns
         som rows: 0
         som cols: 0
   initialization: Random_Vectors
     neighborhood: Bubble
    alpha initial: 0.1
      alpha final: 0.005
    sigma initial: 5.0
      sigma final: 0.5
```

4. SOMClusterViewer Visualize clusters created with the SOM algorithm
version: 5

```
som cluster filename: Use 1st output from 3. SOMClustering
```

5. ClassNeighbors Select genes that most closely resemble a profile
version: 1

```
        data filename: Use 1st output from 1. PreprocessDataset
       class filename: ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_train.cls
marker gene list file: marker_results
 marker data set file: data_results
        num neighbors: 2000
     num permutations: 250
           user pval: 0.5
```

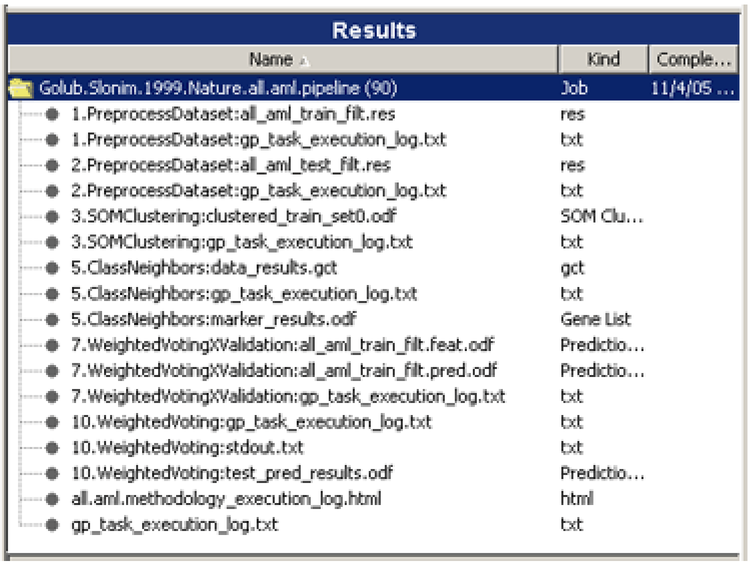
Running a Pipeline

Now that you have examined the pipeline definition, you can run the pipeline. In this exercise you run the Golub.Slonim.1999.Nature.all.aml pipeline and examine the results.

Exercise 3b: Running a pipeline

- Select **`Pipelines->Golub.Slonim.1999.Nature.all.aml`**. (If you are already viewing or editing the pipeline, you do not need to select it from the menu.)
- Click **`Run`** at the bottom of the page.

The pipeline is listed in the Results window with a status of processing. As each task in the pipeline completes, its output files are added to the results window. After a few minutes, a number of visualization tools open as those tasks in the pipeline complete. The pipeline job is completed when all tasks in the pipeline are completed. At that point, your results window should like the one below:



Results

Name	Kind	Comple...
Golub.Slonim.1999.Nature.all.aml.pipeline (90)	Job	11/4/05 ...
1.PreprocessDataset:all_aml_train_filt.res	res	
1.PreprocessDataset:gp_task_execution_log.txt	txt	
2.PreprocessDataset:all_aml_test_filt.res	res	
2.PreprocessDataset:gp_task_execution_log.txt	txt	
3.SOMClustering:clustered_train_set0.odf	SOM Clu...	
3.SOMClustering:gp_task_execution_log.txt	txt	
5.ClassNeighbors:data_results.gct	gct	
5.ClassNeighbors:gp_task_execution_log.txt	txt	
5.ClassNeighbors:marker_results.odf	Gene List	
7.WeightedVotingXValidation:all_aml_train_filt.feat.odf	Predictio...	
7.WeightedVotingXValidation:all_aml_train_filt.pred.odf	Predictio...	
7.WeightedVotingXValidation:gp_task_execution_log.txt	txt	
10.WeightedVoting:gp_task_execution_log.txt	txt	
10.WeightedVoting:stdout.txt	txt	
10.WeightedVoting:test_pred_results.odf	Predictio...	
all.aml.methodology_execution_log.html	html	
gp_task_execution_log.txt	txt	

The pipeline results files are exactly the same as the analysis results files. Right-click (Mac users Option-click) a file to open, save, or delete the file. (Alternatively, click the file and select the desired action from the `Results` menu.)

Creating a Pipeline

You can create a pipeline in one of two ways:

- From scratch. Use Pipelines->New to create a new pipeline. You then add tasks to that pipeline.
- From a results file. Right-click (Mac users Option-click) a results file and select `Create Pipeline` to create a new pipeline that contains the tasks used to create the results file. (Alternatively, click the file and select `Results->Create Pipeline`.) GenePattern adds the module that created the results file to the pipeline and then checks its input file. If the input file for the module was the output file of a previous module, GenePattern adds the previous module to the pipeline and then checks its input file. GenePattern continues to walk back through the chain of modules, adding to the pipeline, until it reaches the initial input file.

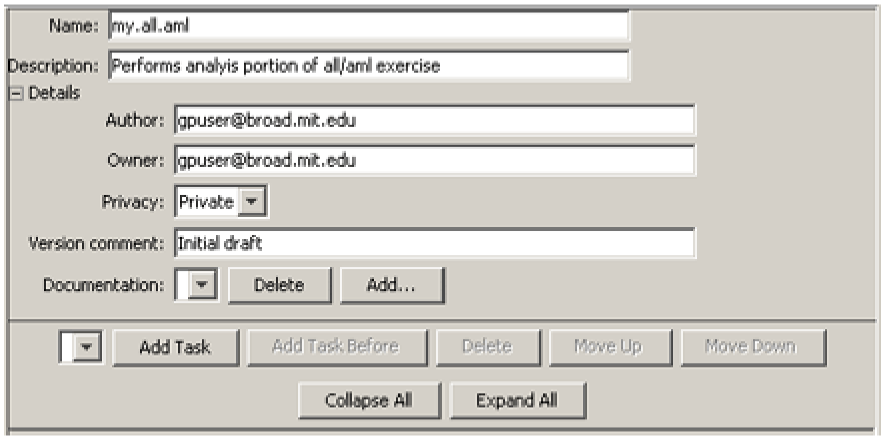
In this exercise, you will create a pipeline from scratch. The pipeline will do the analysis portions of the first all/aml exercise: filtering, SOM clustering, and finding markers for the resulting data.

Exercise 3c: Creating a pipeline

- Select `Pipelines->New`.
- In the `Name` field, enter: my.all.aml.

  - Avoid special characters in pipeline names, such as: !,@,#,$,%,^,&,*,_ Also, machines running Windows cannot accept files with names that consist of the following, regardless of the file extension: `con, prn, aux, nul, com1, com2, com3, com4, lpt1, lpt2, lpt3`. For cross-platform compatibility, it is best to avoid usingthese names.
  - GenePattern does not prevent you from using the same name for multiple pipelines; however, using unique names is strongly recommended.

- In the `Description` field, enter: Performs analysis portion of all/aml exercise.
- Display author information by clicking the plus (+) sign next to `Details`. Completing this section thoughtfully helps to maintain pipelines and ensure that analyses are understandable and reproducible.
- In the `Privacy` field, select Private. A private pipeline can only be seen and run by a user logged in using the email address specified in either the author or owner fields of this form; a public pipeline can be seen and run by all users.
- In the `Version comment` field, enter: initial draft. This comment will be visible on pages used to keep versions up to date, so keep your comments complete and concise.
- You use the `Documentation` field to upload documentation to accompany your pipeline. Documentation is strongly encouraged for "Public" pipelines.


Name: my.all.aml

Description: Performs analyis portion of all/aml exercise

Details

Author: gpuser@broad.mit.edu

Owner: gpuser@broad.mit.edu

Privacy: Private

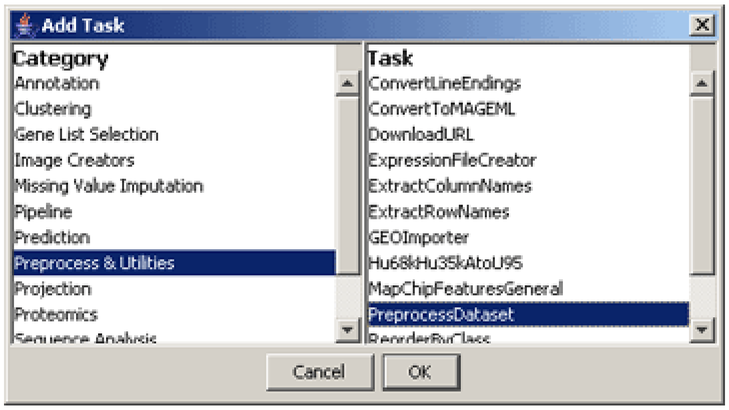
Version comment: Initial draft

Documentation:

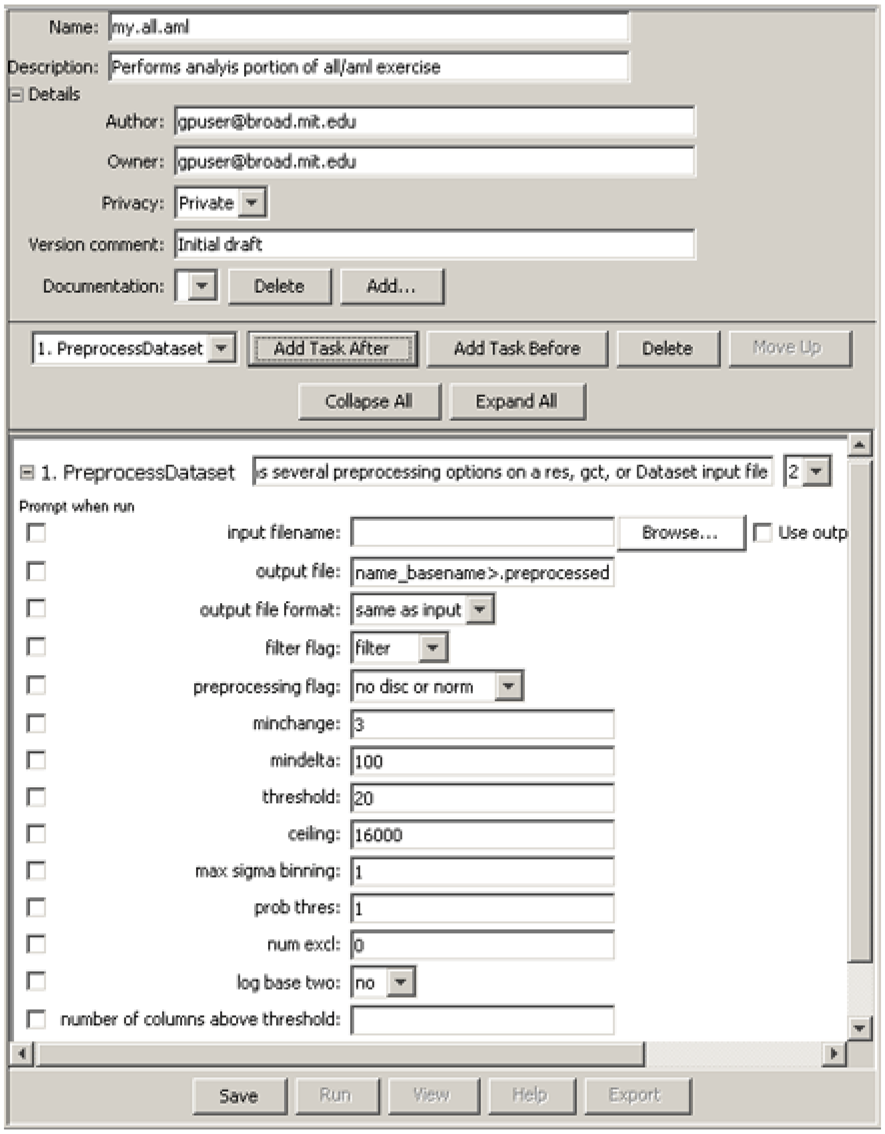
Exercise 3d: Adding tasks to a pipeline

Now, you are ready to add tasks to your pipeline.

- Click the **`Add Task`** button. The Add Task window appears.
- From the Category list, select Preprocess & Utilities. The Task list on the right is populated with all preprocessing modules.
- From the Task list, select PreprocessDataset and click **`OK`**.



The PreprocessDataset task and its parameters are displayed in the pipeline definition window.


Name: my.all.aml

Description: Performs analyis portion of all/aml exercise

Details

Author: gpuser@broad.mit.edu

Owner: gpuser@broad.mit.edu

Privacy: Private

Version comment: Initial draft

Documentation:

1. PreprocessDataset | s several preprocessing options on a res, gct, or Dataset input file | 2

Prompt when run

☐	input filename:	
☐	output file:	name_basename>.preprocessed
☐	output file format:	same as input
☐	filter flag:	filter
☐	preprocessing flag:	no disc or norm
☐	minchange:	3
☐	mindelta:	100
☐	threshold:	20
☐	ceiling:	16000
☐	max sigma binning:	1
☐	prob thres:	1
☐	num excl:	0
☐	log base two:	no
☐	number of columns above threshold:	

The first line of the parameters window shows the task name, description, and a drop-down menu of versions. The drop-down menu allows you to choose which version of the installed PreprocessDataset module to use. Ensure you have selected the latest version.

For each parameter, you can enter a value or select the **`Prompt when run`** check box to have the pipeline prompt for a value each time it is run. Enter the following parameter values:

- For the **`input filename`**, click Browse and navigate to your datasets. If you need to download them again you may do so here. Go into the **`aml_all folder`** and choose **`all_aml_train.res`**.
- For the **`output file format`**, select "gct".
- For the **`filter flag`**, select filter.
- For **`minchange`**, enter 5.

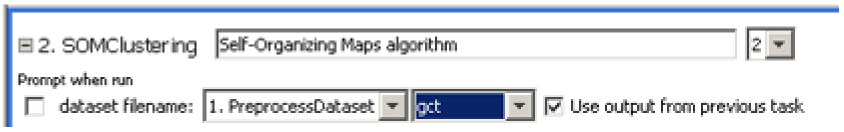
Let the rest of the parameters remain at the default. GenePattern will provide a filename for your results files. If you'd like to use a name other than the one provided you can enter it in the **`output filename`** box.

- Click the **`Add Task After`** button to add a task after Task 1, PreprocessDataset; the task selected in the drop-down menu to the left of the button.
- When the Add Task window appears, select Clustering from the Category list, select SOM Clustering from the Task list, and click **`OK`**.

The SOMClustering task and its parameters are displayed in the pipeline definition window. (You can scroll the window to see the previous task.) In the first line of the task definition, check the version drop-down menu to ensure that the latest version is selected.

As mentioned earlier, pipelines allow you to take the output from one module and use it as the input of another module. You want the input for the SOMClustering task to be the gct output from the PreprocessDataset task. To do this:

- To the left of the **`dataset filename`** field, click the check box **`Use output from previous task`**. The Browse button is replaced by the **`Choose Task`** drop-down menu, which lists all previous tasks in the pipeline.
- Select **`PreprocessDataset`** from the drop-down menu. Because the PreprocessDataset has multiple output files, a Dataset drop-down menu appears.
- Select gct from that drop-down menu.



2. SOMClustering | Self-Organizing Maps algorithm | 2

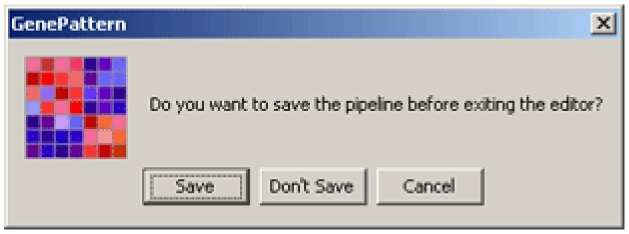
Prompt when run

dataset filename: 1. PreprocessDataset | gct | ☑ Use output from previous task

- For **`cluster range`**, enter 2.
- Save the pipeline by clicking the **`Save`** button at the bottom of the page.
- A confirmation message appears. Click **`OK`**. (The new pipeline is now in the Pipelines menu.)

To run your newly created pipeline:

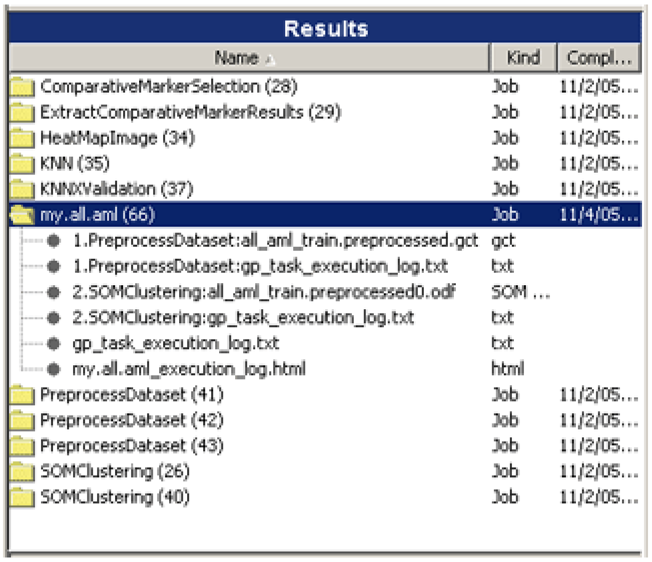
- Click the **`Run`** button at the bottom of the screen.
- A confirmation window appears asking if you want to save the pipeline before exiting the editor. You have just saved the pipeline, so click **`Don't Save`**. (Clicking **`Save`** will create a new version of the pipeline.)



GenePattern

Do you want to save the pipeline before exiting the editor?

The pipeline runs and the results appear in the Results window, as shown below. Right-click (Mac users Option-click) a file to open, save, or delete the file.



Results

Name	Kind	Compl...
ComparativeMarkerSelection (28)	Job	11/2/05...
ExtractComparativeMarkerResults (29)	Job	11/2/05...
HeatMapImage (34)	Job	11/2/05...
KNN (35)	Job	11/2/05...
KNNXValidation (37)	Job	11/2/05...
my.all.aml (66)	Job	11/4/05...
1.PreprocessDataset:all_aml_train.preprocessed.gct	gct	
1.PreprocessDataset:gp_task_execution_log.txt	txt	
2.SOMClustering:all_aml_train.preprocessed0.odf	SOM ...	
2.SOMClustering:gp_task_execution_log.txt	txt	
gp_task_execution_log.txt	txt	
my.all.aml_execution_log.html	html	
PreprocessDataset (41)	Job	11/2/05...
PreprocessDataset (42)	Job	11/2/05...
PreprocessDataset (43)	Job	11/2/05...
SOMClustering (26)	Job	11/2/05...
SOMClustering (40)	Job	11/2/05...

Editing a Pipeline

You can edit any pipeline listed in the Pipelines menu, regardless of whether you originally created the pipeline. You can create a new version of the existing pipeline or creating a new pipeline based on the existing pipeline. To create a new version of the pipeline, edit the pipeline, make the changes, and save it. To create a new pipeline based on an existing one, edit the pipeline, change its name, make any other changes, and save it.

In the following exercises, you edit the pipeline that you just created. You create a new version of the pipeline adding the ComparativeMarkerSelection task and a prompt for the cluster range parameter of the SOMClustering task.

Exercise 3e: Editing a pipeline

To edit the my.all.aml pipeline:

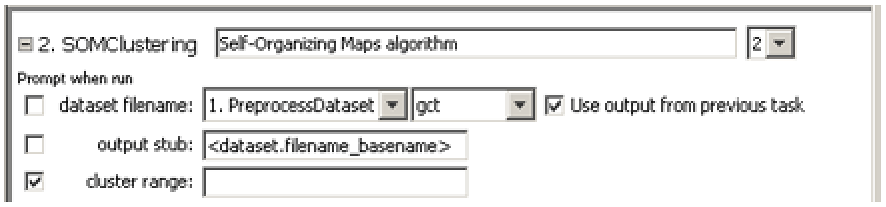
- Click **`Pipelines->my.all.aml`**. Notice that the pipelines that you have created are color-coded in bright pink.
- Click the **`Edit`** button. The pipeline definition is displayed in the window.

First, update the version comment:

- Expand the **`Details`** field (click the plus (+) sign).
- Modify the **`Version comment`** to reflect the changes you intend to make.

Next, have the pipeline prompt the user for the cluster range parameter value for Task 2, SOMClustering, each time it is run:

- Scroll to the **`cluster range`** field of Task 2, SOMClustering.
- Click the **`Prompt when run`** check box to the left of that field.



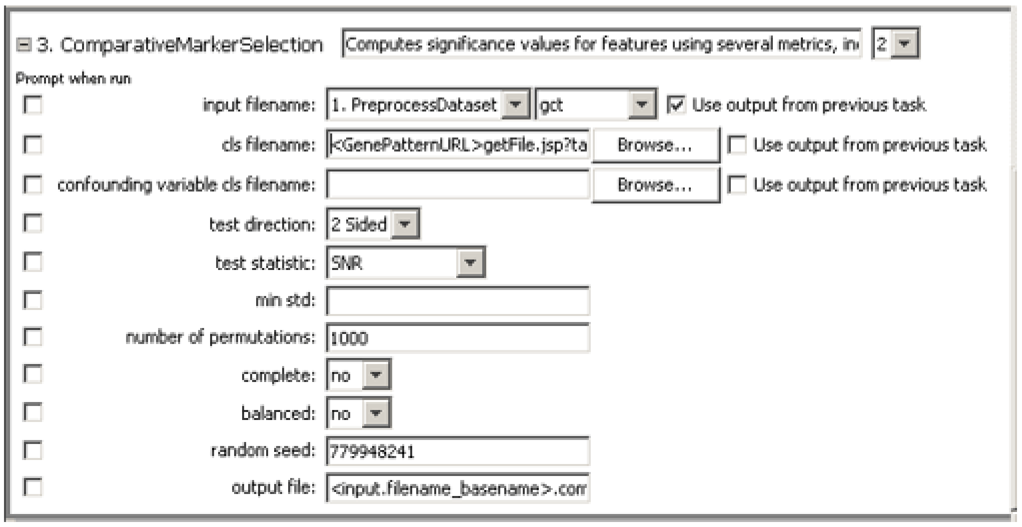
⊟ 2. SOMClustering — Self-Organizing Maps algorithm — 2

Prompt when run

☐	dataset filename:	1. PreprocessDataset	gct	☑ Use output from previous task
☐	output stub:	<dataset.filename_basename>		
☑	cluster range:			

Now, add the ComparativeMarkerSelection task to the end of the pipeline (after the SOMClustering task):

- Locate the **`Add Task`** buttons at the top of the task definition window.
- In the drop-down menu to the left of the **`Add Task After`** button, select the SOMClustering task.
- Click the **`Add Task After`** button. The Add Task window appears.
- When the Add Task window appears, select GeneListSelection from the Category list, ComparativeMarkerSelection from the Task list, and click **`OK`**. The ComparativeMarkerSelection task and its parameters are displayed in the pipeline definition window.
- The input to this analysis should be the gct file that results from the PreprocessDataset task. To select this file: select the **`Use output from previous task`** checkbox to the right of the **`input filename`** field; from the Choose Task drop-down menu, select PreprocessDataset; from the Dataset drop-down menu, select gct.
- The **`cls filename`** parameter for this analysis will be the **`all_aml_train.cls file`**. Click the Browse button, navigate to your tutorial datasets, go into the **`aml_all`** folder, and choose **`all_aml_train.cls`**.
- Leave the other parameters set at their default values.



⊟ 3. ComparativeMarkerSelection — Computes significance values for features using several metrics, in — 2

Prompt when run

☐	input filename:	1. PreprocessDataset	gct	☑ Use output from previous task
☐	cls filename:	<GenePatternURL>getFile.jsp?ta	Browse...	☐ Use output from previous task
☐	confounding variable cls filename:		Browse...	☐ Use output from previous task
☐	test direction:	2 Sided		
☐	test statistic:	SNR		
☐	min std:			
☐	number of permutations:	1000		
☐	complete:	no		
☐	balanced:	no		
☐	random seed:	779948241		
☐	output file:	<input.filename_basename>.com		

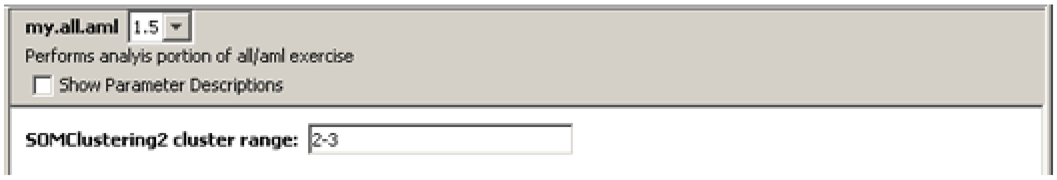
Other editing features, which were not used in this exercise, include:

- **`Move Up`** button, which rearranges the tasks in the pipeline. For example, to run the ComparativeMarkerSelection task before the SOMClustering task, you would click the ComparativeMarkerSelection task and then click **`Move Up`**.
- **`Delete`** button, which deletes a task from the pipeline. For example, to delete the ComparativeMarkerSelection task, you would click the ComparativeMarkerSelection task and then click **`Delete`**.

Finally, to save your changes and run the pipeline:

- Click **`Save`**. When the confirmation message appears, click **`OK`**.
- Click **`Run`**. When the confirmation message appears, click **`Don't Save`** (you just saved the pipeline).

This time, when you run the pipeline, it prompts you for the cluster range parameter, as shown below:



my.all.aml 1.5
Performs analyis portion of all/aml exercise
Show Parameter Descriptions

SOMClustering2 cluster range: 2-3

- Enter 2 for the SOMClustering cluster range value.
- Click **`Run`**. The pipeline results appear in the Results window.

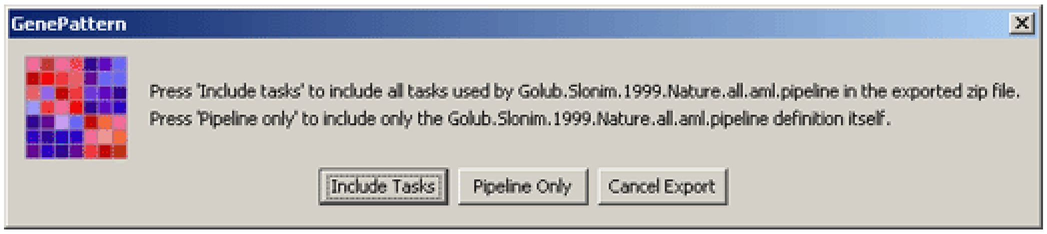
Exporting a Pipeline

Exporting a pipeline creates a zip file, which can easily be transported. If you want to share the pipeline (and its tasks) with other GenePattern users, you can send them the pipeline zip file. The other users can then import the pipeline into their GenePattern environments.

Exercise 3f: Exporting a pipeline

In this exercise, you export the Golub.Slonim.1999.Nature.all.aml pipeline.

- Select **`Pipelines->Golub.Slonim.1999.Nature.all.aml`**.
- Click the **`Export`** button at the bottom of the window. The export confirmation window appears.



GenePattern

Press 'Include tasks' to include all tasks used by Golub.Slonim.1999.Nature.all.aml.pipeline in the exported zip file.
Press 'Pipeline only' to include only the Golub.Slonim.1999.Nature.all.aml.pipeline definition itself.

You can export either the pipeline or the pipeline and its tasks:

- Click **`Pipeline Only`** to create a zip that contains the pdf documentation for the pipeline and a programmatic description of the pipeline.
- Click **`Include Tasks`** to create a zip file that contains multiple zip files: the pipeline zip file (as described above) and a zip file for each task. The zip file for each task contains the pdf documentation for the task and the program files that implement task.

To import a zipped pipeline file into your GenePattern environment, select **`File->Import Module`**. This menu item allows you to import pipelines, analysis modules, and visualization tools.

Deleting a Pipeline

To delete a pipeline, use the GenePattern Web Client (see Deleting Modules).

Suites

Use suites to filter the menus that list the analysis modules, visualization modules, and pipelines showing only those that you are interested in. To use suites:

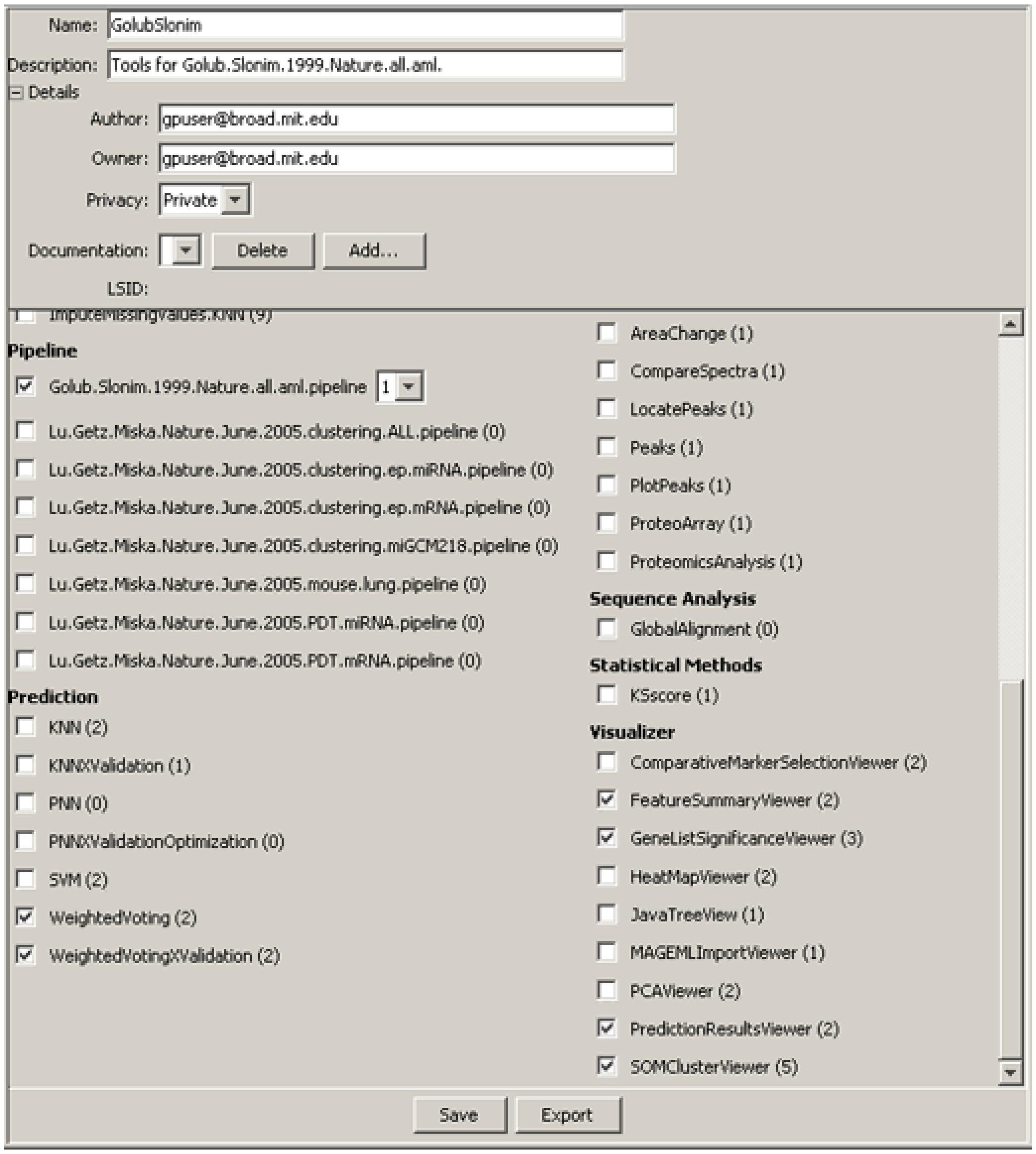
- You create a suite that lists the modules and pipelines that you are interested in.
- You apply the suite as a filter to the menus. The menus will show only the modules and pipelines that are listed in the suite.

In the following exercises, you filter the menus to show the Golub.Slonim.1999.Nature.all.aml pipeline and the analysis and visualization modules used in that pipeline.

Exercise 4a: Creating a suite

To create a suite:

- Select **`Suites->New`**. The suite definition appears in the window.
- In the **`Name`** field, enter a name for the suite: GolubSlonim.
- In the **`Description`** field, enter: Tools for Golub.Slonim.1999.Nature.all.aml.
- Expand the **`Details`** field and, in the **`Privacy`** field, select Private.
- Select the check boxes for the modules used in the Golub.Slonim.1999.Nature.all.aml pipeline: under Clustering, SOMClustering; under Gene List Selection, ClassNeighbors; under Pipeline, Golub.Slonim.1999.Nature.all.aml (select version 1 of the pipeline); under Prediction, WeightedVoting and WeightedVotingXValidation; under Preprocess & Utilities, PreprocessDataset; under Visualizer, FeatureSummaryViewer, GeneListSignificanceViewer, PredicationResultsViewer, and SOMClusterViewer.
- Click **`Save`**.
- When the confirmation message appears, click **`OK`**.


Name: GolubSlonim

Description: Tools for Golub.Slonim.1999.Nature.all.aml.

⊟ Details

Author: gpuser@broad.mit.edu

Owner: gpuser@broad.mit.edu

Privacy: Private

Documentation: Delete Add...

LSID:

☐ ImputeMissingValues.KNN (9)

Pipeline

- ✓ Golub.Slonim.1999.Nature.all.aml.pipeline 1
- ☐ Lu.Getz.Miska.Nature.June.2005.clustering.ALL.pipeline (0)
- ☐ Lu.Getz.Miska.Nature.June.2005.clustering.ep.miRNA.pipeline (0)
- ☐ Lu.Getz.Miska.Nature.June.2005.clustering.ep.mRNA.pipeline (0)
- ☐ Lu.Getz.Miska.Nature.June.2005.clustering.miGCM218.pipeline (0)
- ☐ Lu.Getz.Miska.Nature.June.2005.mouse.lung.pipeline (0)
- ☐ Lu.Getz.Miska.Nature.June.2005.PDT.miRNA.pipeline (0)
- ☐ Lu.Getz.Miska.Nature.June.2005.PDT.mRNA.pipeline (0)

Prediction

- ☐ KNN (2)
- ☐ KNNXValidation (1)
- ☐ PNN (0)
- ☐ PNNXValidationOptimization (0)
- ☐ SVM (2)
- ✓ WeightedVoting (2)
- ✓ WeightedVotingXValidation (2)

- ☐ AreaChange (1)
- ☐ CompareSpectra (1)
- ☐ LocatePeaks (1)
- ☐ Peaks (1)
- ☐ PlotPeaks (1)
- ☐ ProteoArray (1)
- ☐ ProteomicsAnalysis (1)

Sequence Analysis

- ☐ GlobalAlignment (0)

Statistical Methods

- ☐ KSscore (1)

Visualizer

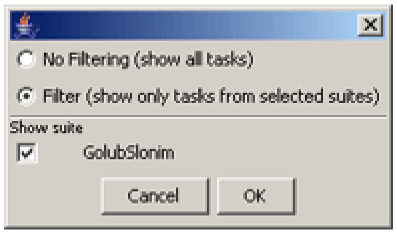
- ☐ ComparativeMarkerSelectionViewer (2)
- ✓ FeatureSummaryViewer (2)
- ✓ GeneListSignificanceViewer (3)
- ☐ HeatMapViewer (2)
- ☐ JavaTreeView (1)
- ☐ MAGEMLImportViewer (1)
- ☐ PCAViewer (2)
- ✓ PredictionResultsViewer (2)
- ✓ SOMClusterViewer (5)

Save Export

Exercise 4b: Applying a filter

Now, use the GolubSlonim suite to filter the Analysis, Visualization, and Pipeline menus:

- Select **`Suites->Filter`**.
- In the Filter window, select the **`Filter`** radio button. The window lists the available suites.
- Click the check box next to GolubSlonim, and then click **`OK`**.


○ No Filtering (show all tasks)

● Filter (show only tasks from selected suites)

Show suite

✓ GolubSlonim

Cancel OK

Notice that the Analysis, Visualization, and Pipeline menus now contain only the analysis modules, visualization modules, and pipelines that are listed in the GolubSlonim suite. To remove the filter from the menus:

- Select **`Suites->Filter`**.
- In the Filter window, select the **`No Filtering`** radio button.
- Click **`OK`**.

Notice that the Analysis, Visualization, and Pipeline menus now show all of the analysis modules, visualization modules, and pipelines.

Exercise 4c: Editing a suite

To edit a suite:

- Select `Suites-><suite-name>`. For example, to edit the GolubSlonim suite, select `Suites->GolubSlonim`. The suite definition appears in the window.
- Make the desired changes and click the Save button. GenePattern creates a new version of the suite.

Exercise 4d: Exporting a suite

Exporting a suite creates a zip file, which can easily be transported. If you want to share the suite with other GenePattern users, you can send them the suite zip file. The other users can then use the GenePattern Web Client to import the suite on to their GenePattern servers.

To export a suite:

- Select `Suites-><suite-name>`. For example, to export the GolubSlonim suite, select `Suites->GolubSlonim`. The suite definition appears in the window.
- Click the `Export` button. The file selection window appears.
- Select a destination directory and name for the zip file, then click `Save`.

To import a suite, you must use the GenePattern Web Client:

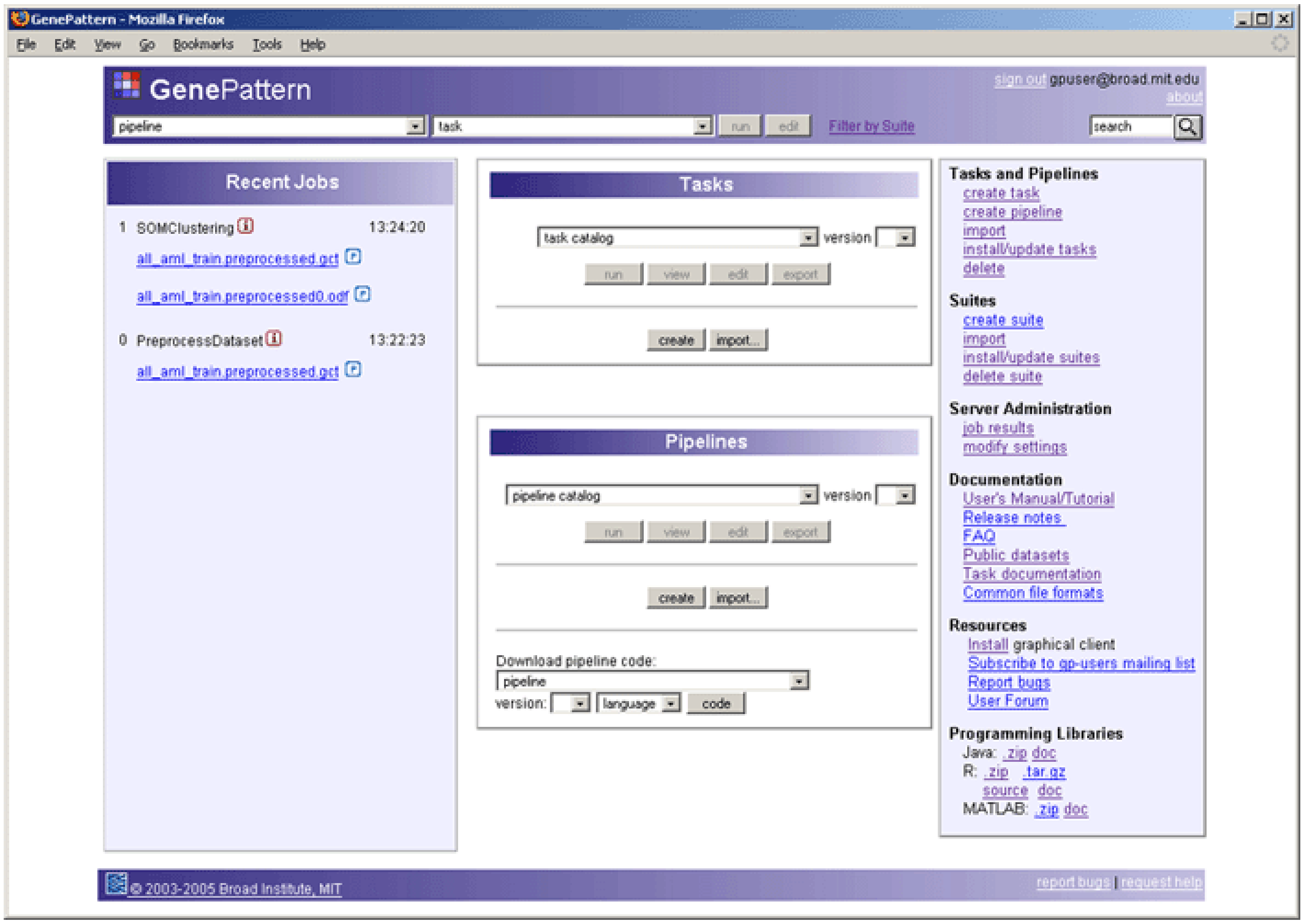
- Launch the GenePattern Web Client.
- Under Suites in the right column, select `import`.

You will learn more about the GenePattern Web Client in the next section of the tutorial.

GenePattern Web Client

The GenePattern Web Client (also called the GenePattern home page or GenePattern server home page) is primarily used for administrative tasks. To explore the Web Client, you will need to have both the GenePattern server and GenePattern Web Client running. If you have not launched them, you can follow the directions for launching the GenePattern server and Web Client in Getting Started.

After you launch the GenePattern Web Client, you will see the web page shown below:



- The GenePattern icon in the upper right corner returns you to this home page.
- The Recent Jobs section lists the jobs that the server has recently executed.

  Click the information icon next to a job to display the job's status and its parameters.

  Click a results file to view the file.

  Click the pipeline icon next to a results file to create a pipeline that contains the modules used to generate that file.

- The Tasks section lists the analysis and visualization modules. Both the Web Client and the GenePattern Graphical Environment (GPGE) allow you to run, view, edit, export, and import these modules. The Web Client also allows you to create and delete modules, as described in Task Integrator.
- The Pipelines section lists the pipeline modules. Both the Web Client and the GenePattern Graphical Environment (GPGE) allow you to run, view, edit, export, create, and import these modules. For information about working with pipelines in the Web Client, see Using Pipelines.

In the column on the right:

- The Tasks and Pipelines links provide quick access to key functions that are also available in the Tasks and Pipelines sections. In addition, you can use these links to delete modules (analysis modules, visualization modules, and pipelines) and to install modules from the Broad Repository. For more information, see Task Integrator.
- The Suites links allow you to create, import, install, and delete suites, as described in Using Suites. Use the Filter by Suite link in the title bar to have the Tasks and Pipelines menus show only those modules listed in a particular suite.
- The Server Administration links provide access to two key server areas: the job results window, as described in Managing Jobs, and the server administration window, as described in Server Administration.
- The Documentation links provide quick access to all GenePattern documentation, including the pdf documentation files for all analysis, visualization, and pipeline modules.
- The Resources links provide quick access to various support resources for GenePattern. In addition to these links, note the two links in the lower right corner of the window, which you can use to request help or report bugs.

- The Programming Libraries links provide quick access to the software and documentation that you need to use GenePattern from a programming language. For more information, click the doc link for the appropriate language.

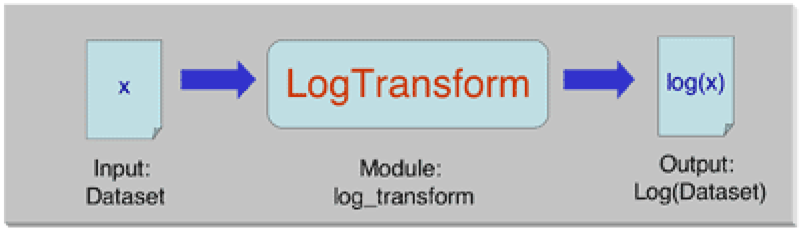
GenePattern Task Integrator

The *GenePattern Task Integrator* allows you to add tasks and visualizers without writing code. It makes it easy to add "third-party" tools, or to maintain a common repository of methodologies at your organization. To access the Task Integrator, use the Tasks section of the GenePattern Web client, as shown in the following exercises.

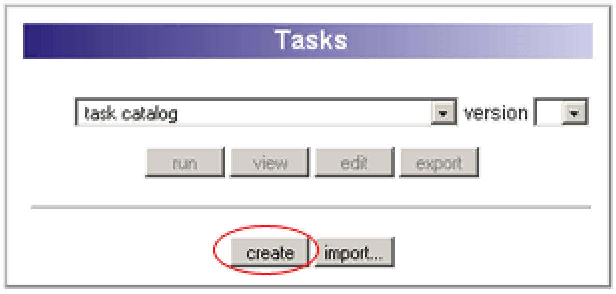
Adding Modules

GenePattern provides three ways to add modules. You can add modules you've written yourself, you can download and update modules from the Broad Institute's module repository, or you can import modules from a zip file. We will cover these options in the following exercises.

Exercise 5a: Adding a Module
In this exercise you will learn how to add an analysis module to the GenePattern server. The module you will add is a Perl script that log-transforms all the positive values in a dataset, as shown in the figure below. Any negative or zero values will be set to 0.



- From the GenePattern home page, under the Tasks section, shown below, click on **`create`**.



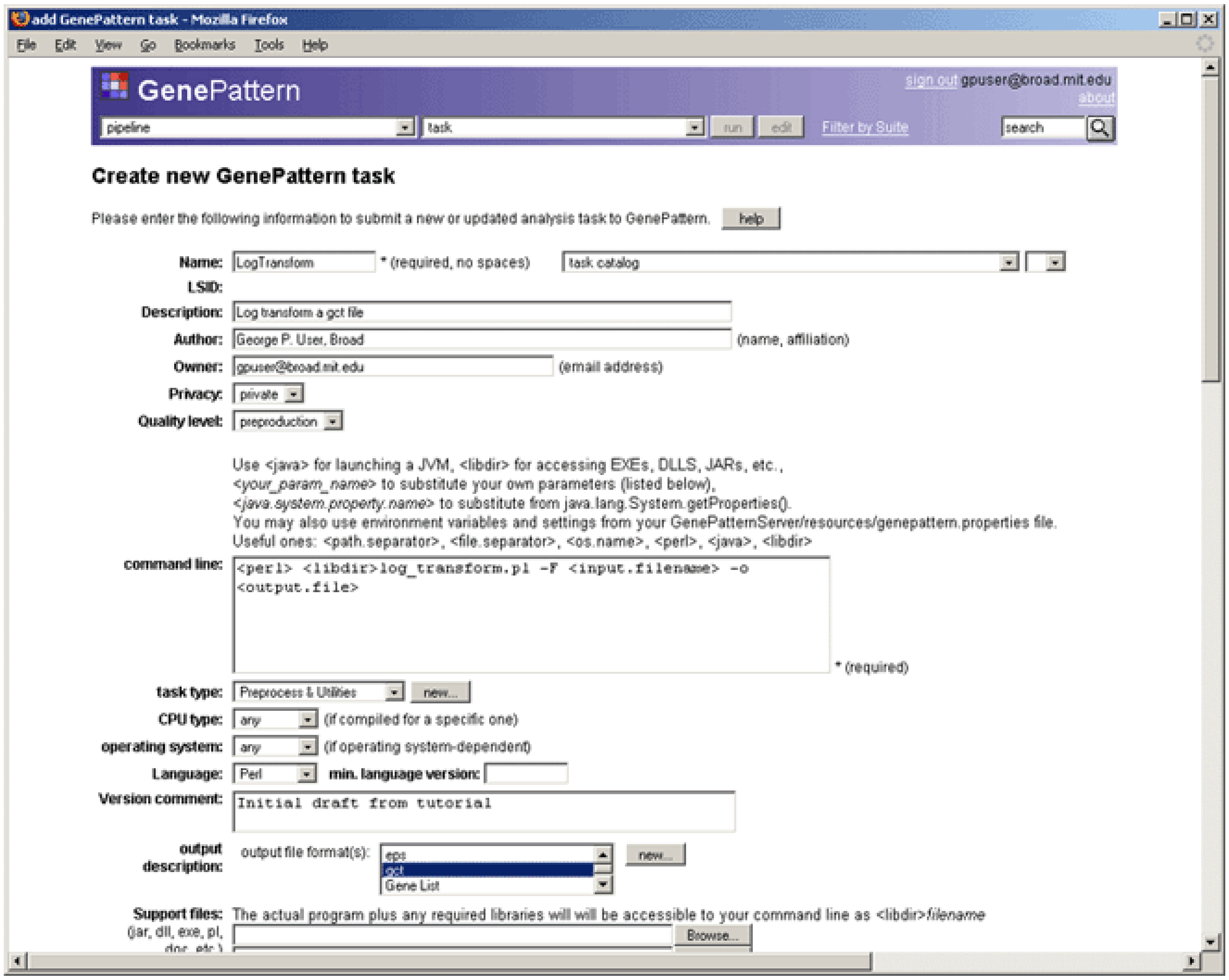
- You will see a form containing fields that GenePattern will use to describe and store this module. Click **`Help`** at the top of the form to display instructions for adding a task, including descriptions of the fields on the form. Enter the following information into the fields:
  - **Name**: **`LogTransform`**
  - **LSID**: filled in by GenePattern
  - **Description**: **`Log transform a gct file`**
  - **Author**: **`your name and affiliation`**

- **Owner**: filled in by GenePattern - your login name
- **Privacy**: if you mark **`private`**, then only you (as identified by your login name) will be able to view it. If it is marked **`public`**, all users connecting to this GenePattern server will have access to it. Select **`private`**.
- **Quality level**: this is a measure of the robustness of this module, as determined by the author. Filling this field in accurately will help other users to know how confident they can be in the results produced by this module. The choices are **`development`**, **`preproduction`**, and **`production`**. Select **`preproduction`**.
- **Command line**: this is the most important part of the task integrator. GenePattern provides several conventions that allow you to run your code without needing to explicitly specify the location of a Perl interpreter, Java JVM, or R interpreter:
    - The keyword **`<libdir>`** indicates the pathname to your program. Because the GenePattern server will place the module in a location whose path may change, your module should not assume a pathname to an interpreter or executable. For instance, you must specify an interpreter on the command line (for example "perl myscript" or "java my jarfile") rather than embedding them in the script itself, as in "myscript.pl", where the first line of the script is **`#!/usr/bin/perl`**).
    - The keywords **`<perl>`**, **`<java>`**, and **`<R>`** refer to the Java, Perl, and R installations that were installed with GenePattern. If the module you are adding is Java, Perl, or R, you can use these keywords in the command line. If your module is in executable format, such as compiled C code, you do not need to specify a keyword. If your module is a script that must be run by an interpreter that is not supplied in the GenePattern installation, such as Python or LISP, you must refer to it as an absolute pathname. (Eg. **`/usr/bin/python`**).
      If you have several scripts using the same language, or will be running the same language across multiple systems where it might be located in different directories, you can add a definition for the language in your GenePatternServer/resources/genepattern.properties file (eg. **`python=/usr/bin/python`**), restart the GenePattern server, and then refer to it in the command line as **`<python> foo.py`**.
    - Other items in brackets indicate the names that you are giving to each parameter. In this example, there are two parameters: the input file (the value after the **`-F`** option) and the output file (the value after the **`-o`** option). These appear as **`<input.filename>`** and **`<output.file>`**. The names you give to the parameters here will be the names they are given in the GenePattern client interfaces.

  To add the command line for this module, cut and paste the following values into the command line input box:
  ```
  <perl> <libdir>log_transform.pl -F <input.filename> -o <output.file>
  ```

- **task type**: **`Preprocess & Utilities`**
- **CPU**:**`Any`**
- **Operating system**:**`Any`**
- **Language**:**`Perl`**
- **Version comment**: this is where you describe what is special or interesting about this version of the module.
- **output description**: select all file formats output by your module. In this case choose **`gct`**. Only file formats which are output for the module are acceptable for input to the module. Other modules may output several different files. In that case you'll need to shift-click or option-click or apple-click to get the appropriate selections.

Your screen should resemble the screen shown below:


add GenePattern task - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
GenePattern
sign out gpuser@broad.mit.edu
about
pipeline
task
run
edit
Filter by Suite
search

Create new GenePattern task

Please enter the following information to submit a new or updated analysis task to GenePattern. help

Name: LogTransform * (required, no spaces) task catalog
LSID:
Description: Log transform a gct file
Author: George P. User, Broad (name, affiliation)
Owner: gpuser@broad.mit.edu (email address)
Privacy: private
Quality level: preproduction

Use <java> for launching a JVM, <libdir> for accessing EXEs, DLLS, JARs, etc.,
<*your_param_name*> to substitute your own parameters (listed below),
<*java.system.property.name*> to substitute from java.lang.System.getProperties().
You may also use environment variables and settings from your GenePatternServer/resources/genepattern.properties file.
Useful ones: <path.separator>, <file.separator>, <os.name>, <perl>, <java>, <libdir>

command line:
```
<perl> <libdir>log_transform.pl -F <input.filename> -o
<output.file>
```
* (required)

task type: Preprocess & Utilities new...
CPU type: any (if compiled for a specific one)
operating system: any (if operating system-dependent)
Language: Perl **min. language version:**
Version comment: Initial draft from tutorial

output description: output file format(s): eps, gct, Gene List new...

Support files: The actual program plus any required libraries will will be accessible to your command line as <libdir>*filename*
(jar, dll, exe, pl, doc, etc.) Browse...

The next step is to upload the actual module and support files:

- On the first line in the Support Files section, click **`Browse...`**
- Navigate to **`gp_tutorial_files/log_transform`**.
- Select the file **`log_transform.pl`** and click **`Open`**
- On the second line in the Support Files section, click **`Browse...`**
- Select the file **`LogTransform.pdf`** and click **`Open`**. This is the documentation file for the LogTransform module. GenePattern automatically recognizes documentation extensions such as .pdf .doc, and .txt and will launch a file with this extension when the user clicks the **`Help`** button in a GenePattern client.

The input fields below the command line allow you to describe each parameter and set its default values if needed. This is the information that GenePattern uses to display information for an analysis module. The **`prefix when specified`** field is intended to permit optional flags. If the value you entered in **`name`** requires an element to precede it in the command line, you may enter that in the **`prefix when specified`** field.

In this example you will enter descriptions for the two parameters for **`log_transform`**, which are **`input.filename`** and **`output.file`**.

- On the first line in the parameters section, type the following information:
  - **name**: **`input.filename`**
  - **description**: **`The dataset to be transformed (gct format)`**
  - **type**: choose **`input file`**
  - **fileFormat**: choose **`gct`**
- On the next line in the parameters section, type the following information:
  - **name**: **`output.file`**
  - **description**: **`The name of the new transformed file`**

    - **type**: because this is a name for a user to fill in, choose `text`
- Click `save`.

You will see a message informing you that the task has been saved. To confirm that the task has been added to the GenePattern server, do the following:

- Launch the Java client. If your Java client is already running, you can select the `File->Refresh->Modules` menu item.
- Select the `Analysis->Preprocess & Utilities` menu
- Confirm that `LogTransform` is in the list of modules.

Notes:

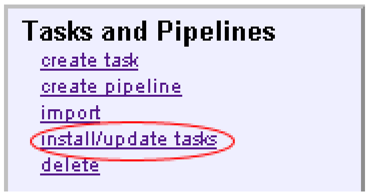
!

- **A task has access to your server.** By adding a task, a user is able to execute arbitrary code on the server which could take the form of malicious code. Virus scanner software would be some protection. So would running under an appropriately privileged (ie. non-root) user.
- **All arguments are passed as strings.** Tasks receive all arguments as strings, even if they are apparently numeric. Methods that are expecting numbers need to convert them explicitly (eg. as.integer(arg)).
- **Windows forbidden filenames.** Machines running Windows cannot accept files with names that consist of the following, regardless of the file extension: `con, prn, aux, nul, com1, com2, com3, com4, lpt1, lpt2, lpt3`. For cross-platform compatibility it is best to avoid files with these names.
- **MATLAB modules.** If you plan to write modules in MATLAB, review the notes on coding MATLAB modules in MATLAB Notes.

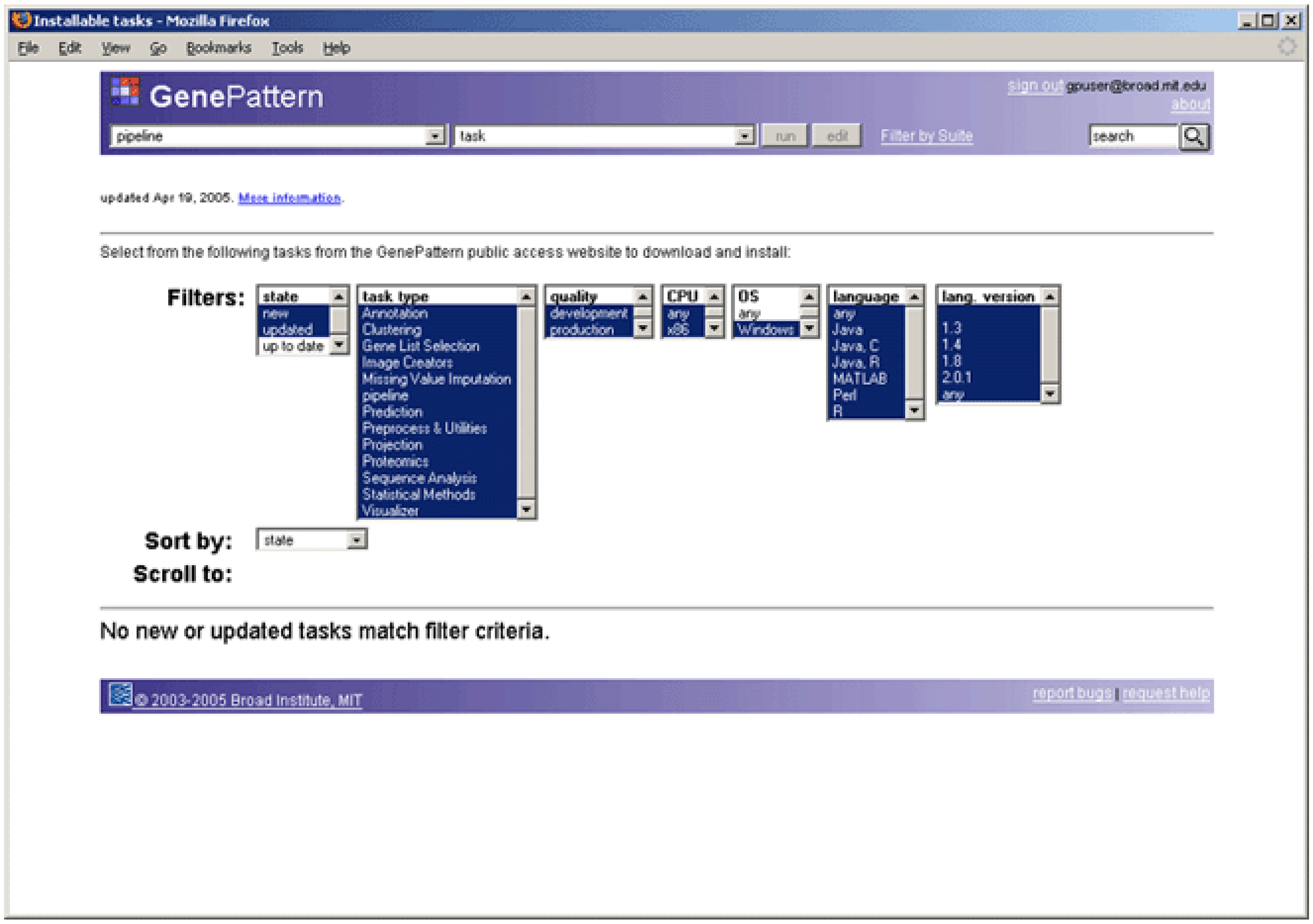
Exercise 5b: Installing/Updating a module

GenePattern has the capability to automatically detect new and updated modules at the Broad Institute's repository. This exercise will demonstrate how to add and update modules from that repository.
To add and update Broad modules:

- Navigate to the GenePattern home page.
- Under the `Tasks and Pipelines` section in the right column, click on `install/update task`.



You will see a message indicating that GenePattern is fetching tasks from the catlog. After a few moments, you will see a screen that resembles the following screen.



updated Apr 19, 2005. More information.

Select from the following tasks from the GenePattern public access website to download and install:

No new or updated tasks match filter criteria.

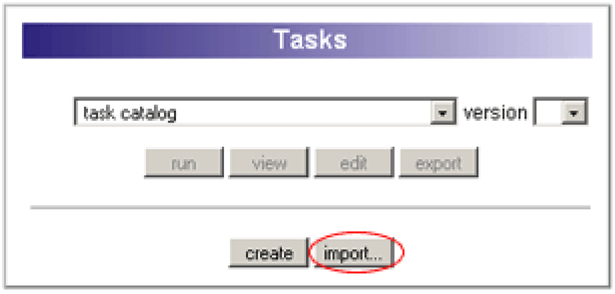
© 2003-2005 Broad Institute, MIT

If there are new tasks they will be listed at the bottom of the page with an **`install checked`** button. This interface provides a variety of filtering options but in general most users will want to click **`install checked`**. This will install the latest modules from the Broad Institute.

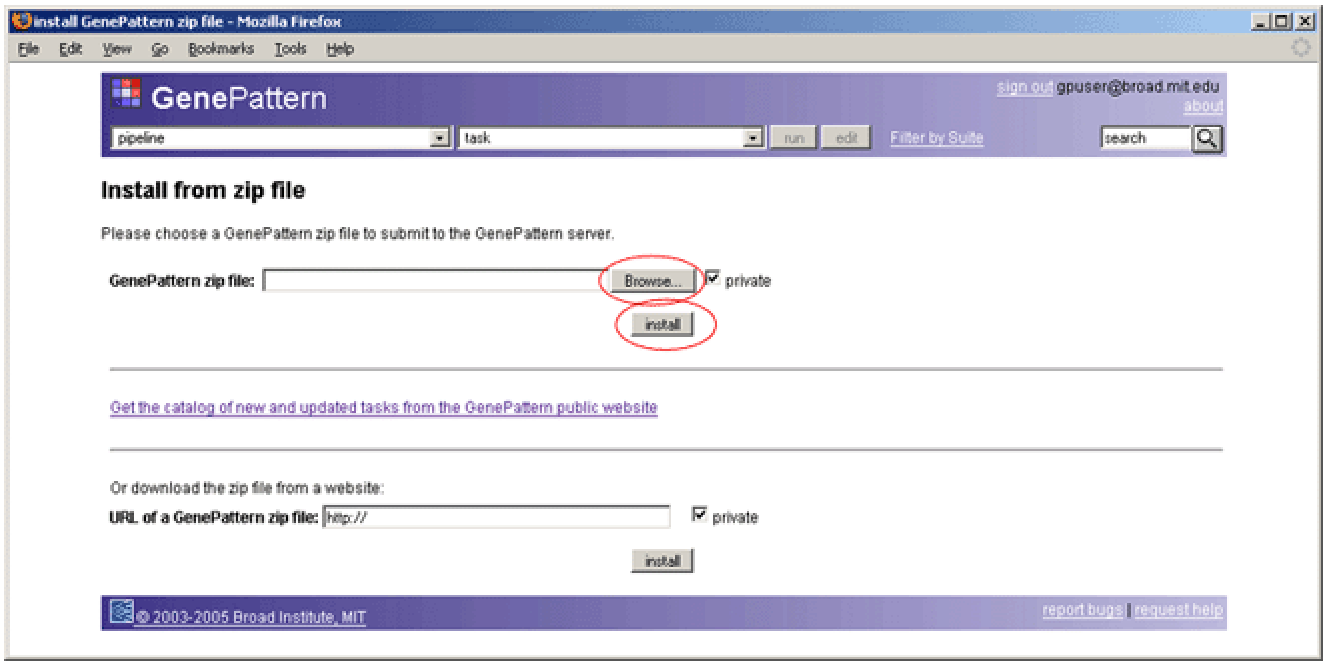
Exercise 5c: Importing modules as zip files

To import a task:

- Navigate to the GenePattern home page.
- Select **`import`** from the **`Tasks`** section.



- You will see a page, shown below, that prompts you to either upload a Zip file or enter it as a URL.
- To upload a zipped task, click the **`Browse`** button.
- Navigate to your .zip file.
- Click the **`install`** button.


Install GenePattern zip file - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
GenePattern
sign out gpuser@broad.mit.edu
about
pipeline
task
run
edit
Filter by Suite
search

Install from zip file

Please choose a GenePattern zip file to submit to the GenePattern server.

GenePattern zip file: [Browse...] [✓] private

[install]

Get the catalog of new and updated tasks from the GenePattern public website

Or download the zip file from a website:

URL of a GenePattern zip file: http:// [✓] private

[install]

© 2003-2005 Broad Institute, MIT
report bugs | request help

You will see a message that the module has been successfully installed.

! Currently, the Task Integrator can only export and import modules whose support files do not contain subdirectories.

This same process can be used to import pipelines and suites.

- You can import a pipeline in one of two ways: in the Tasks section of the GenePattern home page, click `import`, or under `Tasks and Pipelines` in the right column of the GenePattern home page, click `import`.
- To import a suite, under `Suites` in the right column of the GenePattern home page, click `import`.

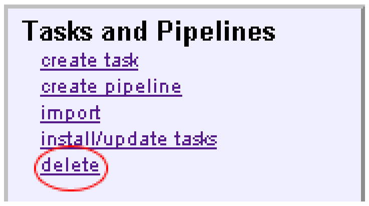
Deleting Modules

Because reproducible research creates a new copy of a task or pipeline whenever any aspect of it is changed, eventually there may be a number of obsolete tasks and pipelines residing on your server. To keep tasks up-to-date GenePattern provides an interface for deleting the old ones.

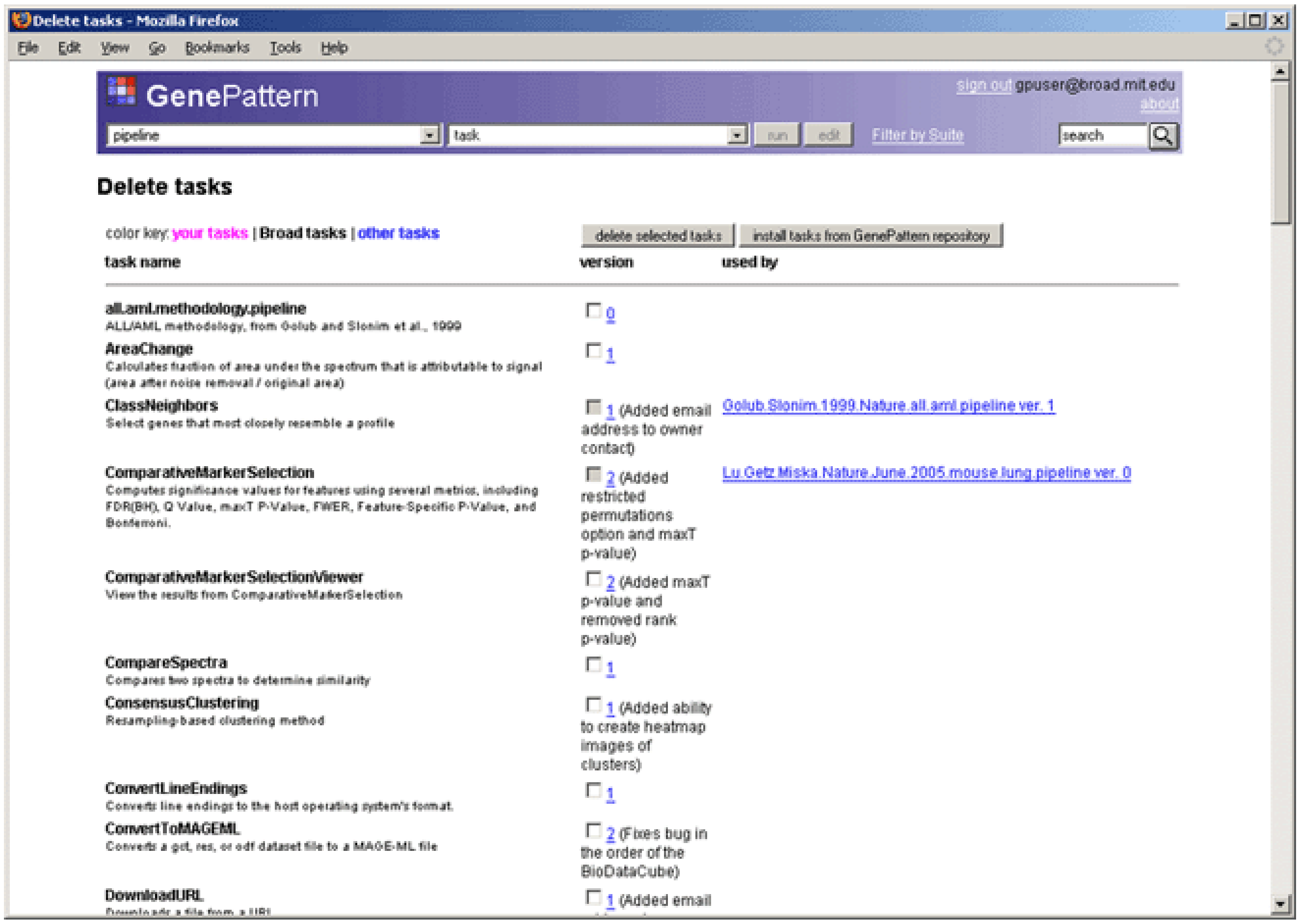
Exercise 5d: Deleting a module

In this exercise you will learn how to remove a module from a GenePattern server:

- Navigate to the GenePattern home page.
- Under the `Tasks and Pipelines` section in the right column, click `delete`.



You will see a page listing all installed tasks, color-coded by provenance.


Delete tasks - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
GenePattern
sign out gpuser@broad.mit.edu
about
pipeline
task
run
edit
Filter by Suite
search

Delete tasks

color key: your tasks | Broad tasks | other tasks

delete selected tasks | install tasks from GenePattern repository

task name	version	used by
all.aml.methodology.pipeline ALL/AML methodology, from Golub and Slonim et al., 1999	☐ 0	
AreaChange Calculates fraction of area under the spectrum that is attributable to signal (area after noise removal / original area)	☐ 1	
ClassNeighbors Select genes that most closely resemble a profile	☐ 1 (Added email address to owner contact)	Golub.Slonim.1999.Nature.all.aml.pipeline ver. 1
ComparativeMarkerSelection Computes significance values for features using several metrics, including FDR(BH), Q Value, maxT P-Value, FWER, Feature-Specific P-Value, and Bonferroni.	☐ 2 (Added restricted permutations option and maxT p-value)	Lu.Getz.Miska.Nature.June.2005.mouse.lung.pipeline ver. 0
ComparativeMarkerSelectionViewer View the results from ComparativeMarkerSelection	☐ 2 (Added maxT p-value and removed rank p-value)	
CompareSpectra Compares two spectra to determine similarity	☐ 1	
ConsensusClustering Resampling-based clustering method	☐ 1 (Added ability to create heatmap images of clusters)	
ConvertLineEndings Converts line endings to the host operating system's format.	☐ 1	
ConvertToMAGEML Converts a gct, res, or odf dataset file to a MAGE-ML file	☐ 2 (Fixes bug in the order of the BioDataCube)	
DownloadURL	☐ 1 (Added email	

- Scroll through the list and click the radio buttons next to the version number of a module you'd like to delete.

  Note the text on the right-hand side. This is the text entered in the **`Version comment`** box on the Pipeline Designer and Task Integrater pages.

- To delete the modules, click on **`delete selected tasks`**.
- You will see a confirmation dialog box asking you to confirm the deletion. Click `OK`.

You will see a message informing you that the versions you indicated are deleted from the server.

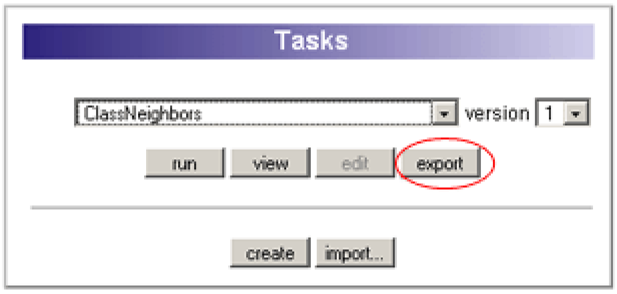
Sharing Modules

GenePattern allows you to package a module, with all of its necessary files, into a single zip file which can then be installed on another GenePattern server. In the following exercise you will learn how to export GenePattern tasks.

Exercise 5e: Exporting modules as zip files

To export a task:

- Navigate to the GenePattern home page.
- In the **`Tasks`** section, select the task you wish to share from the drop-down menu.
- Click the **`export`** button, shown below.
- You will see a dialog box prompting you to save the file. The file you save will be a Zip file that you can then send to another GenePattern user.



Using Suites in the GenePattern Web Client

This section describes how to work with suites in the GenePattern Web Client. If you are unfamiliar with GenePattern suites, see Suites in the GenePattern Graphical Environment tutorial.

Creating Suites

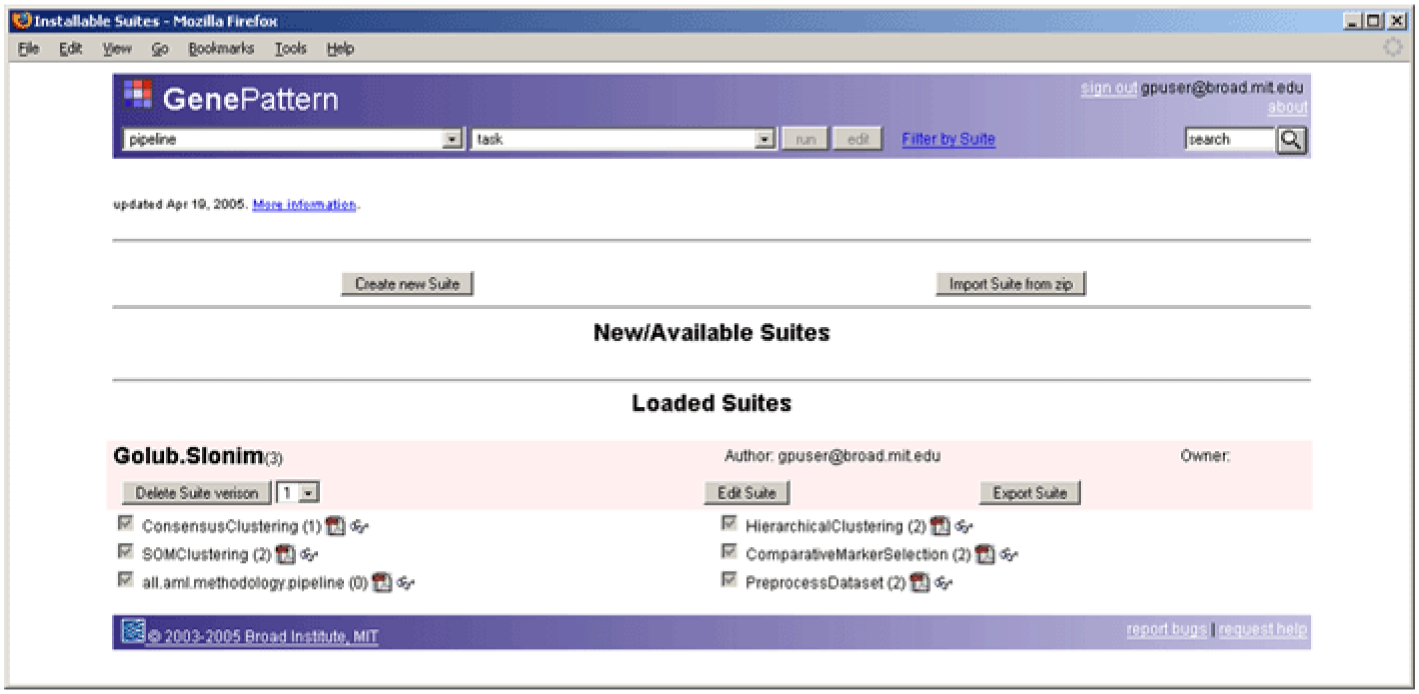
To create a suite:

- Navigate to the GenePattern home page.
- Under the **`Suites`** section in the right column, click **`create suite`**.
- When the Create New GenePattern Suite window appears, define your new suite.
- Click **`save`** at the bottom of the window.

Managing Suites

The Manage Suites window allows you to perform all suite functions: create, import, install, delete, edit, and export. To display the Manage Suites window:

- Navigate to the GenePattern home page.
- Under the **`Suites`** section in the right column, click **`install/update suites`** or **`delete suites`**.


Installable Suites - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
GenePattern
sign out gpuser@broad.mit.edu
about
pipeline
task
run
edit
Filter by Suite
search
updated Apr 19, 2005. More information.
Create new Suite
Import Suite from zip
New/Available Suites
Loaded Suites
Golub.Slonim(3)
Author: gpuser@broad.mit.edu
Owner:
Delete Suite version
1
Edit Suite
Export Suite
ConsensusClustering (1)
SOMClustering (2)
all.aml.methodology.pipeline (0)
HierarchicalClustering (2)
ComparativeMarkerSelection (2)
PreprocessDataset (2)
© 2003-2005 Broad Institute, MIT
report bugs | request help

- The first section in the window allows you to create and import suites.
  - To create a suite, click **`Create New Suite`**. When the Create Suite window appears, define the suite.
  - To import a suite, click **`Import Suite from zip`**. The Import window appears, as described in Importing Modules.
- The New/Available Suites section lists new and updated suites available from the Broad Institute. To install the suites listed here, click the **`install checked`** button. (In the example screen, there are no new suites available and, therefore, no **`install checked`** button.)
- The Loaded Suites section lists the suites available on your server.
  - To delete a suite, select the version to delete from the drop-down menu and click **`delete suite version`**.
  - To edit a suite, click **`Edit Suite`**. When the Create Suite window appears, edit the suite.
  - To export a suite into a zip file, click **`Export Suite`**. The system creates a zip file, then prompts you for a directory location and file name.

Using Pipelines in the GenePattern Web Client

This section describes how to work with pipelines in the GenePattern Web Client. If you are unfamiliar with GenePattern pipelines, see the overview information presented in the tutorial for the GenePattern Graphical Environment (Pipelines).

NOTES

- To run the exercises in this section, the GenePattern server and Web Client must be running. If you have not launched them, you can follow the directions for launching the GenePattern server and Web Client in Getting Started.
- The exercises require the GenePattern tutorial datasets. If you did not download them during installation, you can download them here or from the datasets page, http://www.broad.mit.edu/cancer/software/genep attern/datasets. You will need to use a compression utility such as WinZip (Windows) or StuffIt Expander (Mac) to extract the datasets.

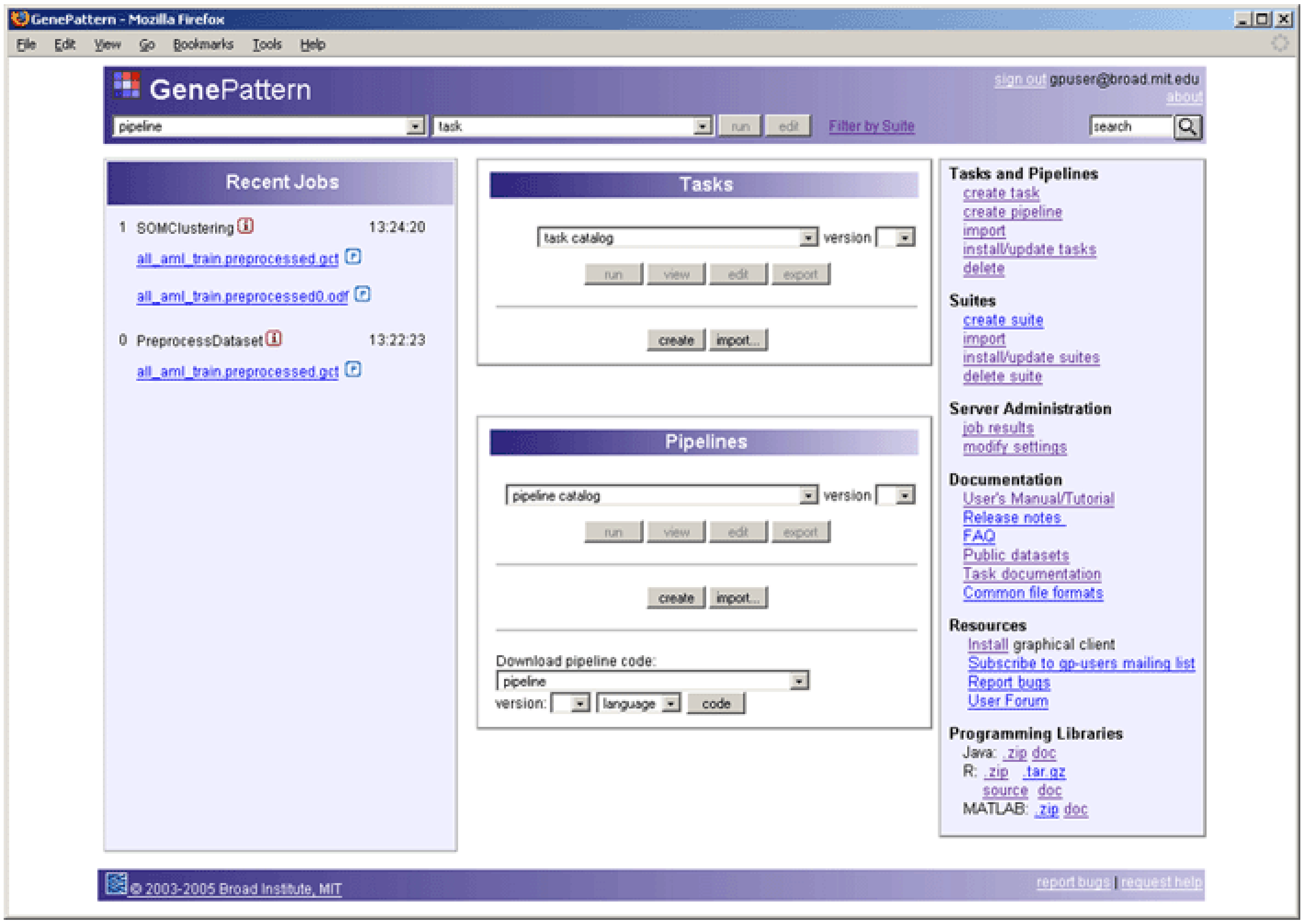
In the following exercises, you will learn how to create, run, and edit pipelines using the GenePattern Web Client.

If you have the GenePattern Web Client running, navigate to the GenePattern home page. If not, start the Web Client:

- Double-click the *GenePatternHome.html* icon (shown below).
- You will see your server's GenePattern home page launch in your Web browser.

GenePatternHome.html

- When the browser launches, you will see a prompt for a username. Type in your email address.



gpuser@broad.mit.edu

pipeline | task

1	SOMClustering	13:24:20
	all_aml_train.preprocessed.gct	
	all_aml_train.preprocessed0.odf	
0	PreprocessDataset	13:22:23
	all_aml_train.preprocessed.gct	

task catalog version

pipeline catalog version

Download pipeline code:
pipeline
version: language

Tasks and Pipelines
create task
create pipeline
import
install/update tasks
delete

Suites
create suite
import
install/update suites
delete suite

Server Administration
job results
modify settings

Documentation
User's Manual/Tutorial
Release notes
FAQ
Public datasets
Task documentation
Common file formats

Resources
Install graphical client
Subscribe to gp-users mailing list
Report bugs
User Forum

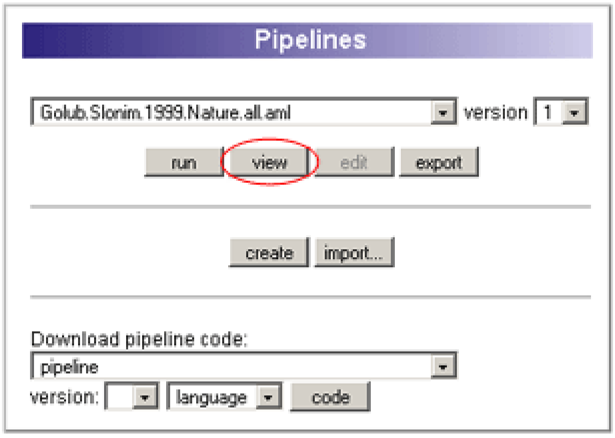
Programming Libraries
Java: .zip doc
R: .zip .tar.gz
source doc
MATLAB: .zip doc

© 2003-2005 Broad Institute, MIT
report bugs | request help

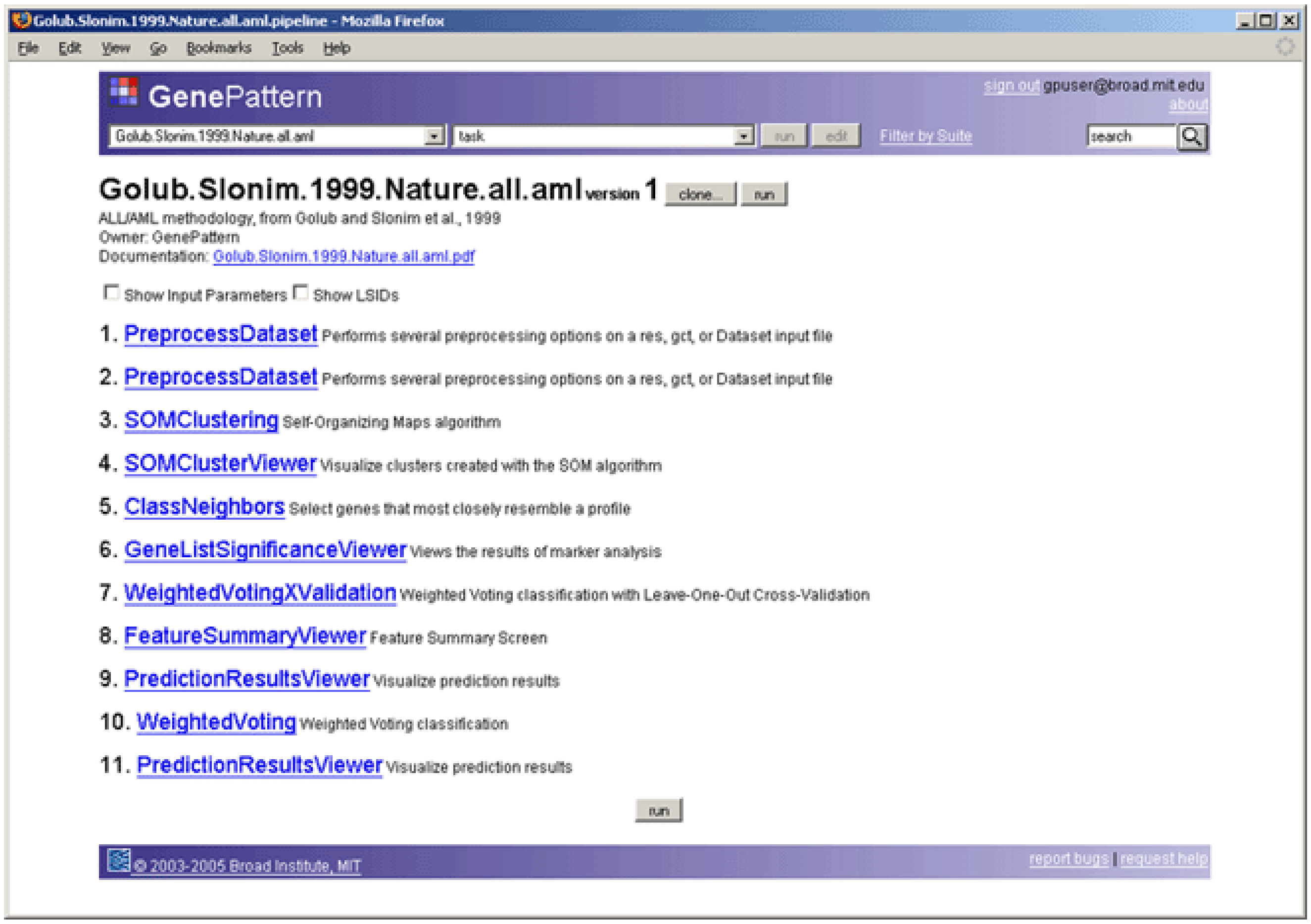
Exercise 6a: Running a pipeline

In this exercise, you will run the Golub.Slonim.1999.Nature.all.aml pipeline. To do so:

- In the **`Pipelines`** section, select **`Golub.Slonim.1999.Nature.all.aml`** from **`Pipeline Catalog`** drop-down menu.
- Select **`1`** from the **`version`** drop-down menu.
- Click the **`view`** button.



You will see a stage-by-stage representation of the pipeline.


Golub.Slonim.1999.Nature.all.aml.pipeline - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
GenePattern
sign out gpuser@broad.mit.edu
about
Golub.Slonim.1999.Nature.all.aml
task
run
edit
Filter by Suite
search
clone...
run
© 2003-2005 Broad Institute, MIT
report bugs | request help

Golub.Slonim.1999.Nature.all.aml version 1

ALL/AML methodology, from Golub and Slonim et al., 1999
Owner: GenePattern
Documentation: Golub.Slonim.1999.Nature.all.aml.pdf

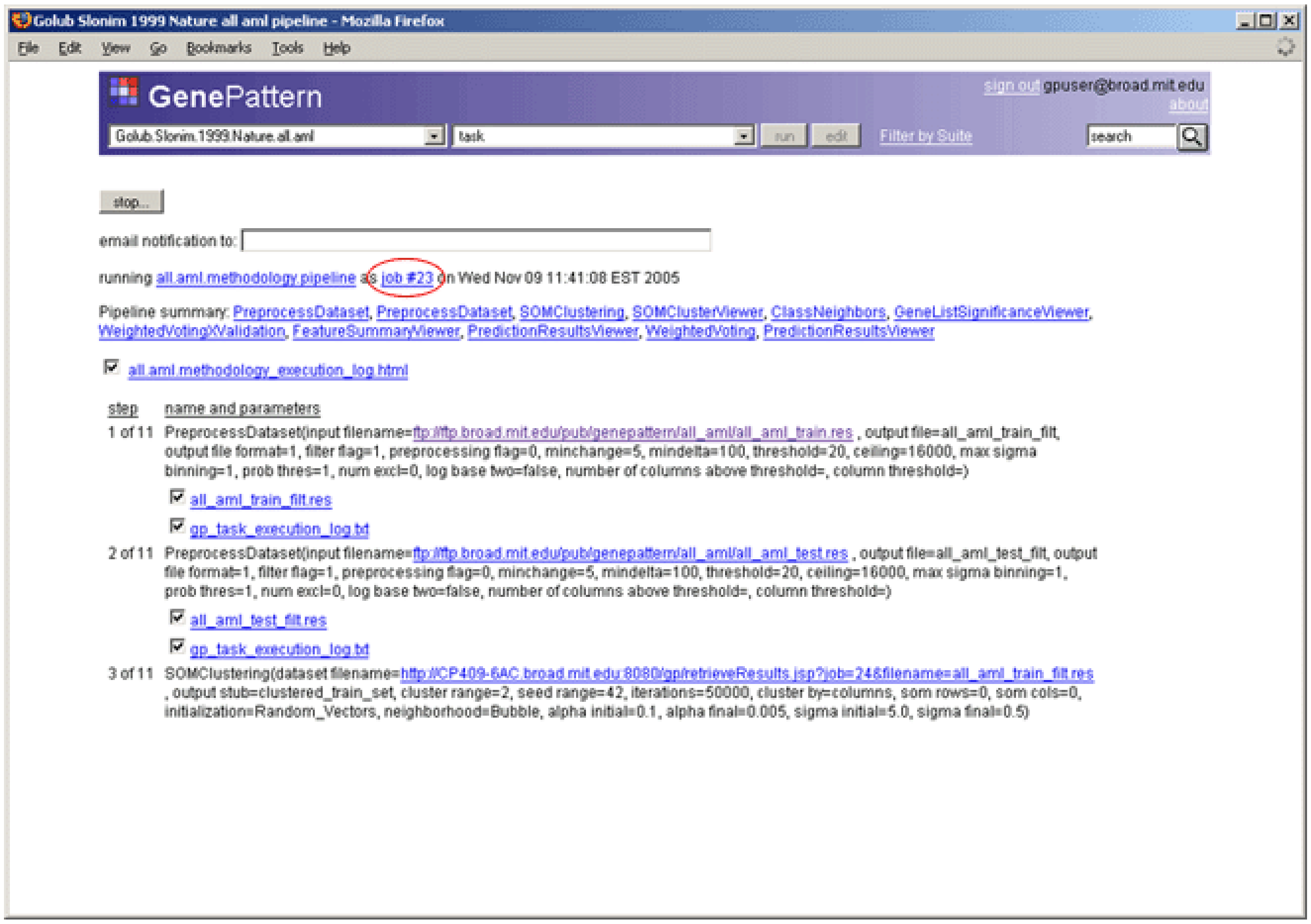
☐ Show Input Parameters ☐ Show LSIDs

1. PreprocessDataset Performs several preprocessing options on a res, gct, or Dataset input file
2. PreprocessDataset Performs several preprocessing options on a res, gct, or Dataset input file
3. SOMClustering Self-Organizing Maps algorithm
4. SOMClusterViewer Visualize clusters created with the SOM algorithm
5. ClassNeighbors Select genes that most closely resemble a profile
6. GeneListSignificanceViewer Views the results of marker analysis
7. WeightedVotingXValidation Weighted Voting classification with Leave-One-Out Cross-Validation
8. FeatureSummaryViewer Feature Summary Screen
9. PredictionResultsViewer Visualize prediction results
10. WeightedVoting Weighted Voting classification
11. PredictionResultsViewer Visualize prediction results

You may select the option to **`Show Input Parameters`** and **`Show LSIDs`**. These checkboxes allow you to investigate the properties of the pipeline without risk of modifying it. If you wanted to make a new pipeline, based on this pipeline from the Broad Institute, you would click **`clone`** at the top.

- Click **`run`** to start the pipeline running on the server.

As the pipeline starts to run, results appear in your browser window. Every task and pipeline you run on GenePattern has an associated *Job ID*. The Job ID is displayed at the top of the browser window, as shown below. You can use the Job ID to refer to your pipeline in future actions. As each task in the pipeline completes, its results appear in the browser window. You can download the result files separately by shift-clicking on their links, or you can check the checkboxes next to them and download a large number of result files as a single zip file.


Golub Slonim 1999 Nature all aml pipeline - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
GenePattern
sign out gpuser@broad.mit.edu
about
Golub.Slonim.1999.Nature.all.aml
task
run
edit
Filter by Suite
search
stop...

email notification to:

running all.aml.methodology.pipeline as job #23 on Wed Nov 09 11:41:08 EST 2005

Pipeline summary: PreprocessDataset, PreprocessDataset, SOMClustering, SOMClusterViewer, ClassNeighbors, GeneListSignificanceViewer, WeightedVotingXValidation, FeatureSummaryViewer, PredictionResultsViewer, WeightedVoting, PredictionResultsViewer

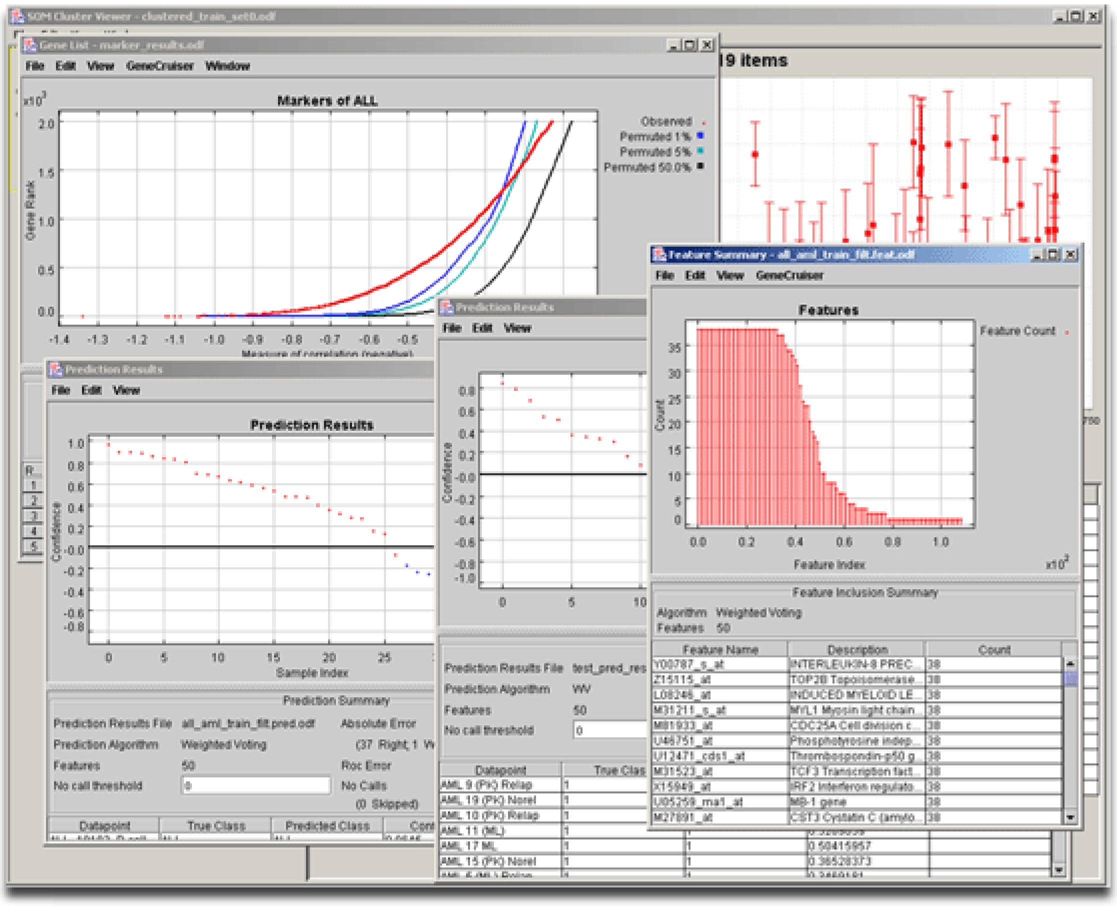
☑ all.aml.methodology_execution_log.html

step	name and parameters
1 of 11	PreprocessDataset(input filename=ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_train.res , output file=all_aml_train_filt, output file format=1, filter flag=1, preprocessing flag=0, minchange=5, mindelta=100, threshold=20, ceiling=16000, max sigma binning=1, prob thres=1, num excl=0, log base two=false, number of columns above threshold=, column threshold=) ☑ all_aml_train_filt.res ☑ gp_task_execution_log.txt
2 of 11	PreprocessDataset(input filename=ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_test.res , output file=all_aml_test_filt, output file format=1, filter flag=1, preprocessing flag=0, minchange=5, mindelta=100, threshold=20, ceiling=16000, max sigma binning=1, prob thres=1, num excl=0, log base two=false, number of columns above threshold=, column threshold=) ☑ all_aml_test_filt.res ☑ gp_task_execution_log.txt
3 of 11	SOMClustering(dataset filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=24&filename=all_aml_train_filt.res , output stub=clustered_train_set, cluster range=2, seed range=42, iterations=50000, cluster by=columns, som rows=0, som cols=0, initialization=Random_Vectors, neighborhood=Bubble, alpha initial=0.1, alpha final=0.005, sigma initial=5.0, sigma final=0.5)

To stop a pipeline before it completes, click the **stop** button.

If the pipeline is going to take a long time, you can type your email address into the email box. You will be notified when your pipeline is finished.

As the pipeline completes, you will see several visualizer windows open (shown below) to display results from different stages.


SOM Cluster Viewer - clustered_train_set0.odf
Gene List - marker_results.odf
File Edit View GeneCruiser Window
Markers of ALL
Observed
Permuted 1%
Permuted 5%
Permuted 50.0%
Gene Rank
Measure of correlation (negative)
19 items
Feature Summary - all_aml_train_filt.feat.odf
File Edit View GeneCruiser
Features
Feature Count
Count
Feature Index
Prediction Results
File Edit View
Prediction Results
Confidence
Sample Index

Prediction Summary

Prediction Results File	all_aml_train_filt.pred.odf	Absolute Error	
Prediction Algorithm	Weighted Voting	(37 Right, 1 W	
Features	50	Roc Error	
No call threshold	0	No Calls	
		(0 Skipped)	

Datapoint	True Class	Predicted Class	Conf

Prediction Results File	test_pred_res
Prediction Algorithm	WV
Features	50
No call threshold	0

Datapoint	True Clas		
AML 9 (PK) Relap	1		
AML 19 (PK) Norel	1		
AML 10 (PK) Relap	1		
AML 11 (ML)	1	1	
AML 17 ML	1	1	0.50415957
AML 15 (PK) Norel	1	1	0.38528373

Feature Inclusion Summary

Algorithm Weighted Voting
Features 50

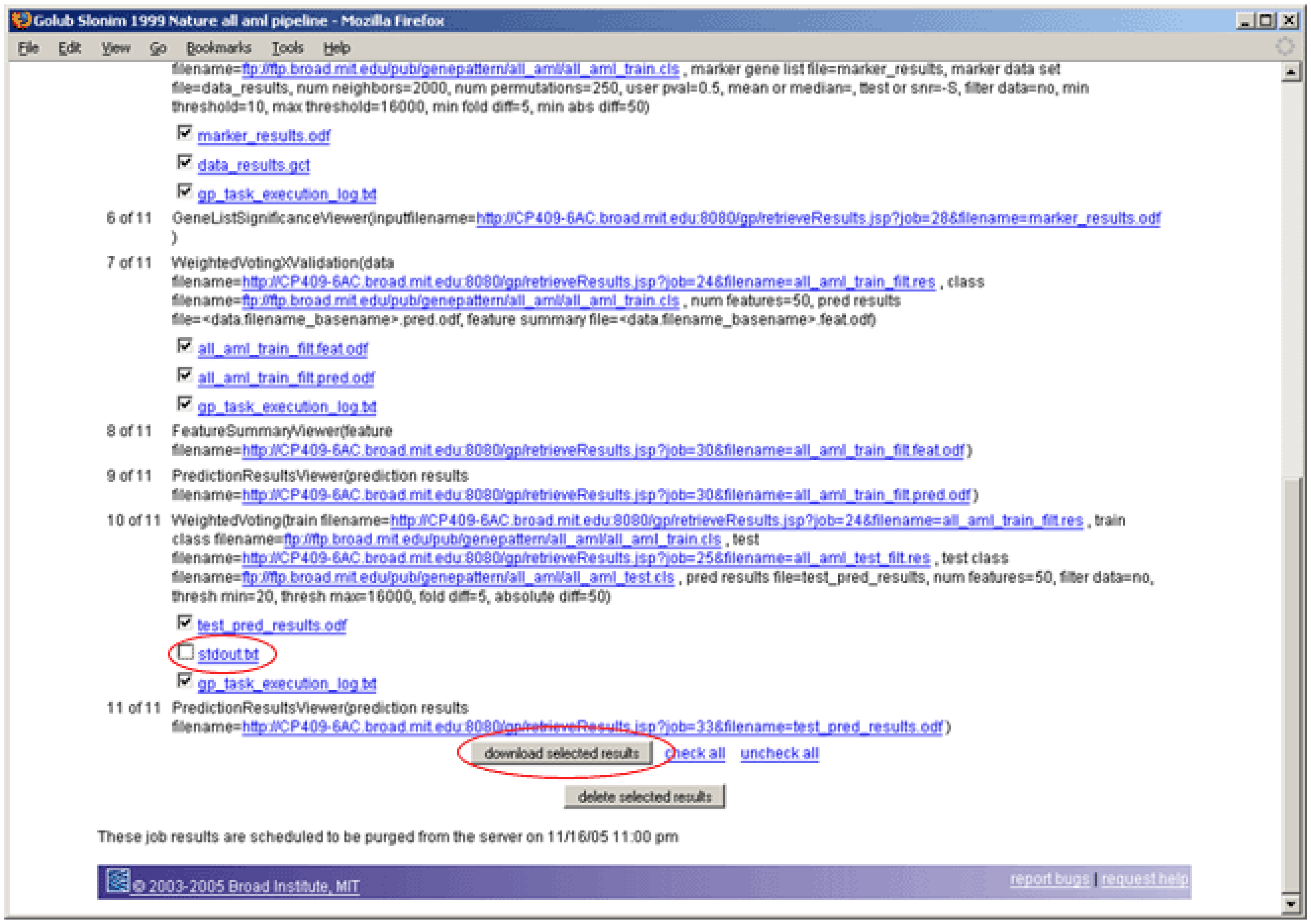
Feature Name	Description	Count
Y00787_s_at	INTERLEUKIN-8 PREC...	38
Z15115_at	TOP2B Topoisomerase...	38
L08246_at	INDUCED MYELOID LE...	38
M31211_s_at	MYL1 Myosin light chain...	38
M81933_at	CDC25A Cell division c...	38
U46751_at	Phosphotyrosine indep...	38
U12471_cds1_at	Thrombospondin-p50 g...	38
M31523_at	TCF3 Transcription fact...	38
X15949_at	IRF2 Interferon regulato...	38
U05259_rna1_at	MB-1 gene	38
M27891_at	CST3 Cystatin C (amylo...	38

Exercise 6b: Saving Pipeline Results

Your browser now displays a list of result files which are now on the server. You can save or delete any or all of these.

In this exercise, you will save all result files except the standard output messages. To do this:

- Uncheck `stdout.txt` for pipeline stage 10.
- Click the `download selected results` button.


Golub Slonim 1999 Nature all aml pipeline - Mozilla Firefox
File Edit View Go Bookmarks Tools Help

filename=ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_train.cls , marker gene list file=marker_results, marker data set file=data_results, num neighbors=2000, num permutations=250, user pval=0.5, mean or median=, ttest or snr=-S, filter data=no, min threshold=10, max threshold=16000, min fold diff=5, min abs diff=50)

- ✓ marker_results.odf
- ✓ data_results.gct
- ✓ gp_task_execution_log.txt

6 of 11 GeneListSignificanceViewer(inputfilename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=28&filename=marker_results.odf)

7 of 11 WeightedVotingXValidation(data filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=24&filename=all_aml_train_filt.res , class filename=ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_train.cls , num features=50, pred results file=<data.filename_basename>.pred.odf, feature summary file=<data.filename_basename>.feat.odf)

- ✓ all_aml_train_filt.feat.odf
- ✓ all_aml_train_filt.pred.odf
- ✓ gp_task_execution_log.txt

8 of 11 FeatureSummaryViewer(feature filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=30&filename=all_aml_train_filt.feat.odf)

9 of 11 PredictionResultsViewer(prediction results filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=30&filename=all_aml_train_filt.pred.odf)

10 of 11 WeightedVoting(train filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=24&filename=all_aml_train_filt.res , train class filename=ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_train.cls , test filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=25&filename=all_aml_test_filt.res , test class filename=ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_test.cls , pred results file=test_pred_results, num features=50, filter data=no, thresh min=20, thresh max=16000, fold diff=5, absolute diff=50)

- ✓ test_pred_results.odf
- ☐ stdout.txt
- ✓ gp_task_execution_log.txt

11 of 11 PredictionResultsViewer(prediction results filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=33&filename=test_pred_results.odf)

These job results are scheduled to be purged from the server on 11/16/05 11:00 pm

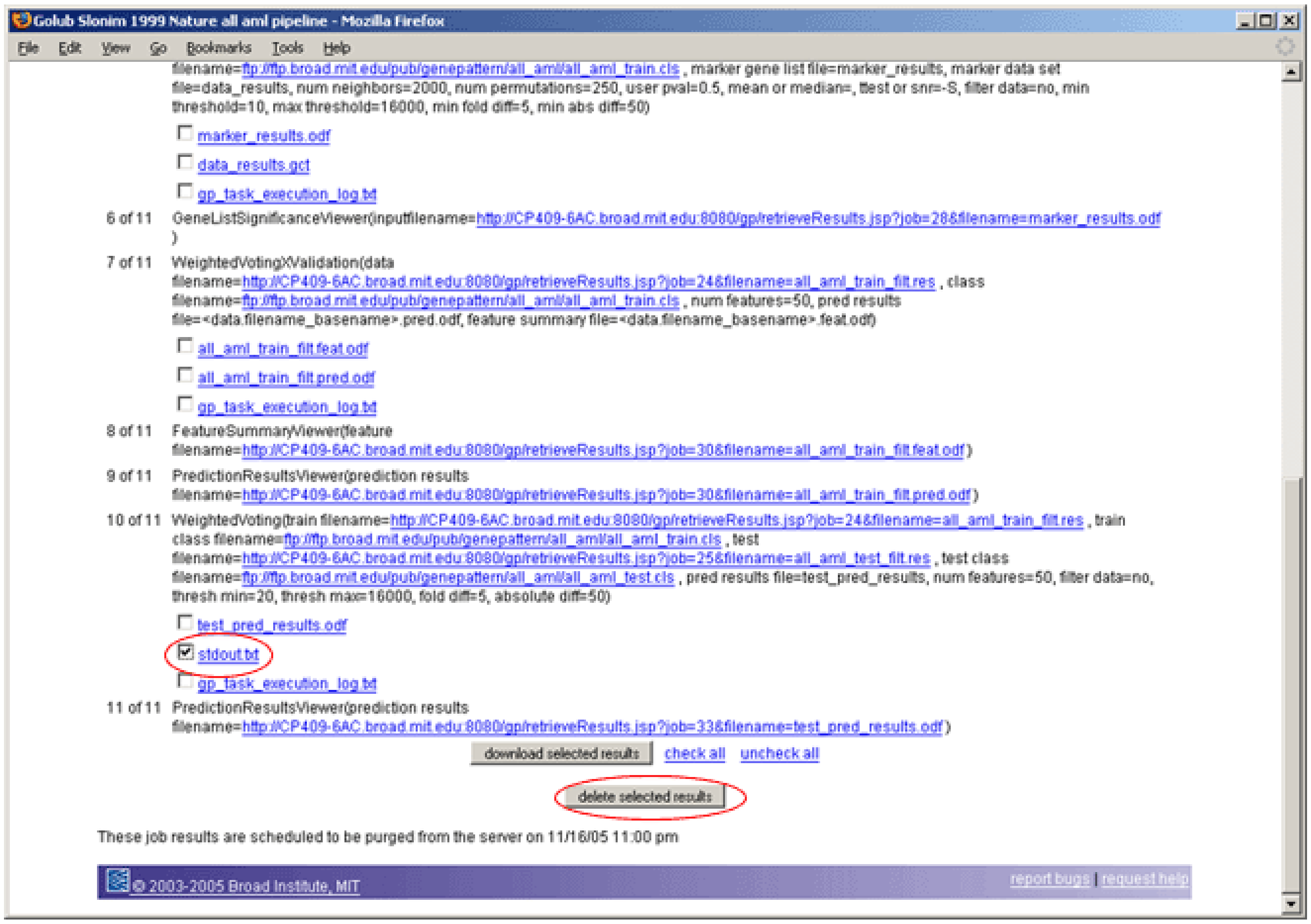
A zip file will be created that contains all checked files. You will see a dialog box prompting you to save or open the file.

- Save the zip file to your local drive.
- Double-click on the zip file you just saved. Notice that all of the files you selected have been saved in the zip file. The path information in the zip file identifies tasks by job number.

Exercise 6c: Deleting Pipeline Results

The GenePattern server automatically deletes result files at a specified interval. This is done to prevent unused result files from accumulating on the server. After you run a pipeline, a message will appear at the bottom of the pipeline results view telling you when the files will be deleted. You can also explicitly delete files by doing the following:

- Select **`Uncheck all`**.
- Check the **`stdout.txt`** file from this pipeline.
- Click on **`delete selected results`**.


Golub Slonim 1999 Nature all aml pipeline - Mozilla Firefox
File Edit View Go Bookmarks Tools Help

filename=ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_train.cls , marker gene list file=marker_results, marker data set file=data_results, num neighbors=2000, num permutations=250, user pval=0.5, mean or median=, ttest or snr=-S, filter data=no, min threshold=10, max threshold=16000, min fold diff=5, min abs diff=50)

- [ ] marker_results.odf
- [ ] data_results.gct
- [ ] gp_task_execution_log.txt

6 of 11 GeneListSignificanceViewer(inputfilename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=28&filename=marker_results.odf)

7 of 11 WeightedVotingXValidation(data filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=24&filename=all_aml_train_filt.res , class filename=ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_train.cls , num features=50, pred results file=<data.filename_basename>.pred.odf, feature summary file=<data.filename_basename>.feat.odf)

- [ ] all_aml_train_filt.feat.odf
- [ ] all_aml_train_filt.pred.odf
- [ ] gp_task_execution_log.txt

8 of 11 FeatureSummaryViewer(feature filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=30&filename=all_aml_train_filt.feat.odf)

9 of 11 PredictionResultsViewer(prediction results filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=30&filename=all_aml_train_filt.pred.odf)

10 of 11 WeightedVoting(train filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=24&filename=all_aml_train_filt.res , train class filename=ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_train.cls , test filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=25&filename=all_aml_test_filt.res , test class filename=ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_test.cls , pred results file=test_pred_results, num features=50, filter data=no, thresh min=20, thresh max=16000, fold diff=5, absolute diff=50)

- [ ] test_pred_results.odf
- [✓] stdout.txt
- [ ] gp_task_execution_log.txt

11 of 11 PredictionResultsViewer(prediction results filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=33&filename=test_pred_results.odf)

These job results are scheduled to be purged from the server on 11/16/05 11:00 pm

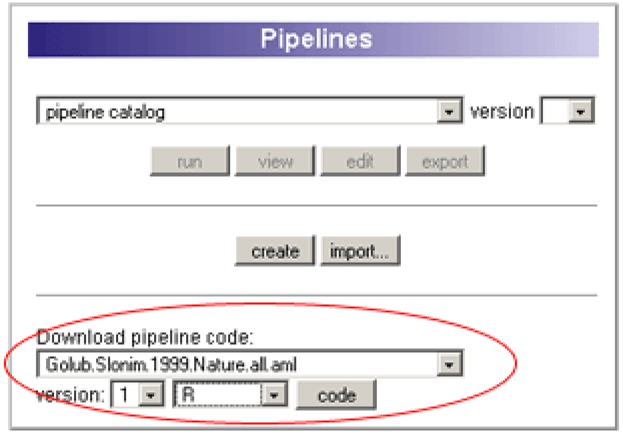
The Job Results window appears in your browser; this window is described later in Managing Jobs. Click on **GenePattern** in the header to return to the GenePattern home page.

Exercise 6d: Converting Pipelines to Code

If you want to use a pipeline as a basis for a more complex methodology, you can convert a pipeline to its equivalent in a programming language. The languages currently supported are Java, MATLAB, and R.

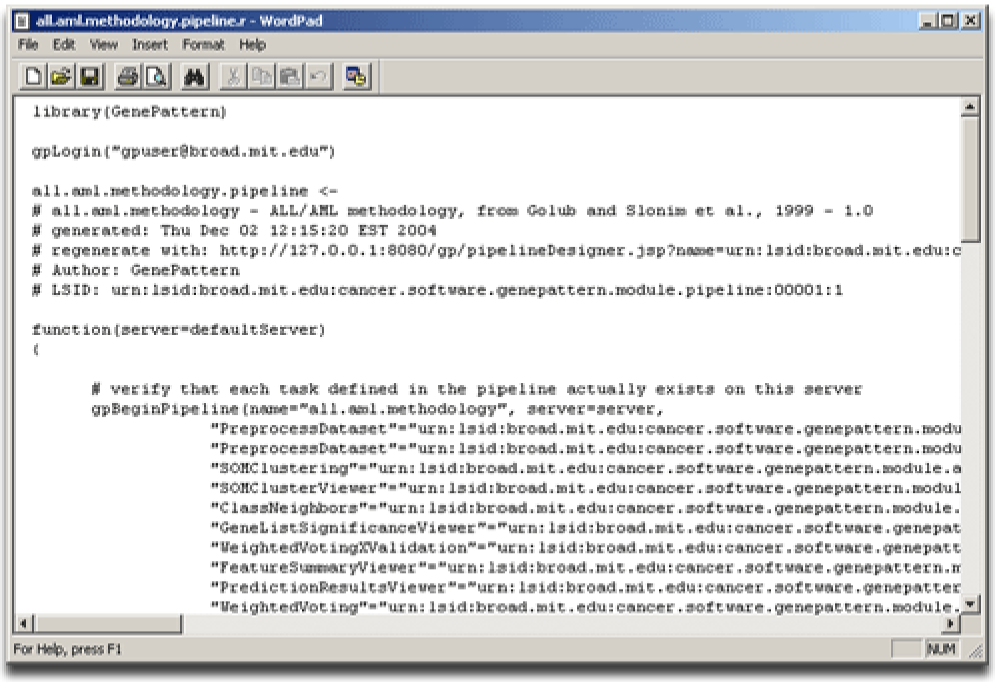
In this example, you will convert the Golub.Slonim.1999.Nature.all.aml pipeline to R:

- On the GenePattern home page, locate the **Download pipeline code** area in the Pipelines section.
- Select **Golub.Slonim.1999.Nature.all.aml** from the **pipeline** drop-down menu.
- Select **1** from the **version** drop-down menu. (By default the latest version is already selected.)
- Select **R** from the **language** drop-down menu.
- Click the **code** button.



- You will be prompted to save or open the file. Save the file to your local drive.
- To view the generated code, open the file in your preferred text editor.

The generated program, shown below, will produce the same results when run in R as it does when run in the GenePattern environment.


all.aml.methodology.pipeline.r - WordPad
File Edit View Insert Format Help
For Help, press F1
NUM

```
library(GenePattern)

gpLogin("gpuser@broad.mit.edu")

all.aml.methodology.pipeline <-
# all.aml.methodology - ALL/AML methodology, from Golub and Slonim et al., 1999 - 1.0
# generated: Thu Dec 02 12:15:20 EST 2004
# regenerate with: http://127.0.0.1:8080/gp/pipelineDesigner.jsp?name=urn:lsid:broad.mit.edu:c
# Author: GenePattern
# LSID: urn:lsid:broad.mit.edu:cancer.software.genepattern.module.pipeline:00001:1

function(server=defaultServer)
{

	# verify that each task defined in the pipeline actually exists on this server
	gpBeginPipeline(name="all.aml.methodology", server=server,
			"PreprocessDataset"="urn:lsid:broad.mit.edu:cancer.software.genepattern.modu
			"PreprocessDataset"="urn:lsid:broad.mit.edu:cancer.software.genepattern.modu
			"SOMClustering"="urn:lsid:broad.mit.edu:cancer.software.genepattern.module.a
			"SOMClusterViewer"="urn:lsid:broad.mit.edu:cancer.software.genepattern.modul
			"ClassNeighbors"="urn:lsid:broad.mit.edu:cancer.software.genepattern.module.
			"GeneListSignificanceViewer"="urn:lsid:broad.mit.edu:cancer.software.genepat
			"WeightedVotingXValidation"="urn:lsid:broad.mit.edu:cancer.software.genepatt
			"FeatureSummaryViewer"="urn:lsid:broad.mit.edu:cancer.software.genepattern.m
			"PredictionResultsViewer"="urn:lsid:broad.mit.edu:cancer.software.genepatter
			"WeightedVoting"="urn:lsid:broad.mit.edu:cancer.software.genepattern.module.
```

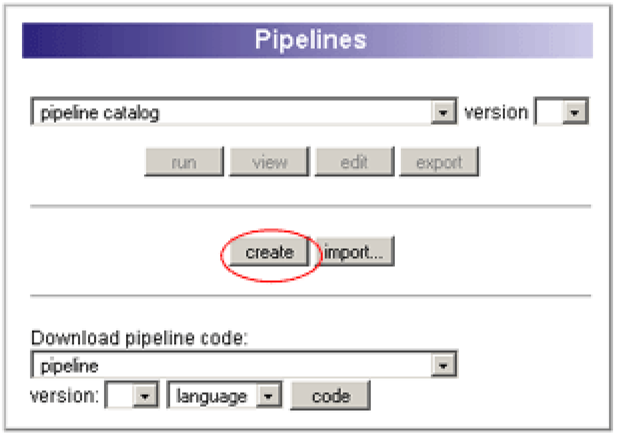
Exercise 6e: Creating a Pipeline

As described in the tutorial for the GenePattern Graphical Environment (Creating Pipelines), you can create a pipeline in two ways:

- From scratch. In this exercise, you will create a pipeline from scratch; the pipeline will do the analysis portions of the Golub.Slonim.1999.Nature.all.aml pipeline: filtering, SOM clustering, and finding markers for the resulting data.
- From a results file. To create pipeline from a results file, on the GenePattern home page, find the results file in the list of Recent Jobs, and click the pipeline icon next to that file.

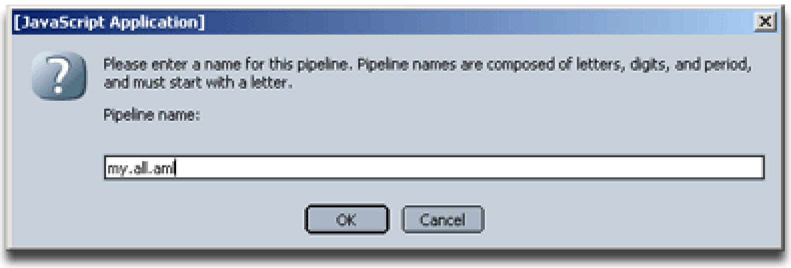
To create a pipeline from scratch:

- Navigate to the GenePattern home page.
- In the **Pipelines** section, click on **create**.



You will see a prompt for the name of the pipeline.

- Enter **my.all.aml**


[JavaScript Application]

Please enter a name for this pipeline. Pipeline names are composed of letters, digits, and period, and must start with a letter.

Pipeline name:

my.all.aml

Note, there are certain characters you shouldn't use in pipeline names. Avoid special characters such as **!,@,#,$,%,^,&,*,_**

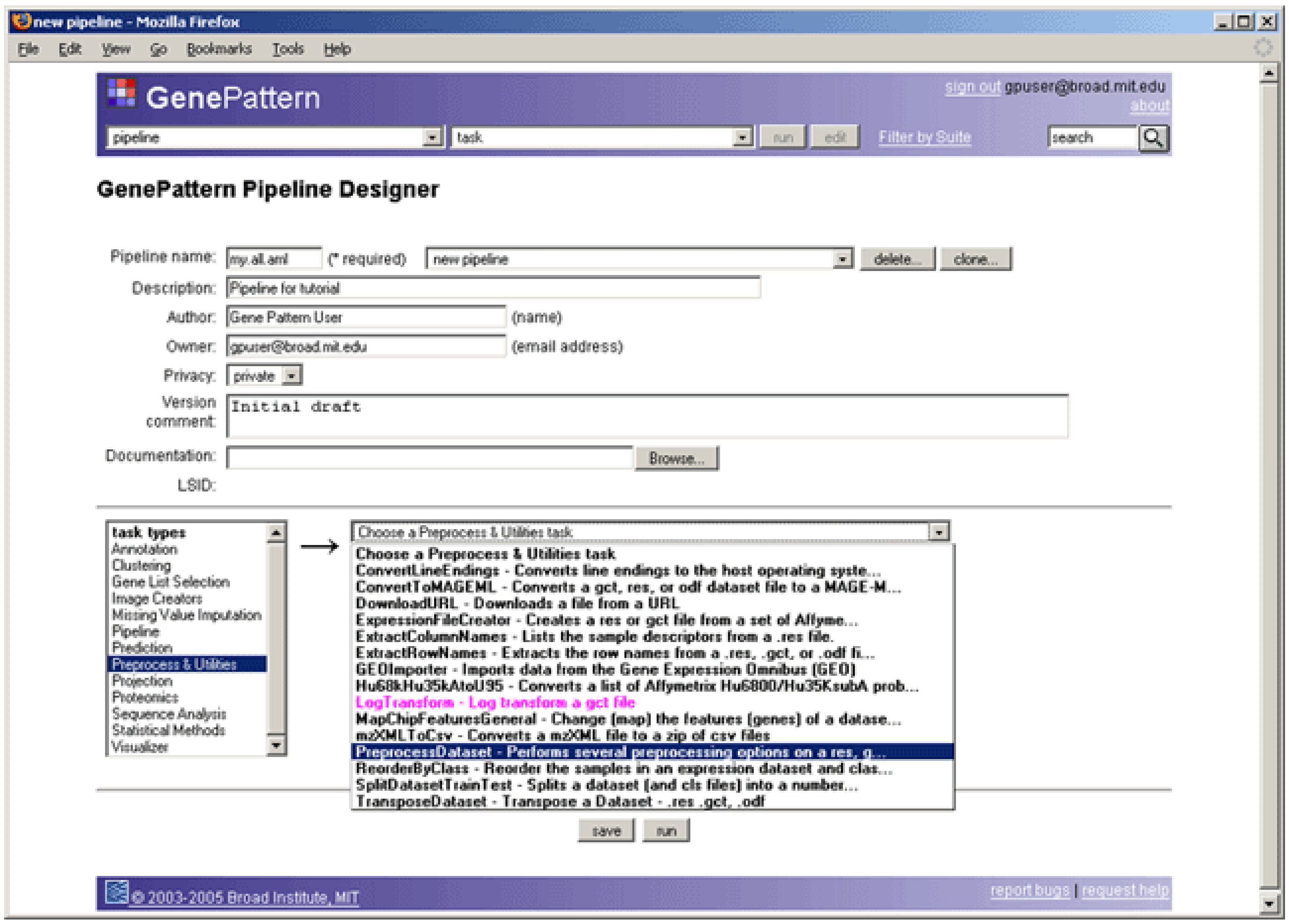
Also, machines running Windows cannot accept files with names that consist of the following, regardless of the file extension: con, prn, aux, nul, com1, com2, com3, com4, lpt1, lpt2, lpt3. For cross-platform compatibility, it is best to avoid using these names.

At the top of the pipeline designer window you will see blank spaces for authorship information and a description of the pipeline. Completing this form thoughtfully helps greatly in maintaining pipelines and ensuring that analyses are understandable and reproducible. In particular, the **Version comment** box allows you to describe this version of your pipeline. This comment will be visible on pages used to keep versions up to date, so be as complete yet concise as possible. Also note that if you choose to make this pipeline "Private" then only the user you have entered in this form (the email address) will be able to see and run this pipeline, and only when logged in with this same email address. If you choose to make it "Public" then all users will be able to see it. Finally, there is an option to upload documentation to accompany your pipeline. Documentation is strongly encouraged for "Public" pipelines.

- In the **task types** list, select **Preprocess & Utilities**.

The dropdown list on the right will be populated with all preprocessing modules.

- Choose **PreprocessDataset**.


new pipeline - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
GenePattern
sign out gpuser@broad.mit.edu
about
pipeline
task
run
edit
Filter by Suite
search

GenePattern Pipeline Designer

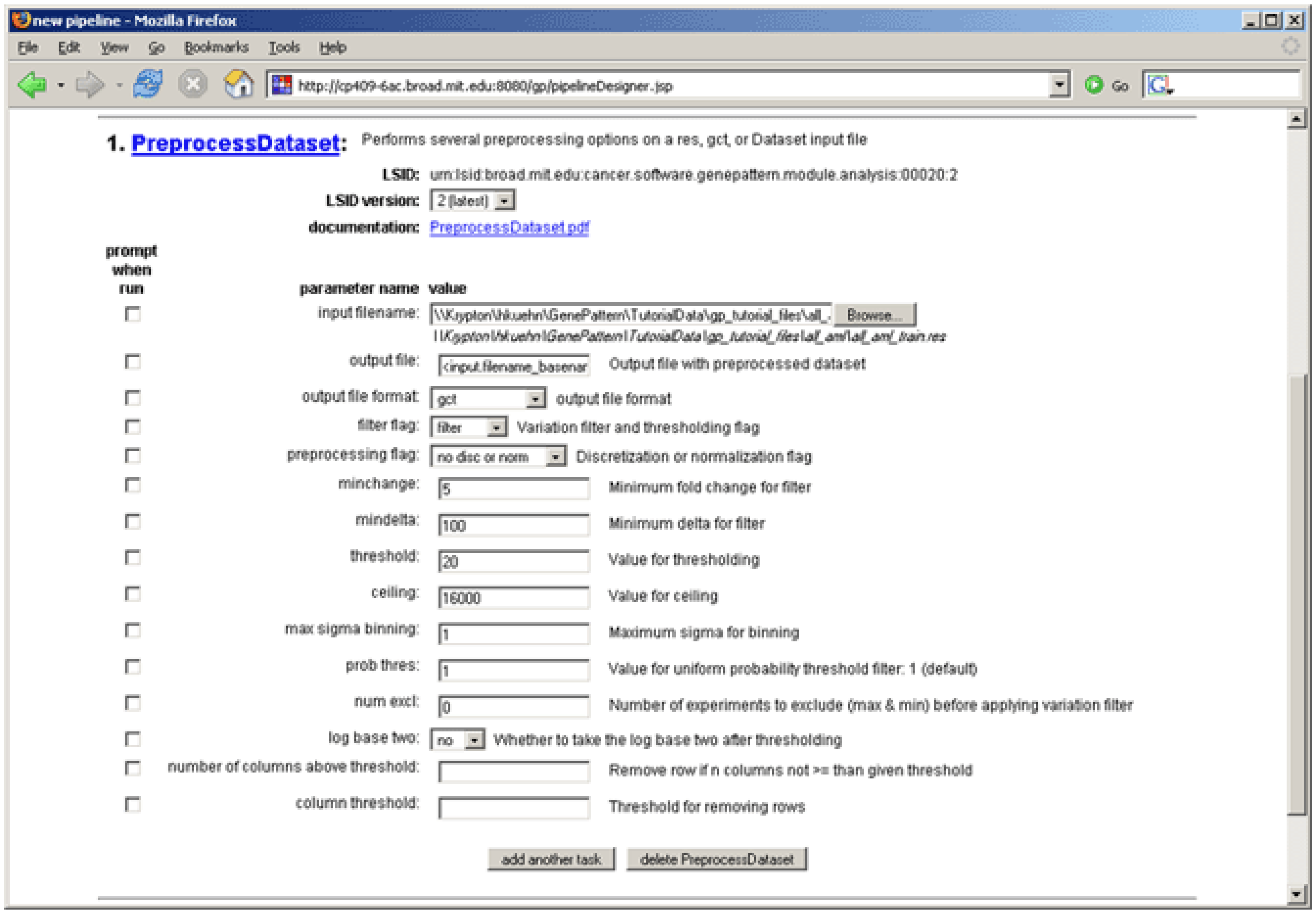
Pipeline name: my.all.aml (* required) new pipeline delete... clone...
Description: Pipeline for tutorial
Author: Gene Pattern User (name)
Owner: gpuser@broad.mit.edu (email address)
Privacy: private
Version comment: Initial draft
Documentation: Browse...
LSID:

task types
Annotation
Clustering
Gene List Selection
Image Creators
Missing Value Imputation
Pipeline
Prediction
Preprocess & Utilities
Projection
Proteomics
Sequence Analysis
Statistical Methods
Visualizer

Choose a Preprocess & Utilities task
Choose a Preprocess & Utilities task
ConvertLineEndings - Converts line endings to the host operating syste...
ConvertToMAGEML - Converts a gct, res, or odf dataset file to a MAGE-M...
DownloadURL - Downloads a file from a URL
ExpressionFileCreator - Creates a res or gct file from a set of Affyme...
ExtractColumnNames - Lists the sample descriptors from a .res file.
ExtractRowNames - Extracts the row names from a .res, .gct, or .odf fi...
GEOImporter - Imports data from the Gene Expression Omnibus (GEO)
Hu68kHu35kAtoU95 - Converts a list of Affymetrix Hu6800/Hu35KsubA prob...
LogTransform - Log transform a gct file
MapChipFeaturesGeneral - Change (map) the features (genes) of a datase...
mzXMLToCsv - Converts a mzXML file to a zip of csv files
PreprocessDataset - Performs several preprocessing options on a res, g...
ReorderByClass - Reorder the samples in an expression dataset and clas...
SplitDatasetTrainTest - Splits a dataset (and cls files) into a number...
TransposeDataset - Transpose a Dataset - .res .gct, .odf

You will see the interface for the **PreprocessDataset** module, as shown below, which allows you to do thresholding, fold change filtering, and omission of a number of highest or lowest-valued samples.

- Check the drop-down menu next to **LSID version**. This menu allows you to choose which of the installed **PreprocessDataset** modules to use. Ensure you have selected the latest version.
- For the input filename, click on **Browse**, and navigate to your datasets. If you need to download them again you may do so here. Go into the **aml_all** folder and choose **all_aml_train.res**.
- Change **output file format** to "gct".
- Set the following parameters:
  For the **filter flag** choose **filter**
  minchange = 5
- Let the rest of the parameters remain at the default. GenePattern will provide a filename for your results files. If you'd like to use a name other than the one provided you can enter it up in the **output file** box.

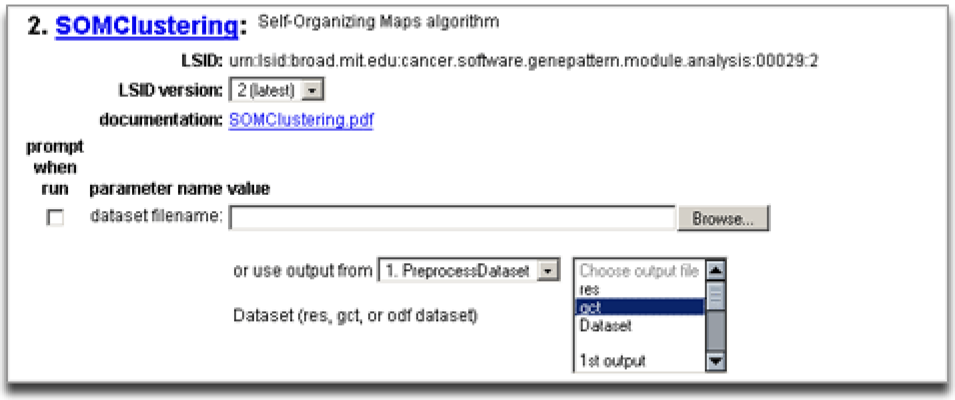

new pipeline - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://cp409-6ac.broad.mit.edu:8080/gp/pipelineDesigner.jsp
Go

1. PreprocessDataset: Performs several preprocessing options on a res, gct, or Dataset input file

LSID: urn:lsid:broad.mit.edu:cancer.software.genepattern.module.analysis:00020:2
LSID version: 2 (latest)
documentation: PreprocessDataset.pdf

prompt when run	parameter name	value	
☐	input filename:	\\Kypton\hkuehn\GenePattern\TutorialData\gp_tutorial_files\all_ [Browse...] \\Kypton\hkuehn\GenePattern\TutorialData\gp_tutorial_files\all_aml\all_aml_train.res	
☐	output file:	<input.filename_basenar	Output file with preprocessed dataset
☐	output file format:	gct	output file format
☐	filter flag:	filter	Variation filter and thresholding flag
☐	preprocessing flag:	no disc or norm	Discretization or normalization flag
☐	minchange:	5	Minimum fold change for filter
☐	mindelta:	100	Minimum delta for filter
☐	threshold:	20	Value for thresholding
☐	ceiling:	16000	Value for ceiling
☐	max sigma binning:	1	Maximum sigma for binning
☐	prob thres:	1	Value for uniform probability threshold filter: 1 (default)
☐	num excl:	0	Number of experiments to exclude (max & min) before applying variation filter
☐	log base two:	no	Whether to take the log base two after thresholding
☐	number of columns above threshold:		Remove row if n columns not >= than given threshold
☐	column threshold:		Threshold for removing rows

- Click **add another task**.
- Select **Clustering** drop-down, and then choose **SOM Clustering**.
- Leave the **LSID version** number as the latest version.
- The key to connecting tasks is in the input and output filenames of each module. Pipelines allow you to take the output from one module and use it as the input of another module. For the **input filename** parameter, next to **use output from**, select **1. Preprocess Dataset**, as shown below.



2. SOMClustering: Self-Organizing Maps algorithm

LSID: urn:lsid:broad.mit.edu:cancer.software.genepattern.module.analysis:00029:2
LSID version: 2 (latest)
documentation: SOMClustering.pdf

prompt when run	parameter name	value
☐	dataset filename:	[Browse...]

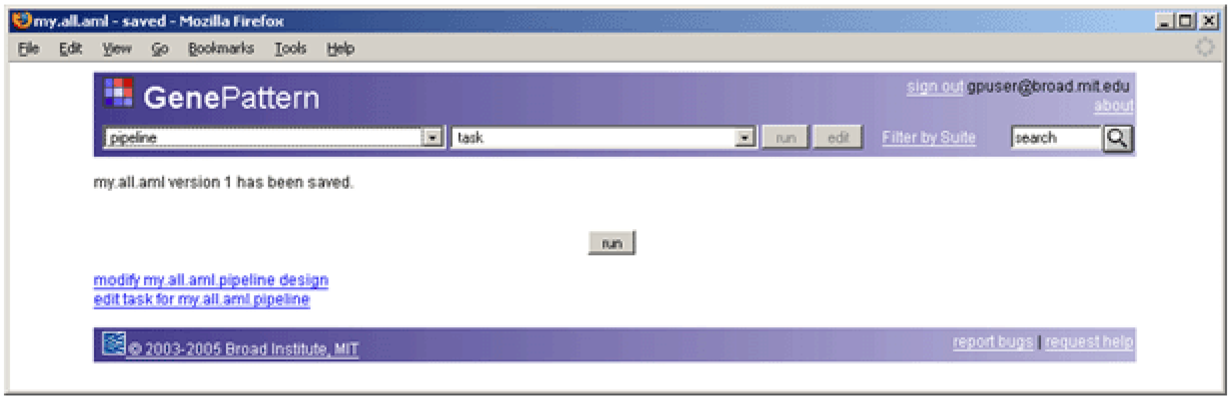
or use output from 1. PreprocessDataset

Dataset (res, gct, or odf dataset)

- Choose **gct** from the **Choose output file** drop-down menu. If a module creates more than one result file, the files are referred to in the order they are created. That order is listed in the documentation.
- For cluster range, enter **2**.
- Save the pipeline by clicking the **save** button at the bottom of the page.

To run this pipeline, do the following:

- Click the **run** button, shown below. The new pipeline may also be run from the **pipeline** dropdown on the header bar and in the **pipeline catalog** dropdown on the GenePattern home page.


my.all.aml - saved - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
GenePattern
sign out gpuser@broad.mit.edu
about
pipeline
task
run
edit
Filter by Suite
search

my.all.aml version 1 has been saved.

run
modify my.all.aml pipeline design
edit task for my.all.aml pipeline
© 2003-2005 Broad Institute, MIT
report bugs | request help

You will see the pipeline tasks in the window as they complete.

Exercise 6f: Editing a pipeline

Once you have created a pipeline you can modify it to include any changes you want to make in the workflow. For example, you might want to add additional stages or substitute one original dataset for another one.

In the following exercises, you will edit the pipeline you just created and modify it to run on a different data set and then also add stages to the pipeline.

- Navigate to the GenePattern home page.
- Under **`Pipelines`**, select the pipeline you just created, which will be called **`my.all.aml`** and will be color-coded in bright pink.
- The **`version`** will be `1`.
- Click the **`edit`** button.

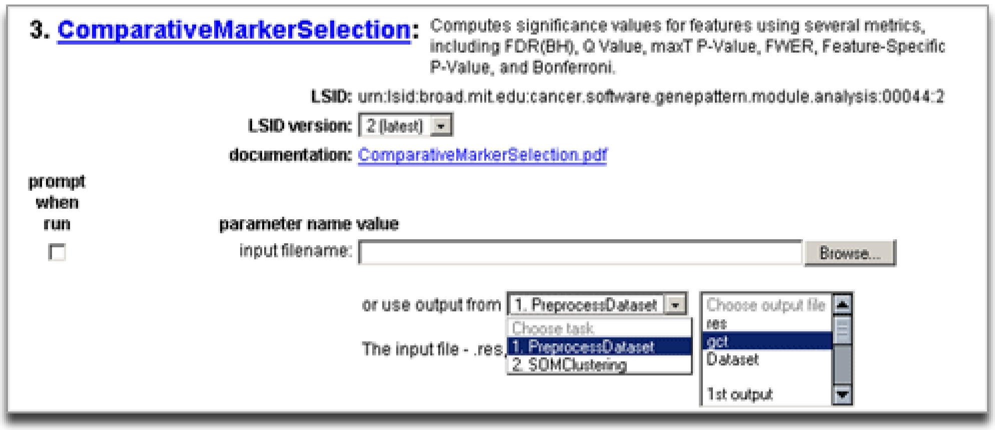
To run this pipeline on a different set of data, do the following:

- First modify the **`Version comment`** to reflect the changes you intend to make.
- Go to stage 1 and click the **`Browse`** button.
- Choose **`all_aml_test.res`** for the input file.
- Scroll to the bottom of the screen and click **`save`**.
- When you see the message that the pipeline has been saved, click the **`run`** button.

Although you saved the new pipeline in this example, you do not need to re-save a pipeline to run it on a new dataset. Once you select the new input file you can click **`run`**, and the pipeline will be run with the new data. However, the stored pipeline will still use the original data set unless you save it with the new one.

You will see your pipeline running. To add a new stage to the pipeline, do the following:

- Navigate to the GenePattern home page.
- Select your pipeline, **`my.all.aml`** and notice that this time the **`version`** is `2`.
- Click **`edit`**.
- Scroll to the bottom and click **`add another task`**.
- Select **`GeneListSelection->ComparativeMarkerSelection`**.
- The **`LSID version`** will be the latest version.
- The input to this analysis should be the preprocessed all_aml_test data, which is the result file from the preprocessing step. In the **`input filename`** parameter, next to **`use output from`**, select **`1. Preprocess Dataset`** from the drop-down menu. Select **`gct`** from the **`Choose output file`** drop-down menu.



3. ComparativeMarkerSelection: Computes significance values for features using several metrics, including FDR(BH), Q Value, maxT P-Value, FWER, Feature-Specific P-Value, and Bonferroni.

LSID: urn:lsid:broad.mit.edu:cancer.software.genepattern.module.analysis:00044:2

LSID version: 2 (latest)

documentation: ComparativeMarkerSelection.pdf

prompt when run **parameter name value**

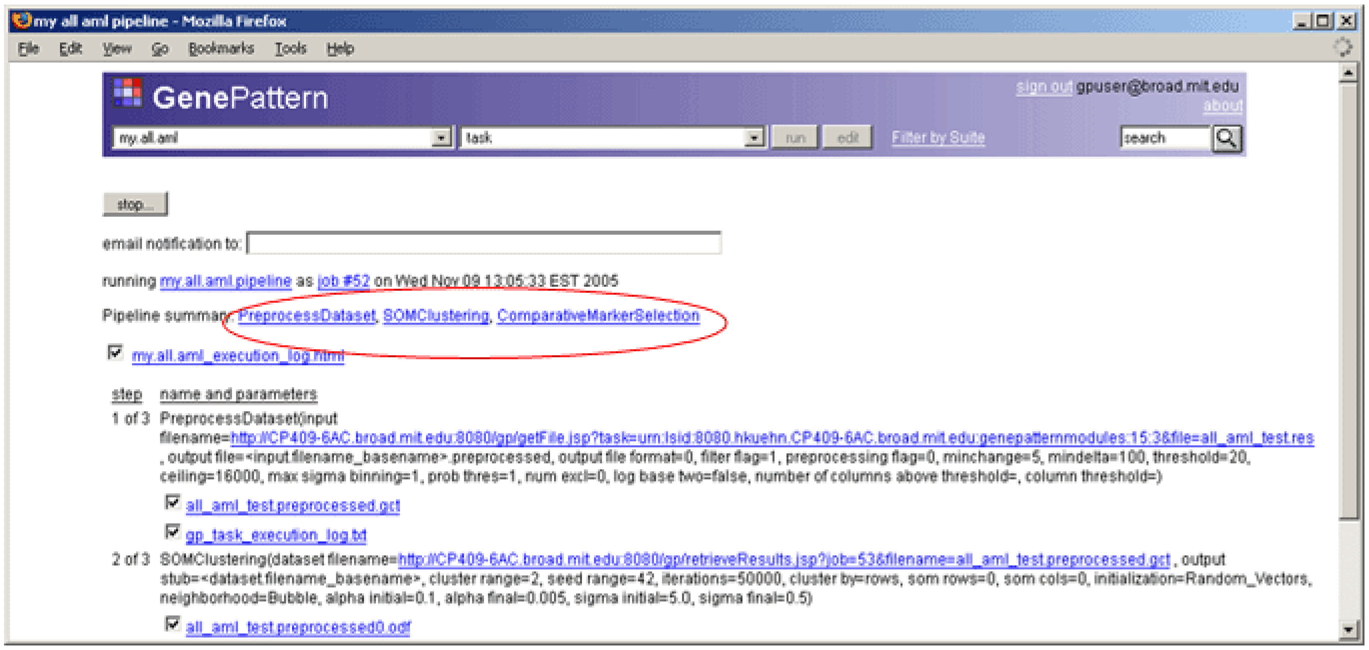
input filename:

or use output from 1. PreprocessDataset

The input file - .res

- For the **`class filename`** parameter, you will use the file that contains the class labels for the all_aml test dataset.
    - Click on **`Browse`**
    - Select **`all_aml_test.cls`**.
- Leave the other parameters at their default values.
- Click the **`save`** button.
- When you see the message that the pipeline has been saved, click the **`run`** button.

You will see the steps in your pipeline listed as they are finished, just as before, with the ComparativeMarkerSelection task at the end.



my all aml pipeline - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
GenePattern
sign out gpuser@broad.mit.edu
about
my.all.aml
task
run
edit
Filter by Suite
search
stop...

email notification to:

running my.all.aml.pipeline as job #52 on Wed Nov 09 13:05:33 EST 2005

Pipeline summary: PreprocessDataset, SOMClustering, ComparativeMarkerSelection

my.all.aml_execution_log.html

step name and parameters

1 of 3 PreprocessDataset(input filename=http://CP409-6AC.broad.mit.edu:8080/gp/getFile.jsp?task=urn:lsid:8080.hkuehn.CP409-6AC.broad.mit.edu:genepatternmodules:15:3&file=all_aml_test.res , output file=<input.filename_basename>.preprocessed, output file format=0, filter flag=1, preprocessing flag=0, minchange=5, mindelta=100, threshold=20, ceiling=16000, max sigma binning=1, prob thres=1, num excl=0, log base two=false, number of columns above threshold=, column threshold=)

all_aml_test.preprocessed.gct

gp_task_execution_log.txt

2 of 3 SOMClustering(dataset filename=http://CP409-6AC.broad.mit.edu:8080/gp/retrieveResults.jsp?job=53&filename=all_aml_test.preprocessed.gct , output stub=<dataset.filename_basename>, cluster range=2, seed range=42, iterations=50000, cluster by=rows, som rows=0, som cols=0, initialization=Random_Vectors, neighborhood=Bubble, alpha initial=0.1, alpha final=0.005, sigma initial=5.0, sigma final=0.5)

all_aml_test.preprocessed0.odf

NOTES

If you're working on a pipeline and you make a mistake in the current stage, you can click on the **`delete`** button.

You can insert or delete steps anywhere in a pipeline - not only at the end.

There is a task type called **`suggested`**. This leads to a list of all the modules which GenePattern knows about that will take your output file format.

prompt when run ☐

There is a check box next to each pipeline parameter with the title **`prompt when run`**. If you want to fill in a parameter when you run the pipeline, instead of when you design it, click **`prompt when run`** next to that parameter.

Managing Jobs in GenePattern

GenePattern provides comprehensive functionality to allow you to view and manage the jobs that are running on your GenePattern server. This includes the ability to view the status of jobs, download and delete job results, stop a job in progress,and receive job results while not logged into the GenePattern server.

In the following text, a **job** is defined as **any GenePattern module running on the server**. A job is therefore a running task or pipeline. Visualizers run on the client and not on the server, so they are not considered server jobs.

Job Ownership

Each job that runs on a GenePattern server has an owner. The owner is identified by the login id. For example, if you logged in as john_doe@abc.com, all jobs that you run on the server will be owned by john_doe@abc.com. You can view jobs run by other users, but you can only delete job results from jobs that you own.

Job Persistence

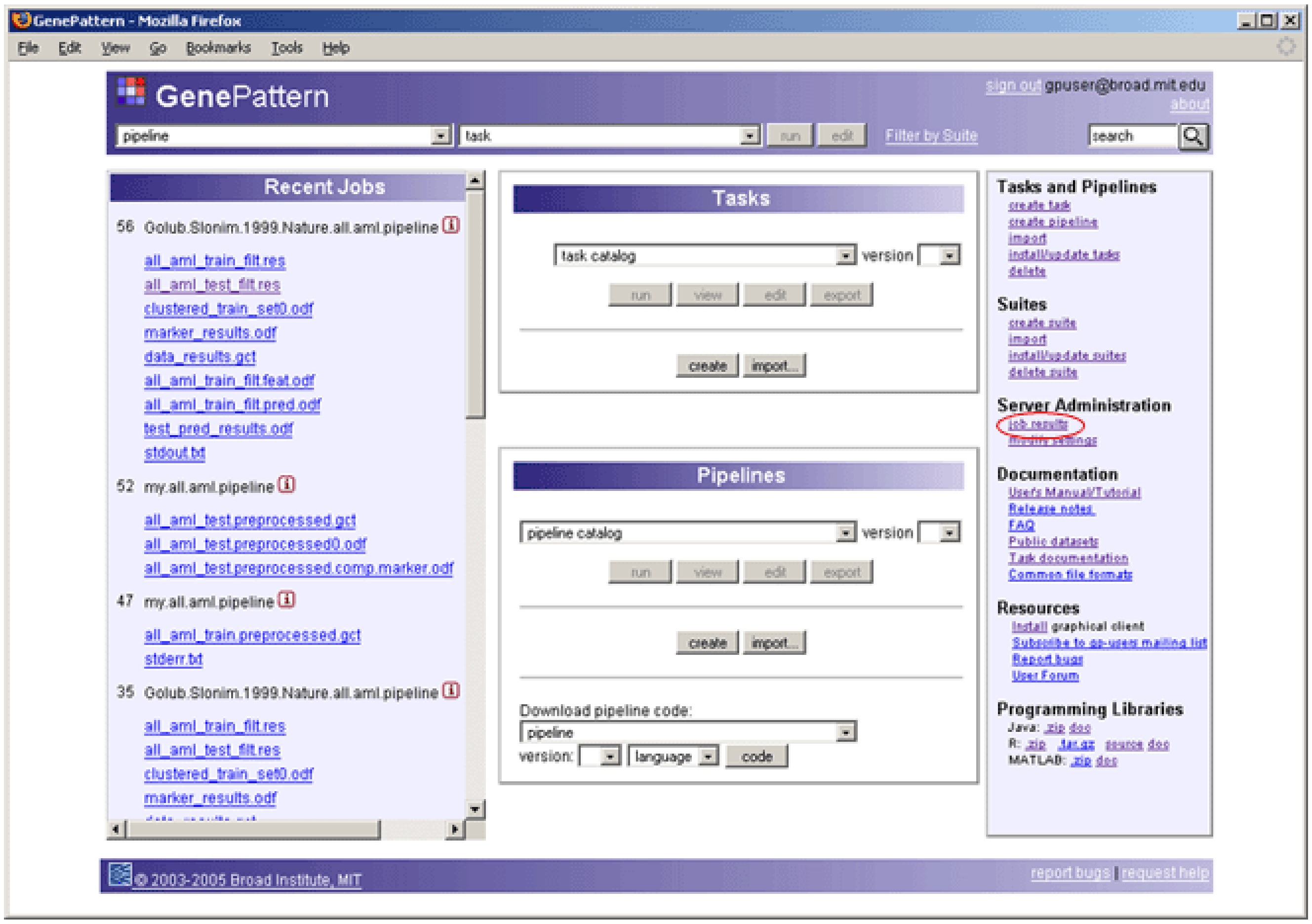
An important feature to note is that jobs have **persistence**. This means:

- **Jobs run independently of the client.** Your GenePattern client doesn't need to be running in order for a job to continue. If you launch a task or pipeline from any environment, that job will continue to run, even if you shut down the client that launched the job. The next time you enter your GenePattern client, you will see the status of any jobs that were running when it was closed.
- **The server restarts jobs after a crash.** If the GenePattern server crashes while executing a job, it will re-launch that job the next time it starts up.

The Jobs Window

To view the jobs on your server:

- Navigate to the GenePattern home page (http://localhost:8080/gp if it is running standalone). The Recent Jobs column lists the jobs you have run on the server and the result files from those jobs. Click a file link to view the file; click the information icon next to the job name to view the job's parameters.
- To download or delete result files, under Server Administration in the right column, click **`job results`**, as shown below:



Recent Jobs

56 Golub.Slonim.1999.Nature.all.aml.pipeline

all_aml_train_filt.res
all_aml_test_filt.res
clustered_train_set0.odf
marker_results.odf
data_results.gct
all_aml_train_filt.feat.odf
all_aml_train_filt.pred.odf
test_pred_results.odf
stdout.txt

52 my.all.aml.pipeline

all_aml_test.preprocessed.gct
all_aml_test.preprocessed0.odf
all_aml_test.preprocessed.comp.marker.odf

47 my.all.aml.pipeline

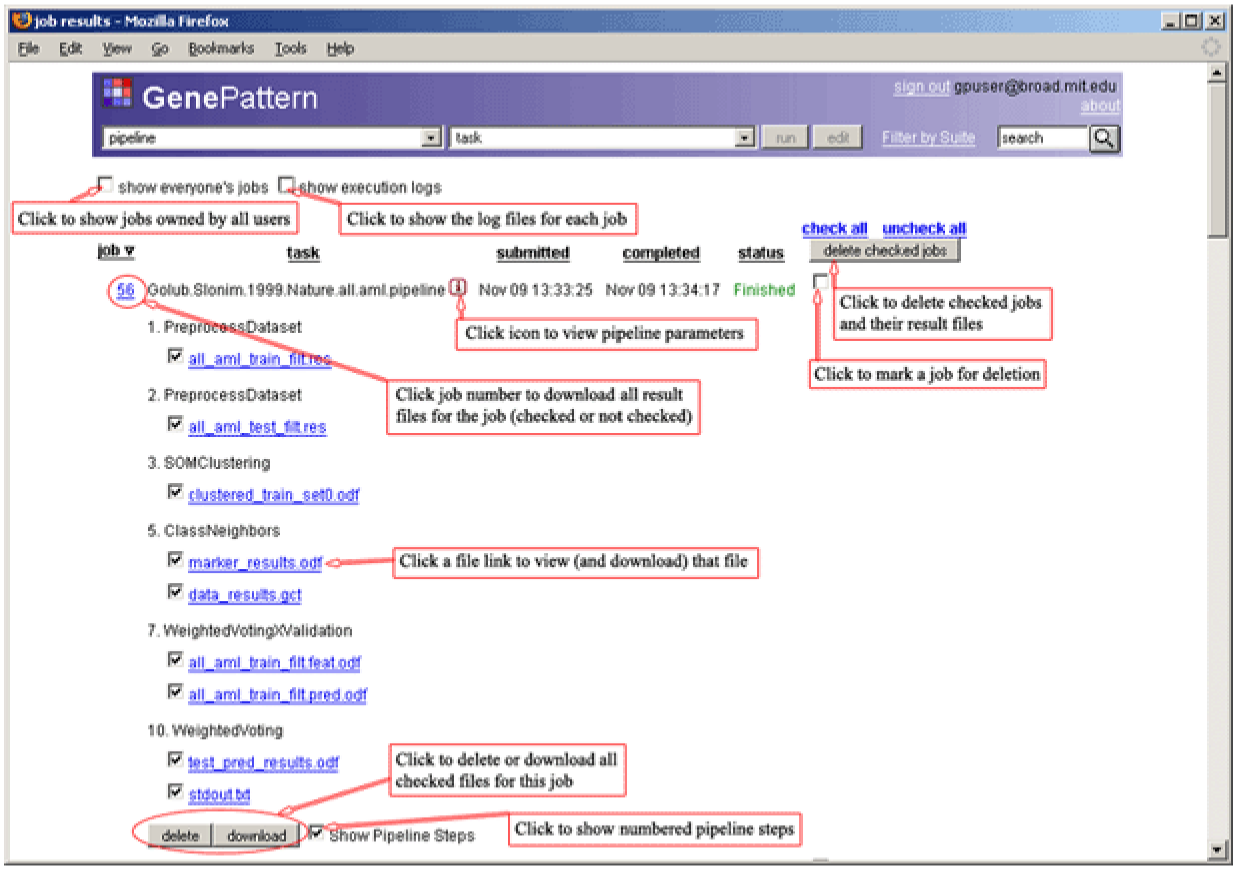
all_aml_train.preprocessed.gct
stderr.txt

35 Golub.Slonim.1999.Nature.all.aml.pipeline

all_aml_train_filt.res
all_aml_test_filt.res
clustered_train_set0.odf
marker_results.odf

GenePattern Admin Frame

You will see a more detailed list of the jobs you have run on the server, and all of the result files from those jobs:



56 Golub.Slonim.1999.Nature.all.aml.pipeline Nov 09 13:33:25 Nov 09 13:34:17 Finished

1. PreprocessDataset
all_aml_train_filt.res
2. PreprocessDataset
all_aml_test_filt.res
3. SOMClustering
clustered_train_set0.odf
5. ClassNeighbors
marker_results.odf
data_results.gct
7. WeightedVotingXValidation
all_aml_train_filt.feat.odf
all_aml_train_filt.pred.odf
10. WeightedVoting
test_pred_results.odf
stdout.txt

Click to show jobs owned by all users

Click to show the log files for each job

Click to delete checked jobs and their result files

Click icon to view pipeline parameters

Click to mark a job for deletion

Click job number to download all result files for the job (checked or not checked)

Click a file link to view (and download) that file

Click to delete or download all checked files for this job

Click to show numbered pipeline steps

Jobs Page

This page allows you to perform the following functions on jobs:

- **Show everyone's jobs.** Click this checkbox to see all jobs that have been run on the server. You will see the owner's name next to each result file.
- **View a job.** Click the information icon to the right of the job name to view its parameters. By default, you see only the result files for each job; to see the execution logs as well, click `show execution logs` at the top of the window. NOTE: If a job is a pipeline, click `Show Pipeline Steps` at the end of the job to view the numbered pipeline tasks.
- **Download job results.** To download individual result files for a job, check the boxes next to those files and then click the `Download` button for that job. To download all files for a job, click the link for its job id (located in the `job` column). In either case, the files are combined in a zip file whose name is the id of that job, and you are prompted to save them. NOTE: If a job is a pipeline, you can download all result files from all steps in the pipeline.
- **Delete job results.** To delete individual files from a job, check the boxes next to those files and then click the `delete` button for that job. To delete an entire job, check the box to the right of the job name and click `delete checked jobs` at the top of the window.

Server Administration

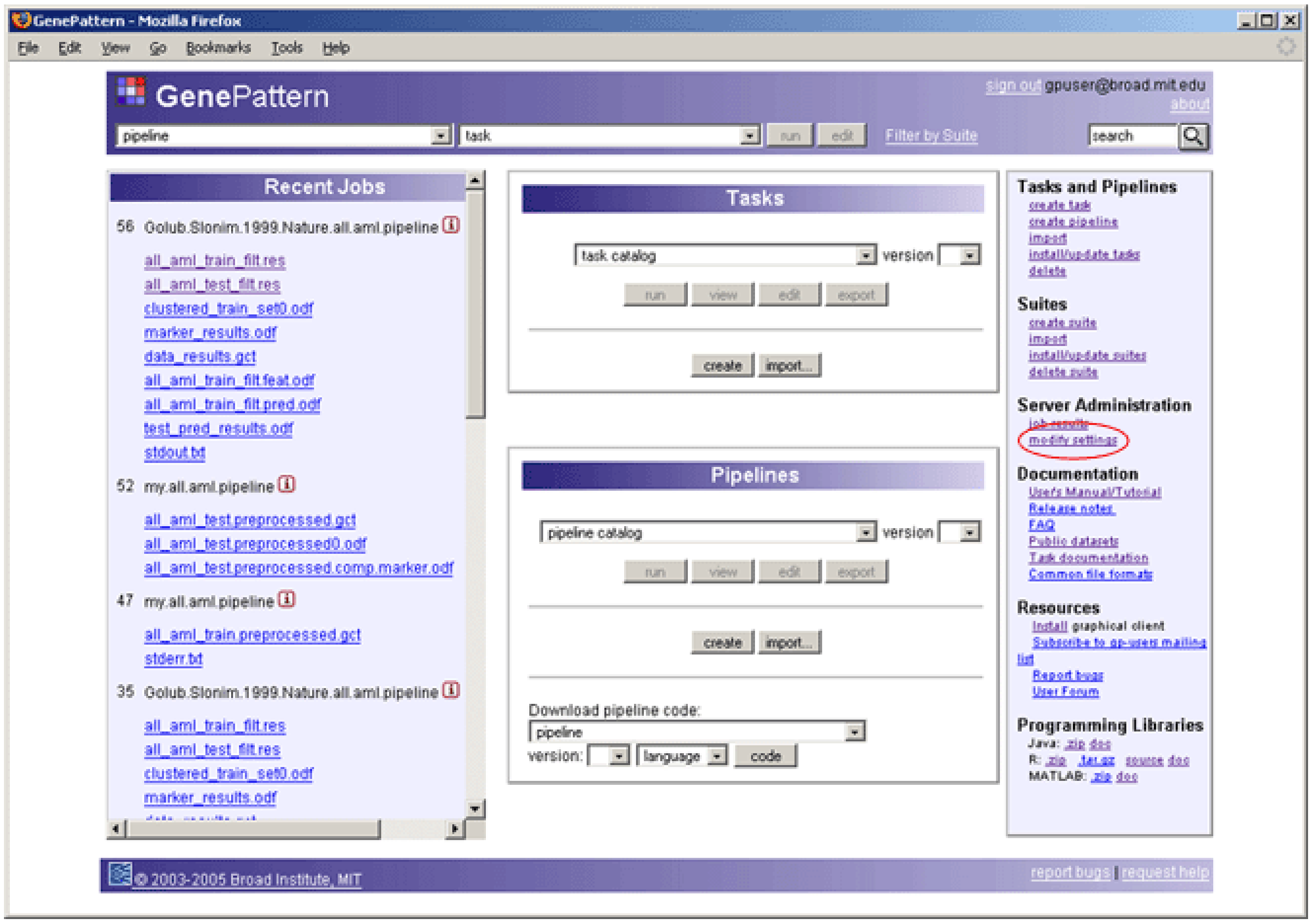
You use the GenePattern server administration options to configure the GenePattern server for your site. The server configuration options that most commonly need to be changed are available from the Server Administration page of the GenePatter Web Client. Additional configurations options, which rarely need to be modified, are stored in the `GenePatternServer/resources/genepattern.properties` configuration file.

Setting Common Configuration Options

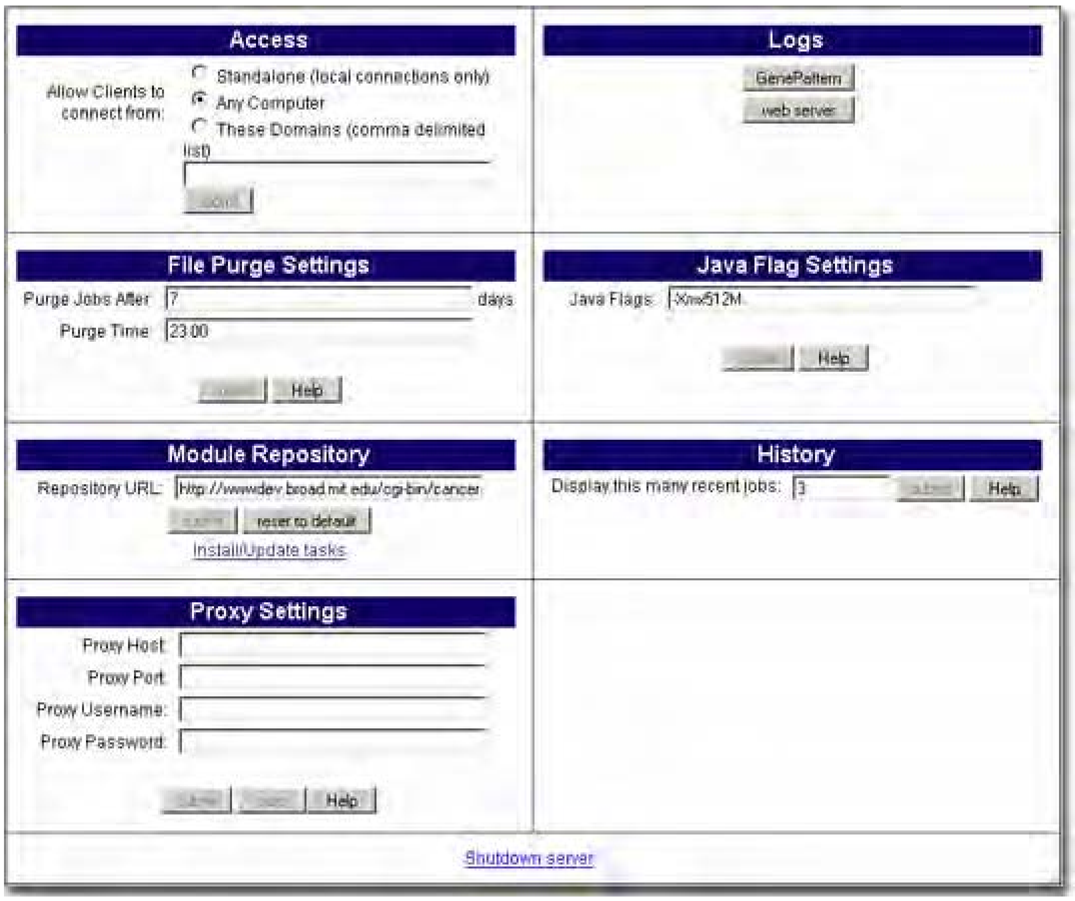
You can customize many aspects of the GenePattern server using the Server Administration page. To view this page:

- Navigate to the GenePattern home page (http://localhost:8080/gp if it is running standalone).

- Under Server Administration in the right column, click **`modify settings`**, as shown below:



You will see the Server Administration Page:



Access

Allow Clients to connect from: ( ) Standalone (local connections only) [✓ Any Computer] ( ) These Domains (comma delimited list)

Logs

File Purge Settings

Purge Jobs After: 7 days

Purge Time: 23:00

Java Flag Settings

Java Flags: -Xmx512M

Module Repository

Repository URL: http://wwwdev.broad.mit.edu/cgi-bin/cancer

History

Display this many recent jobs: 3

Proxy Settings

Proxy Host:

Proxy Port:

Proxy Username:

Proxy Password:

GenePattern Server Administration Page

The Server Administration page contains the following functionality:

- **Access.** To run the GenePattern server completely standalone, which will prevent any other clients from connecting to the server, click **`Standalone`**. To allow unlimited access, click **`Any Computer`**. To restrict access to a series of given domains, click **`These Domains`** and enter a comma-separated list, for example **`broad.mit.edu,dfci.harvard.edu,mit.edu`**. For more information about securing access to your server, see Securing Your GenePattern Server.
- **File Purge Settings** These settings allow you to specify when files will be purged from the server. The **`Purge Jobs After`** setting is the number of days that result files will stay on the GenePattern server before being purged. To learn how to save files locally so they are not purged, see Saving Result Files Locally. If you do not want your files to be purged, set this value to -1. The **`Purge Time`** setting specifies what time of day (24-hour format) the files will be purged.
- **Module Repository.** You can change the location of the module repository that GenePattern queries to determine which modules are new and updated. This functionality is useful if you have an internal module repository whose modules you would like to host. To implement a module repository, please contact the GenePattern development team.
- **HTTP Proxy Settings.** If you are behind a firewall, fill out this information to allow the GenePattern server to download and install modules from the module repository. If you don't know the necessary information, contact your systems administrator.
- **Logs.** To view warnings and messages from the GenePattern server, click **`GenePattern`**. To view messages that have been generated by the Web server that GenePattern uses, click **`web server`**.
- **Java Flag Settings.** Values you enter on this line are passed on the command line to any Java-based analysis modules. This feature allows you to, for instance, specify the amount of memory to allocate when launching Java-based analysis modules.
- **History.** This parameter allows you to specify the number of recent analyses that will be displayed in the top portion of the **`pipeline`** and **`tasks`** drop-down menus at the top of the GenePattern home page.
- **Shut down server.** Click this link to shut down the GenePattern server.

Executing Jobs on a Cluster

Queuing systems such as Load Sharing Facility (LSF) and Sun Grid Engine (SGE) allow computational resources to be used effectively. GenePattern can be seamlessly integrated with either of these programs. To configure your GenePattern server to execute jobs using LSF or SGE, add the `commandPrefixK` property to the `GenePatternServer/resources/genepattern.properties` configuration file. The value of this property is prepended to the command line for every task that is run on the server.

For example, to configure GenePattern to run jobs using LSF, set the commandPrefix as follows:

```
commandPrefix=bsub -K -o lsf_log.txt
```

- The –K flag instructs the bsub command to wait for the job to complete before returning.
- The –o flag specifies the file to which the job will write standard output and standard error messages.

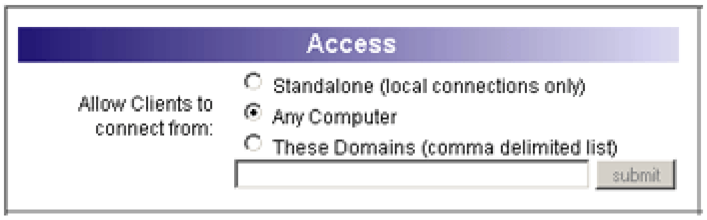
When using LSF, you may also want to set the environment variables BSUB_QUIET and BSUB_QUIET2 to prevent job messages from printing to standard out. Setting the BSUB_QUIET environment variable prevents bsub from printing the messages <<Job is submitted to default queue <normal>>> and <<Waiting for dispatch>>. Setting the BSUB_QUIET2 environment variable tells bsub not to print <<Job is finished>> to standard out.

Securing Your GenePattern Server

Securing a GenePattern server can be done at several different levels to help you control who is allowed to do what on your server. Since GenePattern is in effect primarily a web application (including SOAP interfaces) running in a web server, general approaches used for securing web servers are applicable here. GenePattern also includes some additional security features to allow non-technical users to easily execute some control over access to their server.

Access Filtering

The simplest approach to protecting your GenePattern server is to turn on connection filtering. The GenePattern web server has a servlet filter that can prevent users from accessing your GenePattern instance unless they come from a known computer. Setting connection filtering is done from the Access section of the GenePattern Server Administration page.


Access

Allow Clients to connect from: Standalone (local connections only) [✓ Any Computer] These Domains (comma delimited list)

The available options are:

- Standalone. Do not let any other compute connect. You can only access this GenePattern server from the computer that it is running on.
- Any Computer (default). Allow any other computer to connect to this Genepattern server without restriction.
- These Domains. You may enter a comma delimited list of the domains or specific IP addresses that are allowed to connect to this GenePattern server. GenePattern will scan all incoming connection attempts to see if they match in whole or in part any domain name or IP address in this list. If they match, access is allowed. If not the connection is redirected to a page indicating that this GenePattern server will not allow them to connect.

For more information about the GenePattern administration page, see GenePattern Server Administration.

Authentication

Default authentication into a GenePattern server does not require a password to authenticate that a user is who they claim to be. In version 2, GenePattern does not include an option to allow stronger authentication through a configuration setting, but it can be easily added by any capable java programmer at your site if needed. The remainder of this section is intended to be read by Java Programmers.

GenePattern user authentication is performed by a servlet filter installed in front of the GenePattern web application in its `web.xml` file. To implement stronger authentication you simply need to implement a new servlet filter that performs whatever authentication you desire (e.g. validating login against a database) along with a new login page to get collect the username and password from the user. The new servlet filter must, after authenticating a user, set a session attribute in the `java.servlet.http.HttpSession` object called `userID`. All pages in the GenePattern server and all database interactions will use this as the user's name.

To see the source code for the default `AuthenticationFilter.java` and the `web.xml` files, click the following links: AuthenticationFilter.java, web.xml.

Once you have written and compiled the new ServletFilter:

- Place the jar file containing the new ServletFilter into

  `*/GenePatternServer/Tomcat/webapps/gp/WEB-INF/lib`

- Modify the GenePattern server's `web.xml` document.
- Change the definition of the AuthenticationFilter to use the class that you have provided.
- Add any necessary configuration elements that it requires.
- Restart the GenePattern server for the changes to take effect.

Notes:

- It is important to maintain the existing order of the servlet filters in the `web.xml` document as they are used in the order they are defined in the document. The Authentication filter must come before the Authorization filter for the Authorization filter to work.
- If you look at the code for the default Authentication Filter in the appendix, you will see that it allows requests through that have a parameter called `jsp_precompile` that have come from the localhost. If you do not allow these requests through unauthenticated, you will see a series of errors when you start the GenePattern server as it attempts to precompile the JSP pages. These are **not** fatal errors, but will slow down server response for users the first time that pages are accessed following a server restart.

User Permissions

As of GenePattern 2.0, there is an integrated authorization mechanism built into every GenePattern server. A GenePattern administrator may select which users and groups are allowed to access the various pages and SOAP calls on the server. By default at install time, all users are afforded all authorizations on a server.

Authorizations are based on user identity as provided at GenePattern login. For information on how to control user authentication at login, see Authentication.

Once a user has been authenticated, the next step in securing the server is to assign each user specific rights, or permissions, to allow them to perform different actions on the GenePattern server. The default permissions loaded into GenePattern are as follows:

- **createTask** allows a user to create new analysis tasks on the GenePattern server via
  1. the Task Integrator form
  2. by loading from the task catalog
  3. by installing a zip file containing a task
  4. by installing a pipeline that includes tasks
  5. by installing a suite that includes tasks

  This is a **critical permission** to control: users with createTask permission can upload potentially any executable to be run on the server including worms, viruses and other malware.

- **createPipeline** allows a user to create new pipelines via
  1. using the pipeline designer form
  2. by loading from a zip file containing a pipeline (note: loading pipelines that include task zips will not load the associated tasks unless the user also has the createTask permission)
- **createSuite** allows a user to create new suites via
  1. the create suite form
  2. loading a zip file containing a suite. (note: loading suites that include task zips will not load the associated tasks unless the user also has the createTask permission)
- **administrateServer** allows the user to change server settings on the administrateServer page.

This is a **critial permission** to control: users with administrateServer permissions have the ability to modify connection settings thus preventing other users from logging into and using the server.

- **deleteTask** allows the user to delete tasks from this GenePattern server
- **deleteJob** allows the user to delete jobs from this GenePattern server

Control over permissions is accomplished through editing three configuration files in the GenePatternServer/resources directory: `userGroups.xml`, `permissionMap.xml`, and `actionPermissionMap.xml`. It is recommended that file permissions are applied to these files to ensure they are not modified unintentionally.

userGroups.xml

To see the default `userGroups.xml` file, click here: userGroups.xml.

The `userGroups.xml` file is used to define groups of users to whom permissions may be assigned. Any number of groups may be created as long as their names are unique. If the user named `*` is included in a group, as shown in the default settings, then any and all users are considered to pe a part of that group. A user may belong to any number of groups.

permissionMap.xml

To see the default `permissionMap.xml` file, click here: permissionMap.xml.

The `permissionMap.xml` file is used to map permission names to user groups. Essentially this is where you state that members of a particular group have a particular permission.

As with `userGroups.xml`, the presence of a group named `*` means that any and all groups have that permission and thus any and all users have that permission.

Any number of new permissions may be created as long as their names remain unique. Any number of groups may be assigned a permission by adding additional `<group name="thename"/>` elements to a permission.

To create a permission that no one is allowed to have, simply do not include any group elements in it (especially not `*`). For example:

```
<permission name="NoOneHasThisPermission"></permission>
```

actionPermissionMap.xml

To see the default `actionPermissionMap.xml` file, click here: actionPermissionMap.xml.

The actionPermissionMap.xml is the third and final configuration file for the authorization system. It is used to define what permissions apply to what server actions. In this context, server actions correspond to:

- pages in the web application
- methods in the SOAP interfaces

Essentially this file is simply linking jsp pages and SOAP methods with their required permission names.

It should be noted that many of the existing GenePatternServer web pages have been written so that if a user does not have permission to execute a link, the link is not displayed. This logic is implemented in the jsp pages, not in the configuration files. Should you choose to limit permissions to other additional pages, users may still see the links but will be redirected to an error page when they attempt to execute them.

Secure Sockets Layer (SSL) Support

The GenePattern web application is capable of running in a web server that is configured to use the HTTPS protocol, where essentially the regular http requests are routed through secure sockets layer, making them much harder for hackers to snoop and eavesdrop upon. If you have installed your GenePattern server into a web server other than the default Tomcat instance it is distributed with, configure your web server according to its instructions and then follow the final step of this description. When running under SSL, programming language clients and the GenePattern Graphical Environment may not be able to connect to your GenePattern server.

Step 1. Configure Tomcat for SSL support

Follow the instructions available at http://tomcat.apache.org/tomcat-4.1-doc/ssl-howto.html to configure the Tomcat instance for using SSL. You will need to modify the Tomcat config file (located in `GenePatternServer/Tomcat/conf`) when following these instructions.

Step 2.Configure GenePattern for SSL

Once the Tomcat (or other web server) has been configured for SSL, you need to modify the GenePattern configuration files to ensure that they are in synch with the web server.

Edit the file, `GenePatternServer/resources/genepattern.properties`:

1. Add a new key, `java.net.ssl.trustStore=<path to keystore>`.
   This needs to point to the keystore you created when configuring Tomcat (above) or some keystore that Genepattern can use to establish SSL connections.
2. Modify the value for the key `GENEPATTERN_PORT` to use the https port you selected (step 1 above).
3. Modify the value for the key `GenePatternURL` to use the https protocol and the https port you selected, for example:
   `http://localhost:8080/gp becomes https://localhost:8443/gp`

Save the file and restart your server. Any bookmarked links to your GenePattern server will now need to be updated to the new protocol and port.

GenePattern Programming Language Environment

If you are familiar with programming languages and have some understanding of how to use GenePattern tasks, you'll find that using GenePattern from a programming language is straightforward. Accessing GenePattern tasks programmatically enables you to analyze and visualize data while leveraging the power and flexibility of a programming language.

Using GenePattern from Java
Using GenePattern from MATLAB
Using GenePattern from R

Using GenePattern from Java

Using Java as a GenePattern client enables you to easily run GenePattern tasks and visualizers from within a Java application. Using the GenePattern Java library, you can run GenePattern analyses as easily as calling a routine.

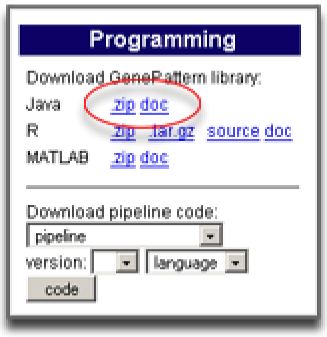
Getting Started in Java

If you aren't already familiar with Java, the best place to get started is at Sun Microsystems' website, http://java.sun.com. There you'll find downloadable programs, samples, tutorials, book suggestions, etc.

The GenePattern Java Library

A zip file for accessing GenePattern from Java is already installed on your GenePattern server, with links available to download to your client from a web browser. The Java library allows you to invoke GenePattern server tasks as if they were local Java methods running on your client, and to get back from the task a list of filenames of result files. The zip file also contains the Javadoc describing the API for accessing the server and running tasks.

If you don't already have the GenePattern library on your computer, point your browser at the GenePattern home page on your server and click on the download link for the GenePattern Java library in the Programming section, as shown here. The downloaded library should be unzipped into the directory where you'll be doing your Java development.



Running a program
Let's examine a simple Java application that preprocesses a dataset and visualizes it using the HeatMapViewer. Below is some sample code, with comments preceding each line explaining its purpose.

You have to initialize various settings once in every application that accesses GenePattern. You'll need to customize the *italicized* server URL settings with those for your GenePattern server and your GenePattern userID (your email address). When you invoke the runAnalysis method, the GenePattern library will invoke the appropriate task on the server, passing all of the input parameters and input files. When the task is complete, the JobResult object can be queried for an array of filenames that are the output from the task. You can download the result files or you can leave them on the server and refer to them by URL, which is especially useful when they are intermediate results that are input to a subsequent processing stage.

The following working examples may be copied and pasted into your Java program so that you can try them out, modify them, and create your own useful solutions.

```
import org.genepattern.data.expr.ExpressionData;
import org.genepattern.client.GPServer;
import org.genepattern.webservice.JobResult;
import org.genepattern.webservice.Parameter;
import org.genepattern.io.IOUtil;
import java.io.File;

public class MyProgram {
   public static void main(String[] args)
                    throws Exception {
      GPServer gpServer=new GPServer("http://localhost:8080",
                                     "your email address");
```

Let's preprocess a dataset. This could be from a URL or a filesystem. For this example, we'll use a publicly-accessible URL, but a filename would be equally valid:

```
      String inputDataset=

"ftp://ftp.broad.mit.edu/pub/genepattern/all_aml/all_aml_train.res";
      JobResult preprocess=gpServer.runAnalysis("PreprocessDataset",
                                                new Parameter[] {
         new Parameter("input.filename", inputDataset)
      });
```

The `JobResult` object named `preprocess` now has a list of filenames (of length 1, in this case). Let's display the results in a heat map:

```
      // view results in a HeatMapViewer visualizer
      gpServer.runVisualizer("HeatMapViewer",
                             new Parameter[] {
         new Parameter("filename", preprocess.getURL(0).toString())
      });
```

Lastly, we'll download the preprocessed data and load it into a matrix for further analysis.

```
      String downloadDirName=String.valueOf(preprocess.getJobNumber());
      // download result files
      File[] outputFiles = preprocess.downloadFiles(downloadDirName);
      // load data into matrix for further manipulation
      ExpressionData expressionData=
         IOUtil.readExpressionData(outputFiles[0].getPath());
   }
}
```

The full source code is available here. From here, you can combine GenePattern analyses with any capabilities that the Java environment has to offer. Use Java's 2-D and 3-D graphics libraries to create graphic output. Summarize and report on the data using your own code. The important lesson here is that tasks create result files, and those files are available to the Java application for processing.

How do you figure out what the arguments that a task accepts? Point your browser at **`http://localhost:8080/gp/getTaskDoc.jsp`** (if you are not running GenePattern locally, substitute the url of your server for **`localhost:8080`**). For quick generation of a basic method for your tasks, try building a pipeline, then downloading the Java code for the pipeline from the Programming section of **`http://localhost:8080/gp/index.jsp`**

Using LSIDs from Java

As of version 1.3 of the GenePattern server, Life Science Identifiers (LSIDs) can be used instead of task names to identify tasks for GenePattern to run. An LSID may be submitted in place of the task name in the methods **`runAnalysis`** and **`runVisualizer`**. When an LSID is provided that does not include a version, the latest available version of the task identified by the LSID will be used. If a task name is supplied, the latest version of the task with the nearest authority is selected. The nearest authority is the first match in the sequence: local authority, Broad authority, other authority.

Using GenePattern from MATLAB

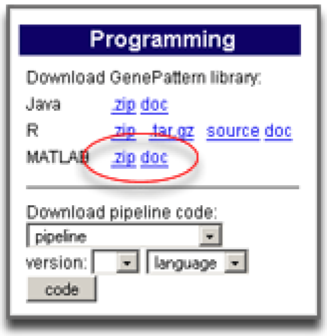
Using MATLAB as a GenePattern client enables you to easily run GenePattern tasks and to manipulate and visualize the results in a powerful, commercial technical computing application that works on most major platforms. Using GenePattern allows you to invoke methods written in many other languages without having to worry about how to launch them.

Getting Started in MATLAB

Resources and documentation are available at http://www.mathworks.com/.

The GenePattern MATLAB Library

A zip file for accessing GenePattern from MATLAB is already installed on your GenePattern server, with links available to download to your client from a web browser. The MATLAB library allows you to invoke GenePattern server tasks as if they were local MATLAB functions running on your client, and to get back from the task a list of filenames of result files. You may also pass files from one task to another while leaving them on the server.



If you don't already have the GenePattern library on your computer, point your browser at the GenePattern home page on your server and click on the download link for the GenePattern MATLAB library in the Programming section, as shown here. The downloaded library should be unzipped into your **`MATLAB7/toolboxes`** directory. If you don't have permission to put files there, you may place them in any other directory. Following the download and extraction of the files, you need to add the directories to your MATLAB path. At a MATLAB prompt, open the pathtool

```
>>pathtool
```

Use the MATLAB pathtool gui to add the GenePatternServer and GenePatternFileSupport directories with sub-folders to the MATLAB search path.

Example: Running a program
Let's examine a simple program that runs a task, displays the resulting output, and loads it into an MATLAB matrix for further analysis. Below is some sample code, with comments preceding each line explaining its purpose.

First is some initialization that's you'll have to do once in every application that accesses GenePattern. You'll need to customize the *italicized* userID settings with those for your GenePattern server and your GenePattern userID (your email address).

When you call the **`runAnalysis`** method on a GenePatternServer object in MATLAB, the GenePattern library will invoke the appropriate task on the server, passing all of the input parameters and input files. When the task is complete, it will return an MATLAB structure that contains a list of filenames that are the output from the task.

The following working examples may be copied and pasted into your MATLAB client so that you can try them out, modify them, and create your own useful solutions.

```
% Create a GenePattern server proxy instance

gp = GenePatternServer('http://localhost:8080','my.email@my.domain');
```

Let's transpose a dataset. You can refer to data by a URL or a local filename. For this example, we'll use a publicly-accessible URL:

```
% input dataset for transpose operation
params.output file name = 'transposed.out'
params.input filename='http://www.broad.mit.edu/mpr/publications/projects/Leukemia/
ALL vs AML train set 38 sorted.res'

% transpose the dataset
transposeResult = gp.TransposeDataset(params)


% alternate call to transpose the dataset
transposeResult = runAnalysis(gp, 'TransposeDataset', params)
```

`transposeResult` is a structure with a list of filenames (of length 1, in this case). Let's get the results. We'll display them in a file viewer window, and also load them into a matrix so that further manipulation can be performed:

```
% display the transposed results
edit 'transposed.out.odf'

% now read the output into a matrix
% so we can do further manipulation in MATLAB
myData = loadGenePatternExpressionFile('transposed.out.odf')
```

From here, you can combine GenePattern analyses with all of the rich functionality of MATLAB. For example, you can use MATLAB's plotting methods to create graphic output. Save modified matrices to files using **`save`**. Summarize and report on the data using your own code. The important lesson here is that tasks create result files, and those files are available to the MATLAB client for processing.

How do you figure out the task/method names available on your server? Run the listMethods function on your GenePatternServer object. There you'll find the list of the available Genepattern modules and visualizers as well as their descriptions. To learn the names of the the input parameters for a module, use the describeMethod function on your GenepatternServer object, passing it the module name.

```
% display the available GenePattern modules
listMethods(gp)

% now look at the parameters for the TransposeDataset module
describeMethod(gp, 'TransposeDataset')
```

An alternative method to get the parameters with their default values filled in is to use the **`getMethodParameters`** function of the GenePatternServer object. This returns a MATLAB structure with named elements for each parameter, filled in with the default value if one exists. After filling in the missing parameters and overriding defaults if desired, this structure can then be passed on to the **`runAnalysis`** method.

```
% display the available GenePattern modules
params2 = getMethodParameters(gp, 'TransposeDataset')

params2.input filename='http://www.broad.mit.edu/mpr/publications/projects/Leukemia/
ALL vs AML train set 38 sorted.res'

% transpose the dataset
transposeResult = gp.TransposeDataset(params2)
```

The GenePattern MATLAB library also has convenience methods to read and write RES, GCT, and ODF files. Even if you never look into the library, you will get plenty of mileage out of extending the techniques shown above on your own analyses.

Using LSIDs from MATLAB

As of GenePattern version 1.3, you can also use Life Science Identifiers (LSIDs) to identify a task when executing GenePattern code in MATLAB. An LSID may be submitted in place of the task name to **getMethodParameters** or **runAnalysis**. When providing an LSID to a method in addition to a task name, the LSID alone is used to determine what task to run. When an LSID is provided that does not include a version, the latest available version of the task identified by the LSID will be used.

```
% Example using LSIDs from MATLAB
params = getMethodParameters(gp,
'urn:lsid::broad.mit.edu:cancer.software.genepattern.
module.analysis:00026:0'); params.output_file_name = 'transposed.out'
params.input_filename='http://www.broad.mit.edu/mpr/publications/projects/Leukemia/
ALL_vs_AML_train_set_38_sorted.res'

% transpose the dataset
transposeResult = runAnalysis(gp,
'urn:lsid::broad.mit.edu:cancer.software.genepattern.
module.analysis:00026:0', params)
```

Using GenePattern from R

Using R as a GenePattern client enables you to easily run GenePattern tasks and to manipulate and visualize the results in a powerful, free statistical desktop package that works on most major platforms. Using GenePattern allows you to invoke methods written in many other languages without having to worry about how to launch them or whether you're passing incorrect parameters.

Getting Started in R

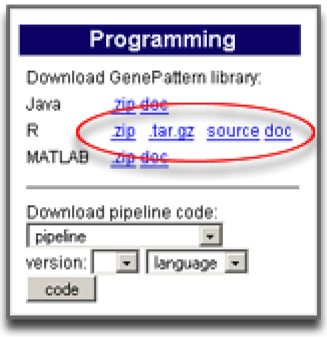
Resources (from www.r-project.org):

- An Introduction to R (PDF, approx. 100 pages, 650kB), based on the former "Notes on R", gives an introduction to the language and how to use R for doing statistical analysis and graphics.
- A draft of the R language definition (PDF, approx. 60 pages, 400kB) which document the language *per se*. That is, the objects that it works on, and the details of the expression evaluation process, which are useful to know when programming R functions.
- Writing R Extensions (PDF, approx. 85 pages, 500kB) covers how to create your own packages, write R help files, and the foreign language (C, C++, Fortran, ...) interfaces.
- R Data Import/Export (PDF, approx. 35 pages, 270kB) describes the import and export facilities available either in R itself or via packages which are available from CRAN.
- R Installation and Administration (PDF, approx. 30 pages, 200kB)
- The R Reference Index (PDF, approx. 2200 pages, 12MB) contains all help files of the R standard and recommended packages in printable form.

Accessing GenePattern from R

An R library for GenePattern is already installed on your GenePattern server, with links available to download to your client from a web browser. The R library allows you to invoke GenePattern server tasks as if they were local R methods running on your client, and to get back from the task a list of filenames of result files, all of which are transferred to your local client. You may also pass files from one task to another while leaving them on the server.

If you don't already have the GenePattern library on your computer, point your browser at the GenePattern home page on your server and click on the download link for the GenePattern R library in the Programming section, as shown here.



Both Windows (zip) and Unix (tar.gz) formats are available. The downloaded library should be untarred or unzipped into your `R/library` directory. If you don't have permission to put files there, the GenePattern release notes detail how you can create a private library in another location and use an environment variable to inform R.

Example: Running a program
Let's examine a simple program that runs a task, displays the resulting output, and loads it into an R matrix for further analysis. Below is some sample code, with comments preceding each line explaining its purpose.

First is some initialization that's you'll have to do once in every application that accesses GenePattern. You'll need to customize the *italicized* userID settings with those for your GenePattern server and your GenePattern userID (your email address).

`gpTasksAsFunctions` creates the proxies for each GenePattern task visible to the user on that server. That means that it creates method definitions, named for each task, with named input parameters. When you call the R method, the GenePattern library will invoke the appropriate task on the server, passing all of the input parameters and input files. When the task is complete, it will return an R list of filenames that are the output from the task.

The following working examples may be copied and pasted into your R client so that you can try them out, modify them, and create your own useful solutions.

```
# Load GenePattern library
library(GenePattern)

# The server will want to know whose tasks to provide
gpLogin("my.email.address@mydomain.com");

# Ask the server to create R methods that proxy
# each of the installed tasks.
gpTasksAsFunctions();
```

Let's transpose a dataset. You can refer to data by a URL or a local filename. For this example, we'll use a publicly-accessible URL:

```
# input dataset for transpose operation
input.ds <-
"http://www.broad.mit.edu/mpr/publications/projects/Leukemia/
ALL_vs_AML_train_set_38_sorted.res";

# transpose the dataset
transpose.out <- TransposeDataset(input.filename=input.ds,
output.file="transposed");
```

`transpose.out` now has a list of filenames (of length 1, in this case). Let's get the results. We'll display them in a file viewer window, and also load them into a matrix so that further manipulation can be performed:

```
# display the transposed results
file.show(transpose.out$transposed.odf);

# now read the output into a matrix
# so we can do further manipulation in R
data <- read.delim(transpose.out$transposed.odf, as.is=T, header=F,
sep="\t", skip=9, comment="";)
data <- as.matrix(data);
cols <- length(data[1,]);
rows <- length(data[,1]);
```

From here, you can combine GenePattern analyses with all of the rich statistical functionality of R. For example, you can use R's `plot` and `legend` methods to create graphic output. Output JPEGs of your visualized data using `savePlot`. Save modified matrices to files using `save`. Summarize and report on the data using your own code. The important lesson here is that tasks create result files, and those files are available to the R client for processing.

How do you figure out what the task/method names and arguments are supposed to be? Point your browser at `http://localhost:8080/gp/taskWrapperGenerator.jsp?name=TransposeDataset` (or whatever task name you are interested in). There you'll find generated R code documenting the input parameters, describing the function of the module, and a link to the online documentation. If you want to browse documentation for all tasks, just go to `http://localhost:8080/gp/getTaskDoc.jsp` (if you are not running GenePattern locally, substitute the url of your server for `localhost:8080`).

The GenePattern R library also has convenience methods to read and write RES, GCT, and CLS files, to enable running of multiple tasks in parallel, to run tasks with input from files that were output from previous tasks without moving them from the server, and other utilities. Even if you never look into the library, you will get plenty of mileage out of extending the techniques shown above on your own analyses.

Using LSIDs from R

As of version 1.3 of the GenePattern server, Life Science Identifiers (LSIDs) can be used instead of task names to identify tasks for GenePattern to run. An LSID may be submitted in place of the task name in the methods `runAnalysis` and `runVisualizer`. When an LSID is provided that does not include a version, the latest available version of the task identified by the LSID will be used. If a task name is supplied, the latest version of the task with the nearest authority is selected. The nearest authority is the first match in the sequence: local authority, Broad authority, other authority.

MATLAB Notes

You can use MATLAB with GenePattern in one of two ways:

- As a GenePattern client, which enables you to easily run GenePattern tasks and to manipulate and visualize the results in a powerful, commercial technical computing application that works on most major platforms. For more information about using MATLAB in this way, see Using GenePattern from MATLAB.
- As the source language for a GenePattern module, which allows you to turn a MATLAB-based analysis algorithm or methodology into a GenePattern module.

This section provides more information about using MATLAB as the source language for a GenePattern module:

- MATLAB Modules in GenePattern describes the options and steps necessary to turn a MATLAB-based analysis algorithm or methodology into a GenePattern module.
- MATLAB File Support describes a set of MATLAB functions, provided with GenePattern, that can be used to read and write expression data files in the standard GenePattern formats including res, gct and odf.

MATLAB Modules in GenePattern

This document describes the options and steps necessary to turn a MATLAB-based analysis algorithm or methodology into a GenePattern module. There are two paths for GenePattern enablement of a MATLAB analysis presented below with different pros and cons.

The first approach (referred to as the direct approach) is to directly call the MATLAB executable from GenePattern, running a MATLAB command directly from the command-line or via a startup.m file. This approach is best suited to use on a stand-alone GenePattern server for a user that already has a MATLAB license, and who will not redistribute the MATLAB-based GenePattern modules to other users who do not have their own MATLAB licenses.The advantages to this approach are that it is the simplest means to getting your m-code running on GenePattern, it can be used for any MATLAB and GenePattern supported platform, and it allows for easier modification of the m-code files as you modify your analysis. The disadvantage of this approach is that passing arguments to MATLAB differs on different computing platforms and it requires a MATLAB license for each concurrent user on the GenePattern server machine.

The second approach (referred to as the compiled approach) is to use the MATLAB Compiler 4.0 to generate a standalone executable which is then used as the GenePattern module. The advantages to this approach are that it allows redistribution of the module to other GenePattern users who do not have their own MATLAB license. It can also be run on a shared server without needing to get a MATLAB license for each concurrent user. Performance may be slightly better for fast running tasks since the startup delay will be shorter but the actual execution time will be approximately the same as for the other approach. The disadvantages are that it requires you to have a MATLAB Compiler license, it can be compiled only for Windows and Linux servers, it must be compiled separately for each platform, and it requires the MATLAB Component Runtime (MCR) to be installed on the GenePattern server machine before the compiled code can be run.

If you are simply using your m-code on your stand-alone GenePattern server, the direct approach is simpler, but if you want to deploy it on a shared server for others to use or to give copies to other people, the compiler approach is preferred.

MATLAB Version

The steps described below require the use of MATLAB Release 14, and the MATLAB Compiler 4.0 or later. Earlier versions of MATLAB used a different deprecated mechanism for deployment which is not compatible with the instructions provided in this document.

Limitations

Currently, the MATLAB Compiler 4.0 is only available on Windows and Linux. Therefore these are the only platforms that you may deploy a compiled MATLAB based task on.

The MATLAB Compiler 4.0 generates executables only for the platform on which it is executing. Therefore if you want to deploy a MATLAB based module onto a GenePattern server running a different OS (Windows and Linux only) then you must create the executable from MATLAB on your target platform. This means you need a MATLAB (and MATLAB Compiler) license for the platform on which you wish to deploy your executable.

Adapting your MATLAB code

Whether you are using the direct aproach or the compiled approach or the direct MATLAB call approach, there are two common patterns you will need to apply to your m-code to make it callable from the command-line.

Entry Function

Calling script m-code from a command line is possible, but may not be useful as you will not be able to pass arguments from GenePattern to the m-code script for it to act upon. In order to be able to pass arguments to your application such as the file paths to entry files, or parameteres for the algorithm, you should create a no-return entry function to serve as the top level call into MATLAB.

e.g.

```
    function analyseThis ( filename, whatToWrite )
    ... (your m-code here)
```

Passing Arguments

Arguments to the MATLAB executable will be passed in to MATLAB from the command line as Strings. Therefore the arguments that your entry function expects need to be Strings as well. One typical use case would be to pass arguments representing the input and output file names for your m-code to use. If numeric parameters values are to be passed, they must be converted from a String into a number in your entry function.

Compilation

If you do not plan to use the compiled m-code approach, continue on to the Distribution section .
Compiling your MATLAB m-code into a standalone executable is described in the MATLAB Compiler Documentation. Please refer to this documentation to understand all of the options available to you.

To summarize the simplest case, from within MATLAB at the MATLAB prompt, execute the following command;

```
   mcc -m analyseThis
```

Where example is the name of your entry function. This will generate several files in your $MATLAB_ROOT/work directory including;

Windows:
analyseThis.exe
analyseThis.ctf

Linux:
analyseThis
analyseThis.ctf

The complete set of generaetd files is described in the following table.

analyseThis *or* analyseThis.exe	Executable file (.exe on Windows)
analyseThis.ctf	Component Framework file
analyseThis.c	C language Source Code
analyseThis.h	C Language Header file
analyseThis_main.c	C language Source Code
analyseThis_mcc_component_data.c	C language Source Code

You will need to remember the location of the executable and ctf files for later use as they will need to be added to the GenePattern module you will create as support files.

Distribution

This section assumes that you are already familiar with deploying GenePattern modules written in other supported languages (Java, R, Perl). If you are unfamiliar with the process of deploying a Genepattern module, please refer to the GenePattern Tutorial section on the Task Integrator before proceeding.

Direct Approach Distribution

To deploy a m-code based MATLAB application as a GenePattern module, in addition to the usual steps for creating a new module, you will need to ensure MATLAB can be run from GenePattern.

Preparing the GenePattern Server

To distribute your entry function as a Module using the direct MATLAB approach, the first step is to assure that MATLAB is in your PATH system environment variable so that GenePattern will be able to call it from a command line.

If you do not want to put MATLAB on your path, you may alternately enter the full path to the MATLAB executable on the GenePattern Module command line. Note that this will make the module less portable to other GenePattern servers.

Windows

1. open a DOS window
2. type "matlab" and hit the 'Enter' key.

If the MATLAB application starts, it is already on your path. If not,

1. open the Start->Settings->Control Panel
2. Double click on System
3. Select the 'Advanced' tab.
4. Click the 'Environment Variables' button.
5. Select or create the PATH variable
   - add the "$MATLAB_ROOT/bin" directory to the path

Retry calling MATLAB from a new dos window.

Linux, Unix and Macintosh

1. Open a terminal window
2. type "matlab" and hit the 'Enter' key

If the MATLAB application starts, it is already on your path. If not, add it to your path appropriately for the unix shell you are using;

e.g. for csh, add this line to your .cshrc file
    export PATH=$PATH:$MATLAB_ROOT/bin

Writing the Module Command-Line

Windows:
On windows, you can get away with the simplest command line where you simply call MATLAB with the -r flag to execute your function and provide the function arguments within a double quoted string.

e.g.
```
    matlab -nosplash -r "analyseThis <p1> <p2>"
```

This will call MATLAB to run without the splash screen (-nosplash) and tell it to execute the string "analyseThis <p1> <p2>" where p1 and p2 are GenePattern module parameters that will be passed to the MATLABcommand line as Strings. The function analyseThis will be looked for on the MATLAB path and it is not even necessary to upload it as a support file although this is recommended.

Linux, Mac, Unix (and Windows):

On platforms other than windows, the execution of the command line differs slightly due to variations in the Java Virtual Machines (VMs) that GenePattern is running. Java VMs on unix platforms attempt to parse and quote the command line resulting in MATLAB generating errors in it 'eval' function if you attempt to use the simple example described above for Windows.

On these platforms it is necessary to use a wrapper Java class to launch MATLAB. This wrapper class will also work on windows and does not rely on the MATLAB PATH variable making it the preferred approach for all platforms using the direct approach to MATLAB execution.

Add runmatlab.jar as a module support file. This file is available by request. Email to gp-help@broad.mit.edu if you would like a copy. Alternately the java source code for the RunMatlab class is included in Appendix C.

Write your command line as follows;

```
    <java> -cp <libdir>runmatlab.jar RunMatlab analyseThis <p1> <p2>
```

Where 'analyseThis' is replaced with your MATLAB entry function name and <p1> <p2> are the arguments to the function. The RunMatlab class will ensure that the arguments are correctly written out and will call MATLAB with the -nosplash and -nodisplay arguments. Source code is avaliable for RunMatlab.java in the appendix.

Compiled Approach Distribution

To deploy a compiled MATLAB based application as a GenePattern module, in addition to the usual steps for creating a new module, you will need to create a launcher script to call the MATLAB Compiler generated application.

Preparing the GenePattern Server

For the GenePattern server to be able to run a MATLAB Compiler generated application, it must first have the MATLAB Component Runtime installed on it. This is a collection of shared libraries distributed by The Mathworks containing the runtime code for MATLAB that is used by the generated application. If the GenePattern server has already had MATLAB installed on it, you do not have to do anything as the libraries will have been installed with MATLAB itself.

Full details can be found in the MATLAB Compiler documentation. Specifically see the section titled "Deployng Components to Other Machines". To summarize this documentation, on the GenePattern server machine you need to run the MCRInstaller.

Windows:
1. Copy <matlabroot>\toolbox\compiler\deploy\win32\MRCInstaller.exe to the server machine.
2. run MCRInstaller.exe

Linux:
1. within MATLAB, at the MATLAB prompt execute the command "
        buildmcr
2. Copy <matlabroot>/toolbox/compiler/deploy/MCRInstaller.zip to the server machine.
3. On the server machine unzip MCRInstaller.zip into a directory (<mcr_root>).
4. update the dynamic library path for the user running the GenePattern server

```
setenv LD_LIBRARY_PATH

    <mcr_root>/runtime/glnx86:

    <mcr_root>/sys/os/glnx86:

    <mcr_root>/sys/java/jre/glnx86/jre1.4.2/lib/i386/client:

    <mcr_root>/sys/java/jre/glnx86/jre1.4.2/lib/i386:

    <mcr_root>/sys/opengl/lib/glnx86:${LD_LIBRARY_PATH}
```

Writing the Launcher Script

The module command line for a MATLAB compiler based module looks a little different than for a normal module. This is a result of the requirement for the exe that the compiler generates requiring the .ctf file that it also generates to be on the path. The easiest way to deal with this is to create a .bat or .sh file to launch the MATLAB application that adds this file to the PATH or LIBPATH and then runs the aaplication itself.

Windows:
For example, for an application called testTwo.exe on Windows, create a .bat called "mllaunch.bat" containing the following lines;

```
    set LIBDIR=%1
    set PATH=%LIBDIR%;%PATH%
    analyseThis %2 %3
```

Linux:
For example, for an application called testTwo on unix, create a .bat called "mllaunch.sh" containing the following lines;

```
    #!/bin/csh



    export MCR_ROOT=<path where you installed the files from MCRInstaller.zip>

    export LD_LIBRARY_PATH=$1:$MCR_ROOT/runtime/glnx86:$MCR_ROOT/sys/os/glnx86:\

        $MCR_ROOT/sys/java/jre/glnx86/jre1.4.2/lib/i386/client:\

        $MCR_ROOT/sys/java/jre/glnx86/jre1.4.2/lib/i386:\

        $MCR_ROOT/sys/opengl/lib/glnx86



    export PATH=$1:$PATH

    chmod a+x $1/analyseThis

    analyseThis $2 $3
```

For the linux operating system, we need to add the line to set the permissions on the executable file to executable since the default for the GenePatternServer is to not set this on uploaded files.

Writing the Module Command-Line

The command line would then call this file, passing the <libdir> parameter as the first argument (so that it can be added to the path).

Windows:
<libdir>mllaunch.bat <libdir><param1> <param2>
Linux:
sh <libdir>mllaunch.sh <libdir> <param1> <param2>

The first <libdir> sets the path to the mllaunch script itself. The second <libdir> is passed as the first argument to the script so that it can add this to the PATH (or LD_LIBRARY_PATH). The <param1> and <param2> arguments are the arguments to the MATLAB application passed in the normal GenePattern manner.

Module Parameters

Parameters to be passed to the MATLAB generated application should be entered in the usual manner as described in the GenePattern Tutorial.
All parameters will be passed into MATLAB from the command line as Strings.

Adding Support Files

For a MATLAB generated application, there are minimally two file required to be added as support files for the application. These are the application executable, and the application *.ctf file that were generated by the MATLAB Compiler. If your application requires additional files for its execution these should also be added as support files for the Module.

Distribution Licensing

MATLAB Licensing Agreement: http://www.mathworks.com/company/aboutus/policies_statements/agreement.pdf

Should you choose to distribute your MATLAB based module to others, you must ensure you are in compliance with the MATLAB licensing agreement whose URL is above. Please refer to the agreement itself for exact details, but I will summarize a few salient points here for GenePattern module developers.

- You may not distribute code that uses MATLAB that competes with one of TheMathWorks products
- You may not modify or remove any license file included with the MCR Libraries
- Module users must be made aware of the MATLAB license agreement in documentation and accept them before installation.
- The application's about box or an equivalent "visible" locaiton must include the legend "MATLAB copyright 1984-2004 the MathWorks Inc.", where the year '2004' represents the year you release your module.

You are responsible to review the MATLAB software license for all details to ensure you comply with them. The above list in no way exempts you from this responsibility.

References

The GenePattern Tutorial
The MATLAB Compiler

Appendix A, Example of Deploying a compiled MATLAB Application to Linux

This section will describe a step by step example of deploying a simple m-file as a GenePattern module on a GenePattern Server. Where the actions are different for Linux and Windows, it is so noted in the steps.

The example will be a simple MATLAB m-file that takes a filename and a String and writes the String out to a file with the given name.

A1, Writing the m-file

The first step is writing the MATLAB m-file that you want to share. For our example of writing a string to a file it consists of the following lines;

```
     % write the variable whatToWrite to a file called filename in the current directory
fid = fopen(filename,'w');

fprintf(fid,'#writing to a file\n\n');
fprintf(fid,whatToWrite);

fclose(fid);
```

A2, Adapting the m-file to be called from a comand line

In order to call it from the command line and pass it parameters, we need to turn this script into a no-return function. To do this we add a single line at the start of the m-file and save this file to the name of the function (i.e. writeToFile.m).

```
function writeToFile( filename, whatToWrite)
        % write the parameter whatToWrite to a file called filename in the current
directory

        fid = fopen(filename,'w');

        fprintf(fid,'#writing to a file\n\n');

        fprintf(fid,whatToWrite);

        fclose(fid);
```

A3, Compile the m-file

Within the MATLAB environment, we call the MATLAB Compiler to convert this function into an application.

```
>> mcc -m writeToFile
```

Within the current working directory, this creates the following files, we will use the only the ctf and the executable in the later steps.
  writeToFile
  writeToFile.ctf

A4, Prepare the GenePattern Server (Linux)

This step needs to be done only for the first MATLAB application you deploy. After you have done it the first time you can reuse the libraries you install for subsequent installations;

Within the MATLAB envronment, create the MCRInstaller zip file

```
>> buildmcr mcrdir
```

This will create a directory beneath the current working directory called "mcrdir" and create a file within it called "MCRInstaller.zip.

Copy this file to your GenePattern server (if it is a different machine) and install it into a directory there, For our example we will add a directory under the GenePatternServer directory called "matlab" and install the library files inside MCRInstaller.zip into that.

```
cd GenePatternServer

mkdir matlab

cd matlab

cp <path to mcrinstaller.zip>MCRInstaller.zip .

unzip MCRInstaller.zip
```

A5, Create the shell file to launch the application (Linux)

We will now create the shell file that sets the PATH and LD_LIBRARY_PATH variables for the environment, ensures the application is executable, and then calls the MATLAB generated application.

In a text editor we create the following file, mllaunch.sh.

```
#!/bin/csh
export MCRROOT=/home/username/GenePatternServer/matlab/v70

export
LD_LIBRARY_PATH=$1:$MCRROOT/runtime/glnx86:$MCRROOT/sys/os/glnx86:$MCRROOT/sys/java/
jre/glnx86/jre1.4.2/lib/i386/client:$MCRROOT/sys/java/jre/glnx86/jre1.4.2/lib/i386:$
MCRROOT/sys/opengl/lib/glnx86

export PATH=$1:$PATH
chmod a+x $1/testTwo
writeToFile $2 $3
```

Note that the MCR_ROOT variable is set to the v70 directory which we created by expanding MCRInstaller.zip.

A6, Create the GenePattern Module

Here we create the GenePattern module to execute the mllaunch.sh launching file. Create the module normally (i.e. entering author etc).

For the command line, we will enter the following;

```
sh <libdir>mllaunch.sh<libdir> <fname> <txt>
```

We will define two text variables

<fname>	the output file name
<txt>	the text to write to the file

We will upload the following files as support files to the module;

mllaunch.sh
whatToWrite
whatToWrite.ctf

A7, Save the module and try it out

As the final step, click on the "Save" button and execute the module. It should create two files, a stdout file that MATLAB writes out with some execution information, and a file with the text you gave it.

A8, Debugging

Problem:
```
error while loading shared libraries: libmwmclmcrrt.so.7.0: cannot open shared object
file: No such file or directory
```

This error (in the stdout file) indicates that you have not correctly set the path to the libraries you installed from MCRInstaller.zip. Double check the path. If it is indeed correct you may be using a different unix shell than was used in this example. Ensure that the correct command is being used to set the PATH and LD_LIBRARY_PATH ("export" in the example) in your mllaunch.sh.

Appendix B, Example of Deploying a compiled MATLAB Application toWindows

This section will describe a step by step example of deploying a simple m-file as a GenePattern module on a GenePattern Server. Where the actions are different for Linux and Windows, it is so noted in the steps.

The example will be a simple MATLAB m-file that takes a filename and a String and writes the String out to a file with the given name.

B1, writing the m-file

The first step is writing the MATLAB m-file that you want to share. For our example of writing a string to a file it consists of the following lines;

```
     % write the variable whatToWrite to a file called filename in the current directory
fid = fopen(filename,'w');

fprintf(fid,'#writing to a file\n\n');

fprintf(fid,whatToWrite);

fclose(fid);
```

B2, Adapting the m-file to be called from a comand line

In order to call it from the command line and pass it parameters, we need to turn this script into a no-return function. To do this we add a single line at the start of the m-file and save this file to the name of the function (i.e. writeToFile.m).

```
function writeToFile( filename, whatToWrite)
        % write the parameter whatToWrite to a file called filename in the current
directory

        fid = fopen(filename,'w');

        fprintf(fid,'#writing to a file\n\n');

        fprintf(fid,whatToWrite);

        fclose(fid);
```

B3, Compile the m-file

Within the MATLAB environment, we call the MATLAB Compiler to convert this function into an application.

```
>> mcc -m writeToFile
```

Within the current working directory, this creates the following files, we will use the only the ctf and the executable in the later steps.
writeToFile.exe
writeToFile.ctf

B4, prepare the GenePattern Server

If MATLAB is installed on this windows machine, this step can be skipped.

This step needs to be done only for the first MATLAB application you deploy. After you have done it the first time you can reuse the libraries you install for subsequent installations;

Copy <matlabroot>\toolbox\compiler\deploy\win32\MRCInstaller.exe to the server machine.

At the DOS prompt, or from windows explorer, run

```
    MCRInstaller.exe
```

B5 , Create the bat file to launch the application

We will now create the shell file that sets the PATH variable for the environment, and then calls the MATLAB generated application.

In a text editor we create the following file, mllaunch.bat;

```
    set LIBDIR=%1
    set PATH=%LIBDIR%;%PATH%
    writeToFile %2 %3
```

B6 , Create the GenePattern Module

Here we create the GenePattern module to execute the mllaunch.sh launching file.   Create the module normally (i.e. entering author etc).

For the command line, we will enter the following;

```
     <libdir>mllaunch.bat <libdir> <fname> <txt>
```

We will define two text variables

<fname> the output file name

<txt>   the text to write to the file

We will upload the following files as support files to the module;

mllaunch.bat
whatToWrite.exe
whatToWrite.ctf

B7, Save the module and try it out

As the final step, click on the "Save" button and execute the module. It should create two files, a stdout file that MATLAB writes out with some execution information, and a file with the text you gave it.

Appendix C, RunMatlab.java

```
import java.io.*;



/**

 * @author genepattern

 *

 * Runs a MATLAB function that exists in a directory passed in in the

 * first command line argument, with the remaining command-line args

 * being passed to the MATLAB function as String arguments.

 *

 * Assumes that MATLAB is on the path.

 *
 * operates by creating a startup.m file in the local working directory.
 * MATLAB reads this by default on startup.  The startup.m it creates
 * adds the libdir to the path, calls the function (with args) and then quits matlab.
 *
 * e.g.  java RunMatlab c:\mystuff fooFunction one two
 *
 * will write a startup.m that looks like this
 *
 * addpath c:\mystuff
 * fooFunction('one','two')
 * quit()
 *
 * and then exec
 *      matlab -nosplash -nodisplay
 */
public class RunMatlab {

    public static void main(String[] args) throws Exception {
       createStartupFile(args);

        String[] cmdArray = new String[3];
        cmdArray[0] = "matlab";
        cmdArray[1] = "-nosplash";
        cmdArray[2] = "-nodisplay";

        Process p = Runtime.getRuntime().exec(cmdArray);

        InputStream outs = p.getInputStream();

        Thread stdoutReader = copyStream(p.getInputStream(), System.out);
                Thread stderrReader = copyStream(p.getErrorStream(), System.err);
```

```
            // drain the output and error streams
            stdoutReader.start();
            stderrReader.start();

            p.waitFor();
}

protected static void createStartupFile(String[] args){
    try {
        File startupM = new File("startup.m");
        BufferedWriter bw = new BufferedWriter(new FileWriter(startupM));
        bw.write("addpath "+args[0]);
        bw.write("\n");
        bw.write(args[1]);
        bw.write("(");
        for (int i=2; i < args.length; i++){
            bw.write("'");
            bw.write(args[i]);
            bw.write("'");
            if ((i != (args.length - 1)) & (i != 0))
                bw.write(", ");
        }
        bw.write(")");
        bw.write("\n");
        bw.write("quit();");

        bw.flush();

        bw.close();
        startupM.deleteOnExit();
        } catch (IOException e){

            e.printStackTrace();

            System.err.println("Could not create startup.m file to execute.  Exitting.");

            System.exit(999);

        }

    }



        protected static Thread copyStream(final InputStream is, final PrintStream out)
throws IOException {

            // create thread to read from the a process' output or error stream

            Thread copyThread = new Thread(new Runnable(){

                public void run() {

                    BufferedReader in = new BufferedReader(new InputStreamReader(is));

                    String line;

                    // copy inputstream to outputstream


                    try {

                        boolean bNeedsBreak;

                        while((line = in.readLine())!=null){

                                out.println(line);

                        }

                    } catch (IOException ioe) {

                        System.err.println(ioe + " while reading from process stream");

                    }

                }

            });

            copyThread.setDaemon(true);

            return copyThread;

    }

}
```

Using GenePattern MATLAB File Support

Introduction

GenePattern File support for MATLAB is a set of MATLAB functions implemented in m-code and Java that can be used to read and write expression data files in the standard GenePattern formats including res, gct and odf.

It consists of two functions that you can use in your m-files,

loadGenePatternExpressionFile
writeGenePatternExpressionFile

Installation

1. Download the GenePattern MATLAB file support zip file
2. expand it into a directory (e.g. MATLAB7/toolbox)
3. From within MATLAB, add this directory to your path. Either of the following MATLAB commands will accomplish this.
   - >> addpath('<full path to matlab>/matlab7/toolbox/GenePattern');
   - >> rehash toolbox

   where <full path to matlab> is replaced by the file system path on your computer.

Example Use

The following example MATLAB m-file code will load a file called 'all_aml_test.res' that is in MATLAB's present working directory, reports the size of the data matrix, and then writes the data back out in the odf format. MATLAB responses are shown in italics.

```
>>all_aml_test = loadGenePatternExpressionFile('all_aml_test.res')

all_aml_test =

                  data: [7129x35 double]
              rowNames: [7129x1 java.lang.String[]]
           columnNames: [35x1 java.lang.String[]]
       rowDescriptions: [7129x1 java.lang.String[]]
    columnDescriptions: [35x1 java.lang.String[]]

>> size(all_aml_test.data)

ans =

        7129          35

>> writeGenePatternExpressionFile('all_aml_test_fromMATLAB', 'odf',all_aml_test.data, \
               all_aml_test.rowNames,
all_aml_test.columnNames,all_aml_test.rowDescriptions);
```

Function Definitions

```
function expressionDataset=loadGenePatternExpressionFile(path)
% Load a res, gct or odf file into a MATLAB structure
%
% Parameters
%   path    - full path to a res, gct or odf format expression file
%
% Return:  A MATLAB structure with the following elements
%   data                - M by N matrix of doubles
%   rowNames            - M*1 array of Strings
%   columnNames         - N*1 array of Strings
%   rowDescriptions     - M*1 array of strings
%   columnDescriptions  - unless loading from a gct file
%

function writeGenePatternExpressionFile(path, format, data, rowNames, columnNames,
rowDescriptions, colDescriptions, calls)
% Write a res, gct, mage or odf file with data provided.
%
% Parameters
%   path                - filename (incl path) to write to
%   format              - One of res, gct, odf, mageml
%   data                - M by N matrix of doubles
%   rowNames            - M*1 array of Strings
%   columnNames         - N*1 array of Strings
%   rowDescriptions     - M*1 array of strings
%   columnDescriptions  - unless loading from a gct filefrom a MATLAB
%   matrix (if absent, columnNames will be used for this)
%   calls               - M*N matrix of ints for A/P calls (res format only)
%
% Return
%   none.
```

GenePattern File Formats

This section describes each of the data file formats supported by GenePattern:

It also describes how to manually create GenePattern data files:

RES File Format

This is a tab delimited file format that is organized as follows:

1. The first line contains a list of labels identifying the samples associated with each of the columns in the remainder of the file. Two tabs (\t\t) separate the sample identifier labels because each sample contains two data values (an expression value and a present/marginal/absent call).
   - Line format: Description (tab) Accession (tab) (sample 1 name) (tab) (tab) (sample 2 name) (tab) (tab) ... (sample N name)
   - For example: **`Description Accession DLBC1_1 DLBC2_1 ... DLBC58_0`**
2. The second line contains a list of sample descriptions. Currently, GenePattern ignores these descriptions.
   - Line format: (tab) (sample 1 description) (tab) (tab) (sample 2 description) (tab) (tab) ... (sample N description)
   - For example, our RES file creation tool places the sample data file name and scale factors in this row: **`MG2000062219AA MG2000062256AA/scale factor=1.2172 ... MG2000062211AA/scale factor=1.1214`**
3. The third line contains a number indicating the number of rows in the data table that is contained in the remainder of the file. Note that the name and description columns are not included in the number of data columns.

    - Line format: (# of data rows)
    - For example: `7129`
4. The rest of the data file contains data for each of the genes. There is one row for each gene and two columns for each of the samples. The first two fields in the row contain the description and name for each of the genes (names and descriptions can contain spaces since fields are separated by tabs). The description field is optional but the tab following it is not. Each sample has two pieces of data associated with it: an expression value and an associated Absent/Marginal/Present (A/M/P) call. The A/M/P calls are generated by microarray scanning software (such as Affymetrix's GeneChip software) and are an indication of the confidence in the measured expression value. Currently, GenePattern ignores the Absent/Marginal/Present call.
    - Line format: (gene description) (tab) (gene name) (tab) (sample 1 data) (tab) (sample 1 A/P call) (tab) (sample 2 data) (tab) (sample 2 A/P call) (tab) ... (sample N data) (tab) (sample N A/P call)
    - For example: **`AFFX-BioB-5_at (endogenous control) AFFX-BioB-5_at -104 A -152 A ... -44 A`**

Sample RES file: all_aml_test.res

GCT File Format

The main difference between RES and GCT file formats is the RES file format contains labels for each gene's absent (A) versus present (P) calls as generated by Affymetrix's GeneChip software.

The GCT format is a tab delimited file format that is organized as follows:

1. The first line contains the version string and is always the same for this file format. Therefore, the first line must be as follows:
    - **`#1.2`**
2. The second line contains numbers indicating the size of the data table that is contained in the remainder of the file. Note that the name and description columns are not included in the number of data columns.
    - Line format: (# of data rows) (tab) (# of data columns)
    - For example: `7129 58`
3. The third line contains a list of identifiers for the samples associated with each of the columns in the remainder of the file.
    - Line format: Name (tab) Description (tab) (sample 1 name) (tab) (sample 2 name) (tab) ... (sample N name)
    - For example: **`Name Description DLBC1_1 DLBC2_1 ... DLBC58_0`**
4. The remainder of the data file contains data for each of the genes. There is one line for each gene and one column for each of the samples. The first two fields in the line contain name and descriptions for the genes (names and descriptions can contain spaces since fields are separated by tabs). The number of lines should agree with the number of data rows specified on line 2.
    - Line format: (gene name) (tab) (gene description) (tab) (col 1 data) (tab) (col 2 data) (tab) ... (col N data)
    - For example: **`AFFX-BioB-5_at AFFX-BioB-5_at (endogenous control) -104 -152 -158 ... -44`**

Occasionally, GCT files are organized in a transposed structure where the columns represent genes and the rows represent samples. The user should take care to check the organization of the file to ensure that the correct preprocessing is performed on the file. See sample *.gct files that come with the distribution for complete examples of the format.

Sample GCT file: allaml.dataset.gct

CLS File Format

The CLS files are simple files created to load class information into GenePattern. These files use spaces to separate the fields.

1. The first line of a CLS file contains numbers indicating the number of samples and number of classes. The number of samples should correspond to the number of samples in the associated RES or GCT data file.
    - Line format: (number of samples) (space) (number of classes) (space) 1
    - For example: **`58 2 1`**
2. The second line in a CLS file contains names for the class numbers. The line should begin with a pound sign (#) followed by a space.
    - Line format: # (space) (class 0 name) (space) (class 1 name)
    - For example: **`# cured fatal/ref`**
3. The third line contains numeric class labels for each of the samples. The number of class labels should be the same as the number of samples specified in the first line.
    - Line format: (sample 1 class) (space) (sample 2 class) (space) ... (sample N class)

- For example: `0 0 0 ... 1`

CLS file for sample RES file: all_aml_test.cls
CLS file for sample GCT file: allaml.phenotype.cls

ODF File Format

The ODF format is similar to the RES or GCT file formats for datasets. The main difference is in the header. The body of data still contains the expression level values for each gene in each sample. Thus the main data block (after the header lines) is a matrix of values. The columns are defined by a name and optionally a description. The rows have a name (name of the gene for instance) and a description (description of the gene). The columns contain the expression values for each gene in a sample. If the first gene in the data block is a particular Tyrosine Kinase then each of the samples contained in each of the columns will have expressions values for that particular Tyrosine Kinase in the first row.

ODF Header for Datasets

The following example shows the header lines of an ODF file. The first five lines are required.
The line numbers are shown for easy reference, they should not be included in your file.

```
1. ODF 1.0
2. HeaderLines=7
3. Model= Dataset
4. DataLines= 3
5. COLUMN_TYPES: String String float float float*
6. COLUMN_DESCRIPTIONS: Sample from DFCI Sample from UK Sample from Children's
7. COLUMN_NAMES: Name Description Sample 1 Biopsy_2 Biopsy_4
8. RowNamesColumn=0
9. RowDescriptionsColumn=1
```

Lines 1 and 2 are required first and second lines. They must both be present in the header and be the first and second lines. They signify that this is an ODF formatted file (of type 1.0) and indicate the number of header lines that follow before the main data block (in this case 7 more). Line 3, required to be somewhere in the header of an ODF file, defines this ODF file as containing Dataset data. Line 4 is required somewhere in the header file. It indicates the number of data rows present in the data block. Line 5 is required somewhere in the header file for any ODF file that has a main data block. It defines the type of data in each column. Line 6 is a tab delimited list of descriptions for each column. Line 7 is a tab delimited list of names for the columns. Line 8 defines which column will have the row names, and Line 9 defines which column will contain the row descriptions.

- The first element of each header line will be a key word. The keyword defines/describes what kind of meta data will be found on the rest of that line.
- A remark is a human readable comment that is skipped by the parser. This line starts with the "#" character and can contain any type of text since it is not parsed. Note that remarks are not counted as header lines and the user can insert them "by hand".
- The number of data lines can be quite large. For example, the U133A chip measures the expression values for about 20,000 human genes. A Dataset created from several samples using the U133A gene chip could have a large value for the DataLines tag.
- The COLUMN_NAMES:, COLUMN_DESCRIPTIONS:, and COLUMN_TYPES: lists must have the same number of elements. Also the number of elements must be equal to the number of columns in the main data block.
- The COLUMN_NAMES:, and COLUMN_DESCRIPTIONS: could be empty, that is simply contain the proper number of tabs but no text.

Main Data Block

The following example shows the first few lines of the main data block:

```
1000_at     X60188 HSERK1 Human ERK1 mRNA    145.3    240.37823    158.66888
1001_at     X60957 HSTIEMR Human tie mRNA    20.5     31.139397    14.053186
1002_f_at   X65962 HSCP450 H.sapiens mRNA    -9.6     118.06088    -8.287777
```

The main data block must be consistent with the header. The first COLUMN_NAMES element is "Name". This label is associated with the first column (values: 1000_at, 1001_at, and 1002_f_at). The second column's label is "Description" which is associated with the second column of the main data block. The next three columns are floating point numbers that represent the gene expression values for each of the samples.

- The first two columns are just text data, and next three columns only contain floating point values. This is consistent with the "String, String, float, float, float" elements in the COLUMN_TYPES: list.

Sample ODF file: all_aml_train.preprocessed0.odf

POL File Format

The POL file format represents a Parameter-Ordered List. This format is a tab-delimited file with 4 columns, consisting of the following:

1. A ranking (an integer corresponding to the rank of the feature)
2. Unique feature identifier
3. Feature description
4. Distance metric (value upon which the rank position is based)

If you don't have a distance metric, you can use the rank in column 4 as well. Lines from a sample pol file are shown below:

```
0       X59798_at       CCND1 Cyclin D10.0
1       S69272_s_at     Cytoplasmic antiproteinase    465.5319538
2       U37012_at       Cleavage and polyadenylation specificity factor      493.1997567
3       X69910_at       P63 mRNA for transmembrane protein    493.5331802
4       U53347_at       Neutral amino acid transporter B mRNA 515.3552173
5       M80899_at       AHNAK AHNAK nucleoprotein (desmoyokin)539.9990741
```

CDT File Format

CDT files will be created by the Hierarchical Clustering Algorithm. This is a format defined at Stanford for their Hierarchical Clustering program. More information on this format is available here.

GTR File Format

GTR files will be created by the Hierarchical Clustering Algorithm. This is a format defined at Stanford for their Hierarchical Clustering program. More information on this format is available here.

ATR File Format

ATR files will be created by the Hierarchical Clustering Algorithm. This is a format defined at Stanford for their Hierarchical Clustering program. More information on this format is available here.

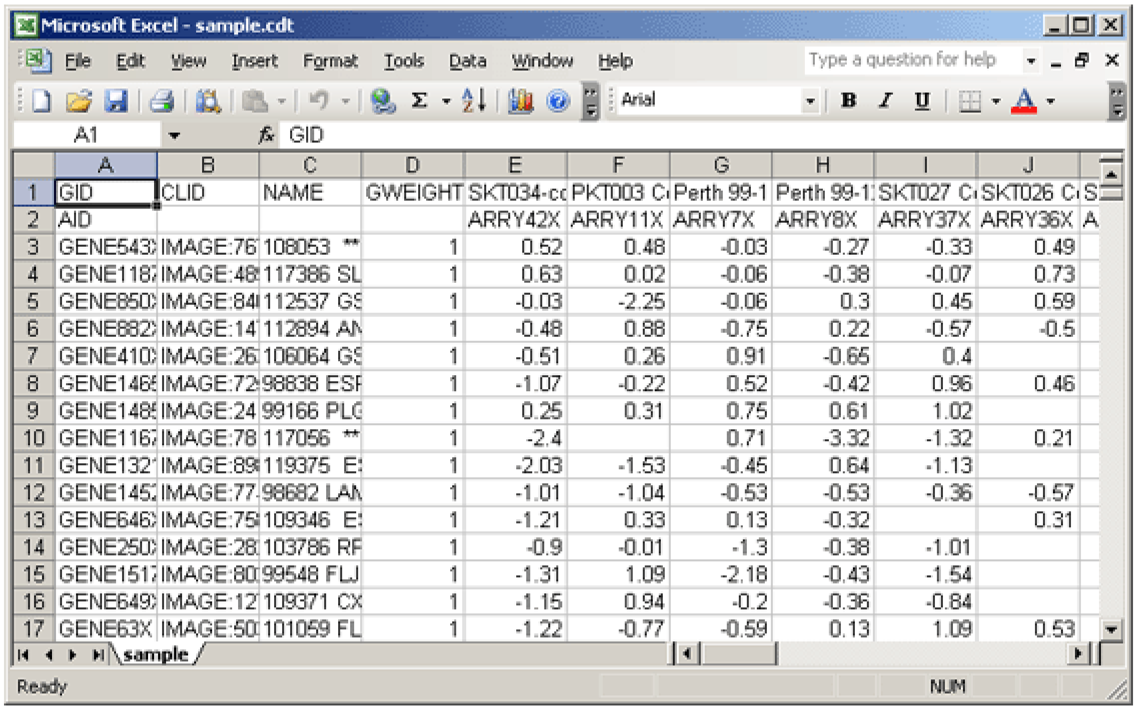
Creating Data Files

Creating GCT or RES files manually is relatively easy since most spreadsheet and database programs allow you to export your data into a file in a tab- separated format. Once you do this, you only need to load the file into a text editor or word processor, make the necessary format changes, and save the file as raw text.

Exercise 7a: Creating Data Files for GenePattern

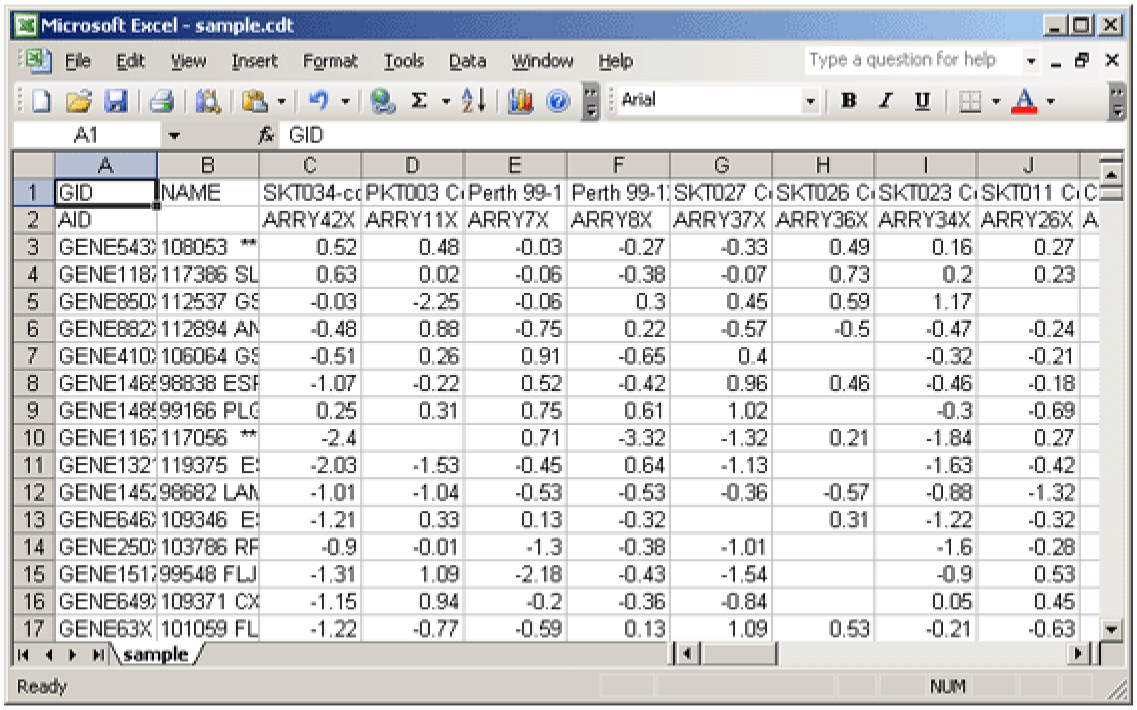
This exercise provides an example of converting a CDT file (sample.cdt) to a GCT file that can be used with GenePattern (sample.imputed.gct). To convert the CDT file:

- Save the (sample.cdt) file to your local drive and open it in Microsoft Excel.


Microsoft Excel - sample.cdt
File Edit View Insert Format Tools Data Window Help
Type a question for help
Arial
A1
GID
sample
Ready
NUM

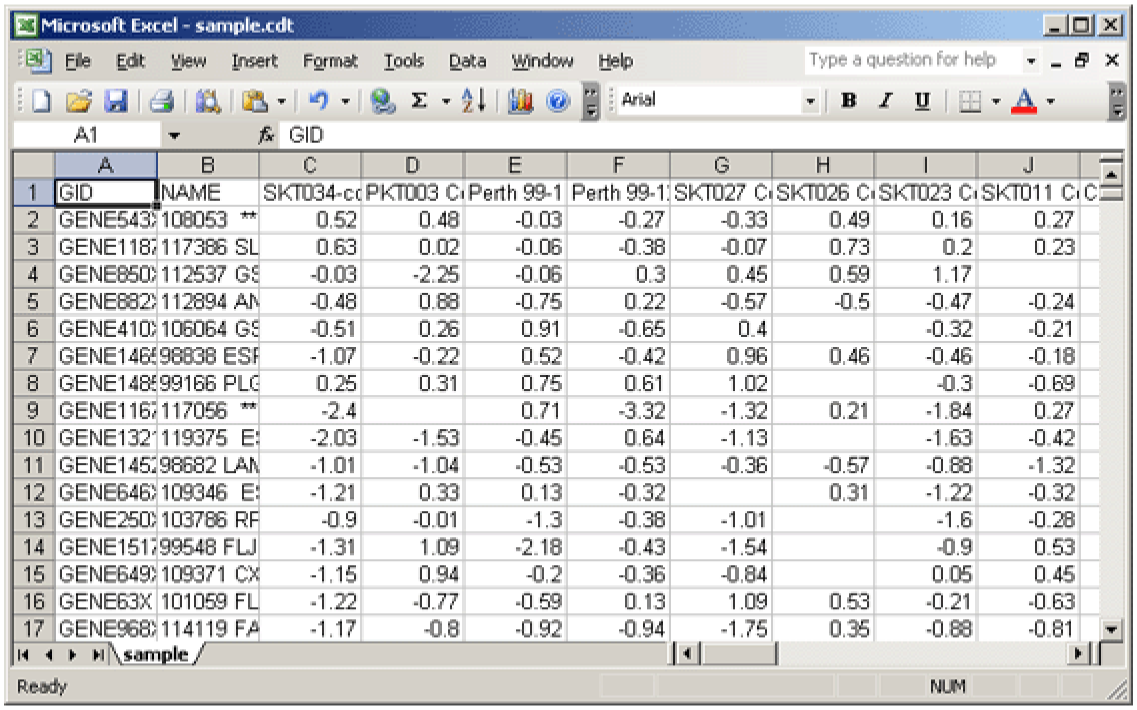
	A	B	C	D	E	F	G	H	I	J
1	GID	CLID	NAME	GWEIGHT	SKT034-cc	PKT003 C	Perth 99-1	Perth 99-1	SKT027 C	SKT026 C
2	AID				ARRY42X	ARRY11X	ARRY7X	ARRY8X	ARRY37X	ARRY36X
3	GENE543	IMAGE:76	108053 **	1	0.52	0.48	-0.03	-0.27	-0.33	0.49
4	GENE118	IMAGE:48	117386 SL	1	0.63	0.02	-0.06	-0.38	-0.07	0.73
5	GENE850	IMAGE:84	112537 GS	1	-0.03	-2.25	-0.06	0.3	0.45	0.59
6	GENE882	IMAGE:14	112894 AN	1	-0.48	0.88	-0.75	0.22	-0.57	-0.5
7	GENE410	IMAGE:26	106064 GS	1	-0.51	0.26	0.91	-0.65	0.4	
8	GENE146	IMAGE:72	98838 ESF	1	-1.07	-0.22	0.52	-0.42	0.96	0.46
9	GENE148	IMAGE:24	99166 PLC	1	0.25	0.31	0.75	0.61	1.02	
10	GENE116	IMAGE:78	117056 **	1	-2.4		0.71	-3.32	-1.32	0.21
11	GENE132	IMAGE:89	119375 E	1	-2.03	-1.53	-0.45	0.64	-1.13	
12	GENE145	IMAGE:77	98682 LAN	1	-1.01	-1.04	-0.53	-0.53	-0.36	-0.57
13	GENE646	IMAGE:75	109346 E	1	-1.21	0.33	0.13	-0.32		0.31
14	GENE250	IMAGE:28	103786 RF	1	-0.9	-0.01	-1.3	-0.38	-1.01	
15	GENE151	IMAGE:80	99548 FLJ	1	-1.31	1.09	-2.18	-0.43	-1.54	
16	GENE649	IMAGE:12	109371 CX	1	-1.15	0.94	-0.2	-0.36	-0.84	
17	GENE63X	IMAGE:50	101059 FL	1	-1.22	-0.77	-0.59	0.13	1.09	0.53

- Delete the CLID and GWEIGHT columns. The GCT file format allows for only two columns of annotations.


Microsoft Excel - sample.cdt
File Edit View Insert Format Tools Data Window Help
Type a question for help
Arial
A1
GID
sample
Ready
NUM

	A	B	C	D	E	F	G	H	I	J
1	GID	NAME	SKT034-cc	PKT003 C	Perth 99-1	Perth 99-1	SKT027 C	SKT026 C	SKT023 C	SKT011 C
2	AID		ARRY42X	ARRY11X	ARRY7X	ARRY8X	ARRY37X	ARRY36X	ARRY34X	ARRY26X
3	GENE543	108053 **	0.52	0.48	-0.03	-0.27	-0.33	0.49	0.16	0.27
4	GENE118	117386 SL	0.63	0.02	-0.06	-0.38	-0.07	0.73	0.2	0.23
5	GENE850	112537 GS	-0.03	-2.25	-0.06	0.3	0.45	0.59	1.17	
6	GENE882	112894 AN	-0.48	0.88	-0.75	0.22	-0.57	-0.5	-0.47	-0.24
7	GENE410	106064 GS	-0.51	0.26	0.91	-0.65	0.4		-0.32	-0.21
8	GENE146	98838 ESF	-1.07	-0.22	0.52	-0.42	0.96	0.46	-0.46	-0.18
9	GENE148	99166 PLC	0.25	0.31	0.75	0.61	1.02		-0.3	-0.69
10	GENE116	117056 **	-2.4		0.71	-3.32	-1.32	0.21	-1.84	0.27
11	GENE132	119375 E	-2.03	-1.53	-0.45	0.64	-1.13		-1.63	-0.42
12	GENE145	98682 LAN	-1.01	-1.04	-0.53	-0.53	-0.36	-0.57	-0.88	-1.32
13	GENE646	109346 E	-1.21	0.33	0.13	-0.32		0.31	-1.22	-0.32
14	GENE250	103786 RF	-0.9	-0.01	-1.3	-0.38	-1.01		-1.6	-0.28
15	GENE151	99548 FLJ	-1.31	1.09	-2.18	-0.43	-1.54		-0.9	0.53
16	GENE649	109371 CX	-1.15	0.94	-0.2	-0.36	-0.84		0.05	0.45
17	GENE63X	101059 FL	-1.22	-0.77	-0.59	0.13	1.09	0.53	-0.21	-0.63

- Delete the second row, which contains array identifiers (AID). The GCT file format allows for only one row of identifiers.

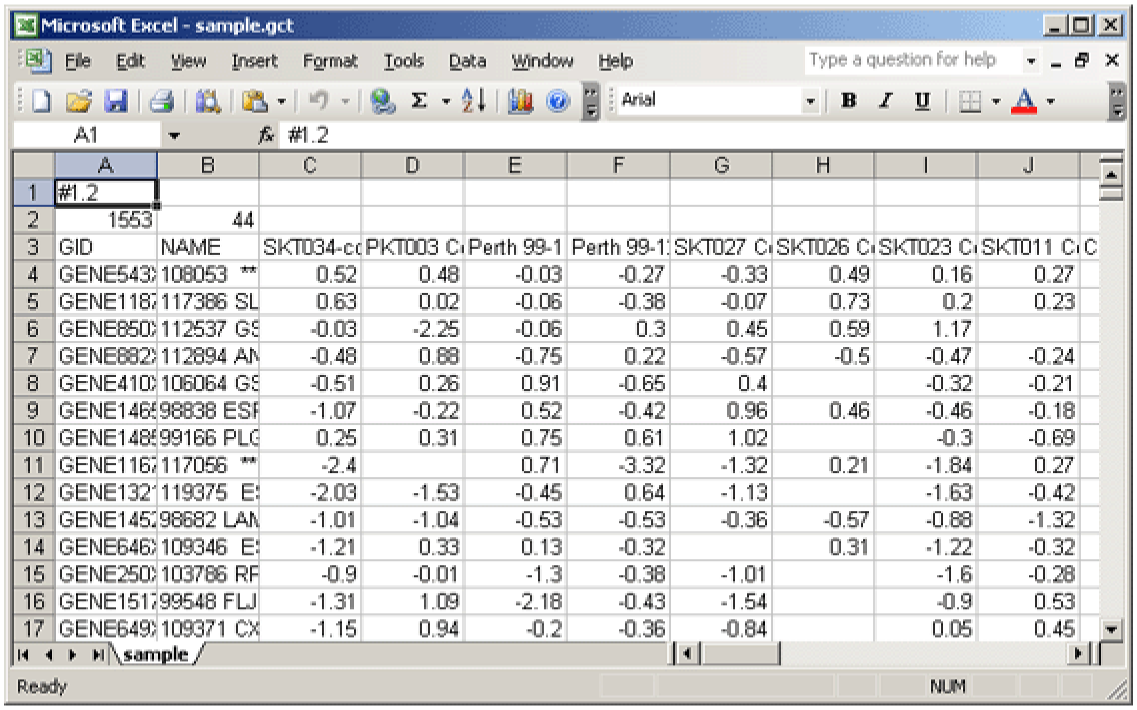


Microsoft Excel - sample.cdt

A1 GID

	A	B	C	D	E	F	G	H	I	J
1	GID	NAME	SKT034-cc	PKT003 C	Perth 99-1	Perth 99-1	SKT027 C	SKT026 C	SKT023 C	SKT011 C
2	GENE543	108053 **	0.52	0.48	-0.03	-0.27	-0.33	0.49	0.16	0.27
3	GENE118	117386 SL	0.63	0.02	-0.06	-0.38	-0.07	0.73	0.2	0.23
4	GENE850	112537 GS	-0.03	-2.25	-0.06	0.3	0.45	0.59	1.17	
5	GENE882	112894 AN	-0.48	0.88	-0.75	0.22	-0.57	-0.5	-0.47	-0.24
6	GENE410	106064 GS	-0.51	0.26	0.91	-0.65	0.4		-0.32	-0.21
7	GENE146	98838 ESF	-1.07	-0.22	0.52	-0.42	0.96	0.46	-0.46	-0.18
8	GENE148	99166 PLG	0.25	0.31	0.75	0.61	1.02		-0.3	-0.69
9	GENE116	117056 **	-2.4		0.71	-3.32	-1.32	0.21	-1.84	0.27
10	GENE132	119375 E	-2.03	-1.53	-0.45	0.64	-1.13		-1.63	-0.42
11	GENE145	98682 LAN	-1.01	-1.04	-0.53	-0.53	-0.36	-0.57	-0.88	-1.32
12	GENE646	109346 E	-1.21	0.33	0.13	-0.32		0.31	-1.22	-0.32
13	GENE250	103786 RF	-0.9	-0.01	-1.3	-0.38	-1.01		-1.6	-0.28
14	GENE151	99548 FLJ	-1.31	1.09	-2.18	-0.43	-1.54		-0.9	0.53
15	GENE649	109371 CX	-1.15	0.94	-0.2	-0.36	-0.84		0.05	0.45
16	GENE63X	101059 FL	-1.22	-0.77	-0.59	0.13	1.09	0.53	-0.21	-0.63
17	GENE968	114119 FA	-1.17	-0.8	-0.92	-0.94	-1.75	0.35	-0.88	-0.81

- Add two header rows at the top of the file:
  - In the first row, first cell, enter: #1.2
  - In the second row, first cell, enter the number of data rows: 1553
  - In the second row, second cell, enter the number of data columns: 44



Microsoft Excel - sample.gct

A1 #1.2

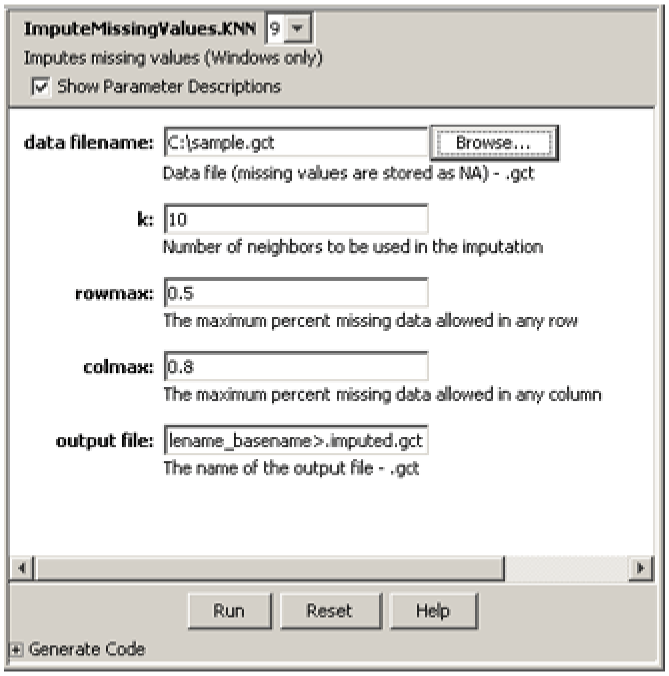
	A	B	C	D	E	F	G	H	I	J
1	#1.2									
2	1553	44								
3	GID	NAME	SKT034-cc	PKT003 C	Perth 99-1	Perth 99-1	SKT027 C	SKT026 C	SKT023 C	SKT011 C
4	GENE543	108053 **	0.52	0.48	-0.03	-0.27	-0.33	0.49	0.16	0.27
5	GENE118	117386 SL	0.63	0.02	-0.06	-0.38	-0.07	0.73	0.2	0.23
6	GENE850	112537 GS	-0.03	-2.25	-0.06	0.3	0.45	0.59	1.17	
7	GENE882	112894 AN	-0.48	0.88	-0.75	0.22	-0.57	-0.5	-0.47	-0.24
8	GENE410	106064 GS	-0.51	0.26	0.91	-0.65	0.4		-0.32	-0.21
9	GENE146	98838 ESF	-1.07	-0.22	0.52	-0.42	0.96	0.46	-0.46	-0.18
10	GENE148	99166 PLG	0.25	0.31	0.75	0.61	1.02		-0.3	-0.69
11	GENE116	117056 **	-2.4		0.71	-3.32	-1.32	0.21	-1.84	0.27
12	GENE132	119375 E	-2.03	-1.53	-0.45	0.64	-1.13		-1.63	-0.42
13	GENE145	98682 LAN	-1.01	-1.04	-0.53	-0.53	-0.36	-0.57	-0.88	-1.32
14	GENE646	109346 E	-1.21	0.33	0.13	-0.32		0.31	-1.22	-0.32
15	GENE250	103786 RF	-0.9	-0.01	-1.3	-0.38	-1.01		-1.6	-0.28
16	GENE151	99548 FLJ	-1.31	1.09	-2.18	-0.43	-1.54		-0.9	0.53
17	GENE649	109371 CX	-1.15	0.94	-0.2	-0.36	-0.84		0.05	0.45

- Save the modified file as a text (tab delimited) file with the name `sample.gct`.

Your original CDT file contained cells that were missing data. Most GenePattern modules require that all cells in a GCT file contain data. You can use the GenePattern analysis module ImputeMissingValues.KNN to add the missing data to your GCT file. To do so:

- Launch either the GenePattern Java or Web Client.
- Select the ImputeMissingValues.KNN analysis task.

- For the **`data filename`** parameter, enter your `sample.gct` file.



ImputeMissingValues.KNN 9

Imputes missing values (Windows only)

☑ Show Parameter Descriptions

data filename: C:\sample.gct
Data file (missing values are stored as NA) - .gct

k: 10
Number of neighbors to be used in the imputation

rowmax: 0.5
The maximum percent missing data allowed in any row

colmax: 0.8
The maximum percent missing data allowed in any column

output file: lename_basename>.imputed.gct
The name of the output file - .gct

- Run the analysis.

The results file, `sample.imputed.gct`, can be sent to most GenePattern modules.