

# PAM

## “Prediction Analysis of Microarrays”

### *Users guide and manual*

Trevor J. Hastie <sup>\*</sup>    Balasubramanian Narasimhan <sup>†</sup>    Robert J. Tibshirani <sup>‡</sup>  
Gilbert Chu <sup>§</sup>

## Contents

<b>1</b>	<b>PAM User Group Announcement</b>	<b>3</b>
<b>2</b>	<b>Summary of Changes</b>	<b>3</b>
2.1	Changes in PAM 2.0 . . . . .	3
2.2	Changes in PAM 1.23 . . . . .	4
2.3	Changes in PAM 1.22 . . . . .	4
2.4	Changes in PAM 1.21a . . . . .	4
2.5	Changes in PAM 1.21 . . . . .	4
2.6	Changes in PAM 1.20 . . . . .	4
2.7	Changes in PAM 1.12 . . . . .	5
2.8	Changes in PAM 1.10 . . . . .	5
2.9	Changes in PAM 1.06 . . . . .	5
2.10	Changes in PAM 1.05 . . . . .	5
<b>3</b>	<b>Introduction</b>	<b>5</b>
<b>4</b>	<b>Obtaining PAM</b>	<b>5</b>

---

<sup>\*</sup>Department of Statistics and Department of Health Research & Policy, Stanford University, Stanford CA 94305.  
Email: hastie@stat.stanford.edu.

<sup>†</sup>Department of Statistics and Department of Health Research & Policy, Stanford University, Stanford CA 94305.  
Email: naras@stat.stanford.edu.

<sup>‡</sup>Department of Health Research & Policy and Department of Statistics, Stanford University, Stanford CA 94305.  
Email: tibs@stat.stanford.edu.

<sup>§</sup>Department of Biochemistry, Stanford University, Stanford CA 94305. Email: chu@cmgm.stanford.edu.

<b>5</b>	<b>System Requirements</b>	<b>6</b>
<b>6</b>	<b>Installation</b>	<b>6</b>
6.1	New installation . . . . .	6
6.2	Upgrading from Previous Version . . . . .	7
<b>7</b>	<b>Uninstalling PAM</b>	<b>7</b>
<b>8</b>	<b>Documentation</b>	<b>7</b>
<b>9</b>	<b>Examples</b>	<b>7</b>
<b>10</b>	<b>Data Formats</b>	<b>8</b>
10.1	Batch Labels and Batch Adjustment . . . . .	9
10.2	Normalization of experiments . . . . .	9
<b>11</b>	<b>Handling Missing Data</b>	<b>9</b>
<b>12</b>	<b>Running PAM: classification problems</b>	<b>10</b>
12.1	Training the Classifier . . . . .	14
12.2	Test Set Prediction . . . . .	22
12.3	Using data in Multiple Sheets . . . . .	22
12.4	Format of the Gene Listing . . . . .	23
<b>13</b>	<b>Running PAM: Survival analysis and regression</b>	<b>23</b>
13.1	Basic idea of supervised principal components . . . . .	23
13.2	Opening screen . . . . .	25
13.3	Training, Cross-validation and Test Set Prediction . . . . .	25
13.4	Learning competing predictors from expression data . . . . .	27
<b>14</b>	<b>Details of the R Excel Connection</b>	<b>27</b>
<b>15</b>	<b>Frequently Asked Questions</b>	<b>27</b>
15.1	General Questions . . . . .	27
15.2	PAM Registration Questions . . . . .	28
15.3	Installation, Uninstallation Questions . . . . .	28
15.4	PAM Usage Questions . . . . .	29
<b>16</b>	<b>Technical Details- Classification</b>	<b>32</b>
16.1	Details of the centroid shrinkage . . . . .	32
16.2	Choosing the amount of shrinkage . . . . .	33
16.3	Class probabilities and discriminant functions . . . . .	33

16.4	Relationship to other approaches . . . . .	34
<b>17</b>	<b>Technical details- supervised principal components</b>	<b>37</b>
17.1	Extracting supervised princapl components . . . . .	37
17.2	Importance scores and a reduced predictor . . . . .	39

## List of Figures

1	Data Format and Highlighting data . . . . .	11
2	The PAM Dialog Box . . . . .	12
3	The PAM Controller . . . . .	14
4	The PAM Controller after training . . . . .	15
5	PAM Plots . . . . .	16
6	More PAM Plots . . . . .	17
7	The PAM Prediction Dialog . . . . .	19
8	The PAM Prediction output for Khan data . . . . .	20
9	The test error and probability plots for Khan data . . . . .	21

## List of Tables

### 1 PAM User Group Announcement

To foster communication between PAM users and make new announcements, we plan to establish a new Yahoo user group. When the group is active, we'll put a notice in this manual as well as the web site. Stay tuned.

### 2 Summary of Changes

The following are changes since the initial release of PAMR 1.02. Note that all releases of *PAMR for Excel* prior to PAM 1.10 were internal releases.

#### 2.1 Changes in PAM 2.0

This is a major new release of PAM. We have implemented the “supervised principal components” method for analyzing survival and regression outcomes. The old PAM survival method- discretizing the time axis and then applying PAM/classification, was not very effective and is no longer available

## 2.2 Changes in PAM 1.23

Bug fix release.

- Fixed bug in `pamr.predict` and gene names in the correct order.
- Fixed bug in `nscv` affecting cv folds: balance and number of folds

## 2.3 Changes in PAM 1.22

Bug fix release.

- Fixed `pamr.to.excel` so that it now expects the gene id and gene names in the correct order.
- Fixed `nscv.R` to match R handling of arguments. This resulted in cross validation failing.

## 2.4 Changes in PAM 1.21a

Changed the installer to include a Stanford Tool manager that would manage all Stanford tools. PAM can be activated and deactivated using the menu.

Changed `pamr.listgenes` to fix a typo. This cause problems.

## 2.5 Changes in PAM 1.21

Fixed `nscv` so that `unclass` is used instead of `codes`. This used to make PAM bomb in R versions 1.9.0 and higher.

Changed `pamr.listgenes` so that it doesn't bomb when significant gene list is of length 1. Reported by Markus Ruschhaupt.

Fixed `pamr.confusion` bug that screwed up individual plots.

Long standing request of Rob Gentleman—to add a namespace. The `excel pamr` package always had a namespace, but now Rob Tibshirani is syncing his source as well.

## 2.6 Changes in PAM 1.20

Added survival analysis capability. Replaced the default missing data imputation algorithm with a faster and more accurate version; the old algorithm is available for reproducibility, but deprecated. Streamlined the installation process to make it more user-friendly: PAM now installs and uninstalls like a standard windows application. Reorganized the controller buttons into more sensible groupings.

Versions 1.13–1.19 were internal releases and skipped.

## 2.7 Changes in PAM 1.12

Bug fixes, mostly in installation process.

## 2.8 Changes in PAM 1.10

Fixed batch adjustment bug. New dialogs added and so the PAM GUI interface is a bit different. Added test error and test probability plots. First public release.

## 2.9 Changes in PAM 1.06

The first stable version that was beta tested internally.

## 2.10 Changes in PAM 1.05

PAMR now has a companion: an Excel front-end that uses the R DCOM connector.

# 3 Introduction

PAM (Prediction Analysis of Microarrays) is a statistical technique for class prediction from gene expression data using nearest shrunken centroids. It is described in Tibshirani, Hastie, Narasimhan, and Chu (2002). The method of nearest shrunken centroids identifies subsets of genes that best characterize each class. The technique is general and can be used in many other classification problems. It can also be applied to survival analysis problems.

PAM Software for the R package (Ihaka and Gentleman, 1996) has been available for some time now. This manual describes the same software adapted to an Excel front-end. Our goal was to produce an environment much like that of the SAM (Tusher, Tibshirani, and Chu, 2001) except with R as the computation engine. A key component that makes this possible is the R DCOM server by Thomas Baier (see <http://cran.us.r-project.org/other-software.html>).

The new version PAM 2.0 includes for the first time the “supervised principal components” method for survival and regression outcomes. This utilizes the R package “superpc”. One can build gene expression predictors for survival times, and can also compare them to existing clinical predictors like grade, stage etc.

# 4 Obtaining PAM

PAM can be freely downloaded from the url <http://www-stat.stanford.edu/~tibs/PAM>. Please note that the Excel front-end is an *addition* to PAM for R. Therefore, new users must download both the *PAM for R* package and the *PAM for Excel* package.

## 5 System Requirements

PAM for Excel requires:

- The latest updates for your operating system available from <http://windowsupdate.microsoft.com>. To prevent any problems, access this and other Microsoft sites using **Internet Explorer** rather than Netscape. Clicking on the **Product Updates** link pops up a box that will automate the installation of the latest patches. Beware that several (time-consuming) reboots are usually needed and you might need administrative privileges to install the patches. It is generally a good idea to update your system for security reasons any way.
- Microsoft Excel 97 or higher. We recommend that users install appropriate Microsoft Office service packs that are available from <http://officeupdate.microsoft.com>. Office 97 users are especially encouraged to do so; there are two service packs for Office 97. The Office 97 service packs are not easy to find; one often has to search the Microsoft website to access them.
- The latest version of **R**. This is freely available from the web-site <http://www.r-project.org>. Use any of the mirrors and download a Windows executable version. The installation is very simple; one has to merely run the setup program.

Please note that people have reported some problems with *PAM for Excel* when multiple versions of R as installed on the same computer. If that is the case with your computer, you might want to uninstall all but the latest version.

We have tested this version (1.20) of software with R version 1.7.1 on the following platforms: Excel 97 and Windows NT, Excel 2000 and Windows 98, Excel 2000 and Windows ME, Excel 2000 and Windows XP, Excel 2002 and Windows XP.

## 6 Installation

Prior to version 1.20, the installation of *PAM for Excel* was quite involved. One had to install an R package and follow it up with the installation of the PAM Excel addin.

The installation/removal is far simpler now, and will be for future versions. However, those of you who are upgrading from earlier versions of *PAM for Excel* should follow the instructions below quite carefully to avoid problems.

### 6.1 New installation

If you have not installed *PAM for excel* before, this section is for you.

Double click on the `pamsetup.exe` file and follow the instructions. If all went well, PAM will be automatically available the next time you fire up Excel.

## 6.2 Upgrading from Previous Version

If you had previously installed an older version of *PAM for Excel* (i.e., a version earlier than 1.20), this section is for you.

1. Fire up Excel and click on the `Tools` menu. Choose `Addins` and uncheck the box against the phrase `Prediction Analysis for Microarrays` or `Pam`.
2. Use the `Control Panel` to uninstall PAM software. If you are asked if shared components should be kept and not discarded, elect to keep them as a conservative measure, unless you are really hard-pressed for space.
3. Fire up R. In the command window, type the command: `remove.packages("pamr")`. This will remove the R package.
4. Double click on `pamsetup.exe` and follow the instructions. PAM will be automatically loaded the next time you fire up Excel.

## 7 Uninstalling PAM

To uninstall *PAM for Excel*, choose the `Uninstall PAM for Excel` item from the **Programs** menu.

## 8 Documentation

After PAM has been installed, the manual is also available as a PDF file in the subdirectory `doc` of the PAM installation directory.

If you don't already have a PDF reader installed, you can do so from the web-site `http://www.adobe.com`.

## 9 Examples

Some examples of the use of PAM are in the directory `C:\Program Files\PAMVB\Examples` in the default installation. All the examples contain essentially one dataset, that described in the PNAS paper. They are meant to familiarize the users with the format in which PAM expects the data.

We briefly describe the examples below.

**khan** This is the data discussed in the PNAS paper (Tibshirani et al., 2002). It contains two sheets, the data on one sheet and prediction data on another.

**khan-missing** The same data as above with some missing values.

**khan-multi** The same data as the first but spread over multiple sheets. This allows one to overcome the limitation of 256 columns in a single worksheet in Excel.

**khan-multi-missing** Same data as the one before with some missing values.

**survival** A dataset with censored survival times.

Instructions on using PAM on these examples is discussed in section 12.

## 10 Data Formats

The data should be put in an Excel spreadsheet. The format should be familiar to users of SAM.

PAM now handles three kinds of problems: a standard classification problem, survival analysis and regression.

**Standard Classification Problem** PAM requires the training data set to contain **Class Labels**. Optionally, **Sample Labels** and **Batch Labels** can be specified when available. Although sample labels are optional, they are highly recommended for identifying samples in PAM output. These can appear in the first few rows of the data set in any order; all remaining rows have gene expression data, one row per gene. The columns represent the different experimental samples.

- The first few rows can contain the class labels, sample labels and batch labels, in any order. The last two are optional, but the class labels are required. The data must start at column 3. See the examples for details.
- The remaining lines contain gene expression measurements one line per gene. We describe the format below.

**Column 1** This should contain the gene name. It is for the user's reference.

**Column 2** This should contain the gene ID, for the user's reference. Note that the gene ID column is the column that is linked to the SOURCE website by PAM. Hence a unique identifier (e.g. Clone ID, Accession number, Gene Name/Symbol or LocusLink ID) should be used in this column, if SOURCE web-site gene lookup is desired.



**Remaining Columns** These should contain the expression measurements as numbers. Missing expression measurements can be left blank or coded as NA, if desired. (The latter is done easily in a good editor or in Excel. In Excel, to change blank fields to NA, choose all columns, pull down the **Edit** menu, choose **Replace** and then **nothing (Blank)** with NA).

The expression measurements need not start directly after the label rows. They can start further down. For example, rows 1 and 2 can contain the sample and class labels and row 15 can be the start of the expression data. The intervening rows will be ignored.

**Survival Analysis and Regression Problems** For survival analysis, the data format is the same as above, except that in lieu of **Class Labels**, one specifies **Survival time** and **Censoring status**. Survival times must be real numbers, in any units (days, months etc.); the censoring indicator can be coded as 1=event (e.g. death) and 0=censored), or as “uncensored” and “censored”. For regression, one specifies the **Outcome**. For both survival and regression problems, sample labels are not required but highly recommended. If you want to compare the gene expression predictor to other competing predictors, sample labels *must* be present. Batch labels are optional as before.

## 10.1 Batch Labels and Batch Adjustment

When batch labels are specified, PAM adjusts the data for the estimated batch effect. Specifically, a one-way analysis of variance is carried out for each gene, with batch as the grouping variable. The residuals from this ANOVA are the adjusted data. In other words, each gene is centered to have mean zero within each batch.

## 10.2 Normalization of experiments

Different experimental platforms require different normalizations. Therefore, *the user is required to normalize the data from the different experiments (columns) before running PAM.*

**PAM does not do any normalization!**

For cDNA data, centering the columns of the expression matrix (that is, making the columns mean equal to zero) is often sufficient.

For oligonucleotide data, a stronger calibration may be necessary: for example, a linear normalization of the data for each experiment versus the row-wise average for all experiments.

# 11 Handling Missing Data

PAM checks for missing values and imputes them if needed. Currently, the only engine that is used is **K-Nearest Neighbor**. The default set up uses 10 nearest neighbors. Here’s is a description of

how it works.

**K-Nearest Neighbor** In the other (default) option — missing values are imputed using a  $k$ -nearest neighbor average in gene space (default  $k = 10$ ):

1. For each gene  $i$  having at least one missing value:
  - (a) Let  $S_i$  be the samples for which gene  $i$  has no missing values.
  - (b) find the  $k$  nearest neighbors to gene  $i$ , using only samples  $S_i$  to compute the Euclidean distance. When computing the Euclidean distances, other genes may have missing values for some of the samples  $S_i$ ; the distance is averaged over the non-missing entries in each comparison.
  - (c) impute the missing sample values in gene  $i$ , using the averages of the non-missing entries for the corresponding sample from the  $k$  nearest neighbors.
2. If a gene still has missing values after the above steps, impute the missing values using the average (non-missing) expression for that gene.

If the number of genes is large, the near-neighbor computations above can take too long. To overcome this, we combine the K-Nearest Neighbor imputation algorithm with a **Recursive Two-Means Clustering** procedure:

1. If number of genes  $p$  is greater than  $p_{max}$  (default 1500):
  - (a) Run a two-means clustering algorithm in gene space, to divide the genes into two more homogeneous groups. The distance calculations use averages over non-missing entries, as do the mean calculations.
  - (b) Form two smaller expression arrays, using the two subsets of genes found in (a). For each of these, recursively repeat step 1.
2. If  $p$  is less than  $p_{max}$ , impute the missing genes using K-Nearest-Neighbor averaging.

This imputation part is actually self-contained and will appear in the near future as a stand-alone R package.

## 12 Running PAM: classification problems

It is best to familiarize yourself with PAM using the examples provided. We'll assume that you have loaded the data in the example file **khan-missing.xls** into Excel. Generally, to begin, you highlight an area of the spreadsheet that represents the data by first clicking on the top-left corner and then shift-clicking on the bottom right corner of the rectangle. Then, click on the PAM button in the toolbar. See illustration in figure 1.

A dialog form shown in figure 2 now pops up. Several input parameters have to be specified.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
	sample1	sample2	sample3	sample4	sample5	sample6	sample7	sample8	sample9	sample10	sample11	sample12	sample13	sample14	sample15	sample16	sample17	sample18
1																		
2	GEN1	adrenin (caldesin)	0.43504276	0.1677761	0.1851657	0.1755307	0.1653904	0.1631629	0.1703763	0.1616145	0.1703763	0.1616145	0.1703763	0.1616145	0.1703763	0.1616145	0.1703763	0.1616145
3	GEN2	adrenin	0.43504276	0.1677761	0.1851657	0.1755307	0.1653904	0.1631629	0.1703763	0.1616145	0.1703763	0.1616145	0.1703763	0.1616145	0.1703763	0.1616145	0.1703763	0.1616145
4	GEN3	phosphotrikinase	-0.45026158	0.41271663	-0.24130732	0.62566493	0.85263549	0.13584123	0.39202189	0.23029169	0.60509474	-0.39202189	0.60509474	-0.39202189	0.60509474	-0.39202189	0.60509474	-0.39202189
5	GEN4	tyrosine c-1	-2.72115441	-2.82514573	-2.87523812	-1.74125487	0.27269533	0.43838429	0.38232189	0.23029169	0.60509474	-0.39202189	0.60509474	-0.39202189	0.60509474	-0.39202189	0.60509474	-0.39202189
6	GEN5	uroporphyrinogen III	-1.21705844	0.632636475	-0.889465403	-0.845366394	-1.84136997	0.026246673	0.38232189	0.23029169	0.60509474	-0.39202189	0.60509474	-0.39202189	0.60509474	-0.39202189	0.60509474	-0.39202189
7	GEN6	ribosomal protein L2	0.827069164	0.65448165	-0.027473877	0.94886622	0.329335956	0.17594067	0.38232189	0.23029169	0.60509474	-0.39202189	0.60509474	-0.39202189	0.60509474	-0.39202189	0.60509474	-0.39202189
8	GEN7	ribosomal protein L1	0.67047174	0.12024823	0.01567679	0.81923578	0.171448914	0.85434364	0.35438794	0.76933943	0.23751917	-0.20049362	0.610364759	0.137411356	0.344015146	-1.21077219	0.51910316	0.971
9	GEN8	pre-mRNA splicing f	0.13566892	0.46571735	0.1919418	-0.28462086	0.03917103	0.85434364	0.35438794	0.76933943	0.23751917	-0.20049362	0.610364759	0.137411356	0.344015146	-1.21077219	0.51910316	0.971
10	GEN9	adenylate cyclase-ss	0.955427439	0.15905233	0.486884701	0.894732158	0.278313322	0.32064331	1.242367192	0.13610383	-0.25270104	-1.572142128	0.64745219	0.62597029	0.731165248	0.507419831	0.507419831	0.507419831
11	GEN10	ubiquitin-cytosome	0.496623919	1.145092925	1.389411365	1.11221922	0.904514085	1.830851989	1.300208161	1.75964831	1.650218379	3.93001062	0.860537256	1.571607666	1.59137577	1.59137577	1.59137577	1.59137577
12	GEN11	ribosomal protein	1.62901312	1.56952225	1.61440062	1.04769667	1.18972576	1.826513342	2.86204539	3.80834841	1.72644495	1.467126542	4.01438343	1.53988929	1.60112446	2.24035413	2.24035413	2.24035413
13	GEN12	histone H3	0.81819267	0.00816563	0.20300664	0.972028137	0.14165224	0.344936315	0.331982334	0.296547036	0.029232629	-0.080597903	0.397301137	0.907613191	1.013344454	-1.791569489	0.594	0.594
14	GEN13	histone H4	0.81819267	0.00816563	0.20300664	0.972028137	0.14165224	0.344936315	0.331982334	0.296547036	0.029232629	-0.080597903	0.397301137	0.907613191	1.013344454	-1.791569489	0.594	0.594
15	GEN14	customer protein c	0.016544714	0.158711691	0.079273181	-0.10788321	-0.38654523	0.36801877	0.148977196	0.654014688	0.212544158	0.172729427	0.737894948	0.412388307	0.650344569	0.123458804	0.932	0.932
16	GEN15	acetyl-Coenzyme A c	-0.545417069	0.27417372	-0.53740729	-0.0438474	0.066790004	0.36801877	0.148977196	0.654014688	0.212544158	0.172729427	0.737894948	0.412388307	0.650344569	0.123458804	0.932	0.932
17	GEN16	Human AMP deaminase	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278	0.85349278
18	GEN17	Tub1 (actin-related)	0.36719771	0.67713819	0.172311684	0.65238472	1.44204495	0.933124439	0.96687688	0.87016159	1.50306326	0.87016159	1.50306326	0.87016159	1.50306326	0.87016159	1.50306326	0.87016159
19	GEN18	small nuclear ribon	-1.04639417	-1.26901633	-0.94482161	-0.80230955	-1.203462684	0.326529355	0.646107983	1.26816259	1.50306326	0.87016159	1.50306326	0.87016159	1.50306326	0.87016159	1.50306326	0.87016159
20	GEN19	Death associated pro	0.691696128	1.24978116	0.89397029	0.39388475	0.631166386	0.92301834	1.422128814	0.53283246	-1.01280362	1.16362936	0.363530715	0.719065919	0.719065919	0.719065919	0.719065919	0.719065919
21	GEN20	Human 30-kDa heat-s	0.912383029	0.546888085	0.20416442	0.714761891	0.55274622	0.10373958	0.656115842	0.52455944	0.29542329	0.68467769	0.54486919	0.68467769	0.54486919	0.68467769	0.54486919	0.68467769
22	GEN21	enoyl-Coenzyme A ly	0.911126435	1.869239026	1.571881414	1.11673158	1.223292122	1.77766654	1.78548027	1.463597073	0.85172503	1.601748404	3.82483732	1.78413042	1.79441181	0.57215128	0.632	0.632
23	GEN22	eukaryotic transl	0.73235455	0.117883041	0.53261051	0.57233495	0.998304629	1.43201199	0.67140393	1.95678932	0.77335765	1.46123905	0.96348622	0.96348622	0.96348622	0.96348622	0.96348622	0.96348622
24	GEN23	actin, beta	1.457695155	1.00201706	1.50345481	0.926574971	0.701016139	1.01046152	0.387428746	0.596611573	0.146288839	0.394675907	0.310001757	1.17386839	0.808132689	1.60003917	0.751	0.751
25	GEN24	calmodulin 2 (phosp	0.65365959	1.691251934	1.50338827	1.069636514	1.129561709	1.613284942	0.579562808	0.27946558	0.188094192	0.87272763	1.386694281	1.23866948	1.33045639	1.60003917	0.751	0.751
26	GEN25	phosphotrikinase	0.05365959	1.691251934	1.50338827	1.069636514	1.129561709	1.613284942	0.579562808	0.27946558	0.188094192	0.87272763	1.386694281	1.23866948	1.33045639	1.60003917	0.751	0.751
27	GEN26	phosphotrikinase	0.05365959	1.691251934	1.50338827	1.069636514	1.129561709	1.613284942	0.579562808	0.27946558	0.188094192	0.87272763	1.386694281	1.23866948	1.33045639	1.60003917	0.751	0.751
28	GEN27	enoyl-Coenzyme A ly	0.911126435	1.869239026	1.571881414	1.11673158	1.223292122	1.77766654	1.78548027	1.463597073	0.85172503	1.601748404	3.82483732	1.78413042	1.79441181	0.57215128	0.632	0.632
29	GEN28	eukaryotic transl	0.73235455	0.117883041	0.53261051	0.57233495	0.998304629	1.43201199	0.67140393	1.95678932	0.77335765	1.46123905	0.96348622	0.96348622	0.96348622	0.96348622	0.96348622	0.96348622
30	GEN29	actin, beta	1.457695155	1.00201706	1.50345481	0.926574971	0.701016139	1.01046152	0.387428746	0.596611573	0.146288839	0.394675907	0.310001757	1.17386839	0.808132689	1.60003917	0.751	0.751
31	GEN30	calmodulin 2 (phosp	0.65365959	1.691251934	1.50338827	1.069636514	1.129561709	1.613284942	0.579562808	0.27946558	0.188094192	0.87272763	1.386694281	1.23866948	1.33045639	1.60003917	0.751	0.751
32	GEN31	phosphotrikinase	0.05365959	1.691251934	1.50338827	1.069636514	1.129561709	1.613284942	0.579562808	0.27946558	0.188094192	0.87272763	1.386694281	1.23866948	1.33045639	1.60003917	0.751	0.751
33	GEN32	phosphotrikinase	0.05365959	1.691251934	1.50338827	1.069636514	1.129561709	1.613284942	0.579562808	0.27946558	0.188094192	0.87272763	1.386694281	1.23866948	1.33045639	1.60003917	0.751	0.751
34	GEN33	enoyl-Coenzyme A ly	0.911126435	1.869239026	1.571881414	1.11673158	1.223292122	1.77766654	1.78548027	1.463597073	0.85172503	1.601748404	3.82483732	1.78413042	1.79441181	0.57215128	0.632	0.632
35	GEN34	eukaryotic transl	0.73235455	0.117883041	0.53261051	0.57233495	0.998304629	1.43201199	0.67140393	1.95678932	0.77335765	1.46123905	0.96348622	0.96348622	0.96348622	0.96348622	0.96348622	0.96348622
36	GEN35	actin, beta	1.457695155	1.00201706	1.50345481	0.926574971	0.701016139	1.01046152	0.387428746	0.596611573	0.146288839	0.394675907	0.310001757	1.17386839	0.808132689	1.60003917	0.751	0.751
37	GEN36	calmodulin 2 (phosp	0.65365959	1.691251934	1.50338827	1.069636514	1.129561709	1.613284942	0.579562808	0.27946558	0.188094192	0.87272763	1.386694281	1.23866948	1.33045639	1.60003917	0.751	0.751
38	GEN37	phosphotrikinase	0.05365959	1.691251934	1.50338827	1.069636514	1.129561709	1.613284942	0.579562808	0.27946558	0.188094192	0.87272763	1.386694281	1.23866948	1.33045639	1.60003917	0.751	0.751
39	GEN38	phosphotrikinase	0.05365959	1.691251934	1.50338827	1.069636514	1.129561709	1.613284942	0.579562808	0.27946558	0.188094192	0.87272763	1.386694281	1.23866948	1.33045639	1.60003917	0.751	0.751
40	GEN39	enoyl-Coenzyme A ly	0.911126435	1.869239026	1.571881414	1.11673158	1.223292122	1.77766654	1.78548027	1.463597073	0.85172503	1.601748404	3.82483732	1.78413042	1.79441181	0.57215128	0.632	0.632
41	GEN40	eukaryotic transl	0.73235455	0.117883041	0.53261051	0.57233495	0.998304629	1.43201199	0.67140393	1.95678932	0.77335765	1.46123905	0.96348622	0.96348622	0.96348622	0.96348622	0.96348622	0.96348622
42	GEN41	actin, beta	1.457695155	1.00201706	1.50345481	0.926574971	0.701016139	1.01046152	0.387428746	0.596611573	0.146288839	0.394675907	0.310001757	1.17386839	0.808132689	1.60003917	0.751	0.751
43	GEN42	calmodulin 2 (phosp	0.65															

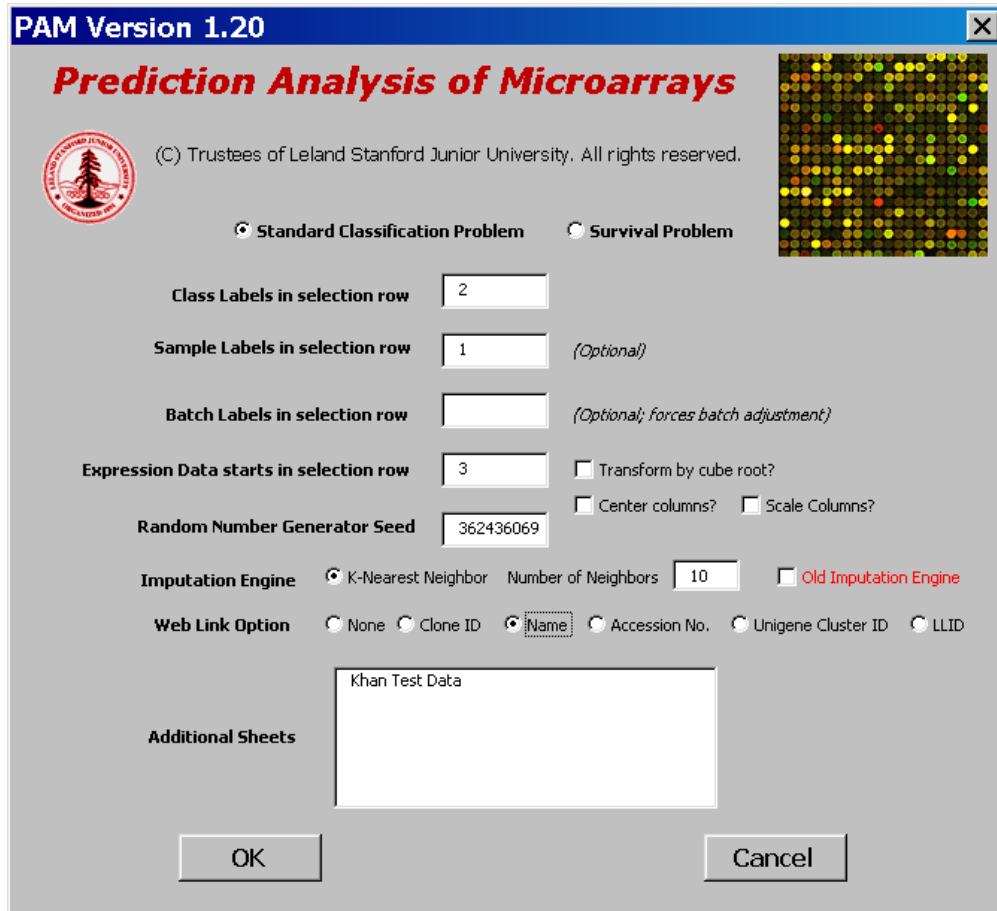


Figure 2: The PAM Dialog Box

**Class Labels** Enter the row number that contains the class labels. For the Khan data example we have provided, this is row 2.

**Sample Labels** Sample labels are optional, however, they are highly recommended. They allow one to identify sample predictions in plots and other output. Therefore, it is possible that they may be mandatory in the future.

The Khan data example has sample labels in row 1.

**Batch Labels** Batch labels are optional. If specified, they force a batch adjustment. This is useful when combining expression data from different experiments.

An example file named `khanbatch.xls` containing batch labels is distributed with the software. It has batch labels in row 1 class labels in row 3, sample labels in row 2.

**Imputation Engine** One can type in the number of neighbors required for the imputation. The default is 10.

**Web Link Option** PAM can hyperlink worksheet cells to source database. The default is to do no hyperlinking. (Some users of SAM worried that when they clicked things by accident Stanford was “snooping” their genes! So we chose this default). There are several choices for the criterion to search by, all self-explanatory.

**Additional Sheets** Additional sheets can be specified to overcome the Excel limitation of 256 columns. Control-click to make multiple selections.

Click the **OK** button to do the analysis. After some time a **PAM Controller** dialog shown in figure 3 pops up. When the **PAM Controller** is active, all input is directed to it (it is modal!). One can dismiss this dialog any time. When needed it can be resurrected by clicking on the **PAM Plot Control** menu button in Excel (right next to the **PAM Button**).

If you had any missing data in your spreadsheet, a new worksheet with the same name as the original sheet but suffixed by **(Imputed)** containing the imputed dataset is added to the workbook. This data can be used in subsequent analyses to save time. If there is no missing data, this worksheet is not added. Note that since PAM adds the **(Imputed)** suffix to an existing sheet name, it is important that the existing sheet name not be too long. Otherwise, an error will occur.

PAM adds three more worksheets to the workbook. The one called **PAM Plots** will contain plots that can be produced. The sheet named **PAM Output** will contain the list of significant genes when it is asked for.

The remaining sheet called **PAM Worksheet (Do Not Edit!)** is used for writing intermediate calculations and data used for plotting. The user can inspect this worksheet to see what the values that were computed by R. As the name of the sheet warns, it is not meant to be edited; the columns in this worksheet are linked to plots and changing anything could clobber a plot. One can use the data to generate other plots as needed. Each column in this worksheet has a header field

that describes the column. Moving the mouse over any header field will provide a more verbose description of what the column was used for.

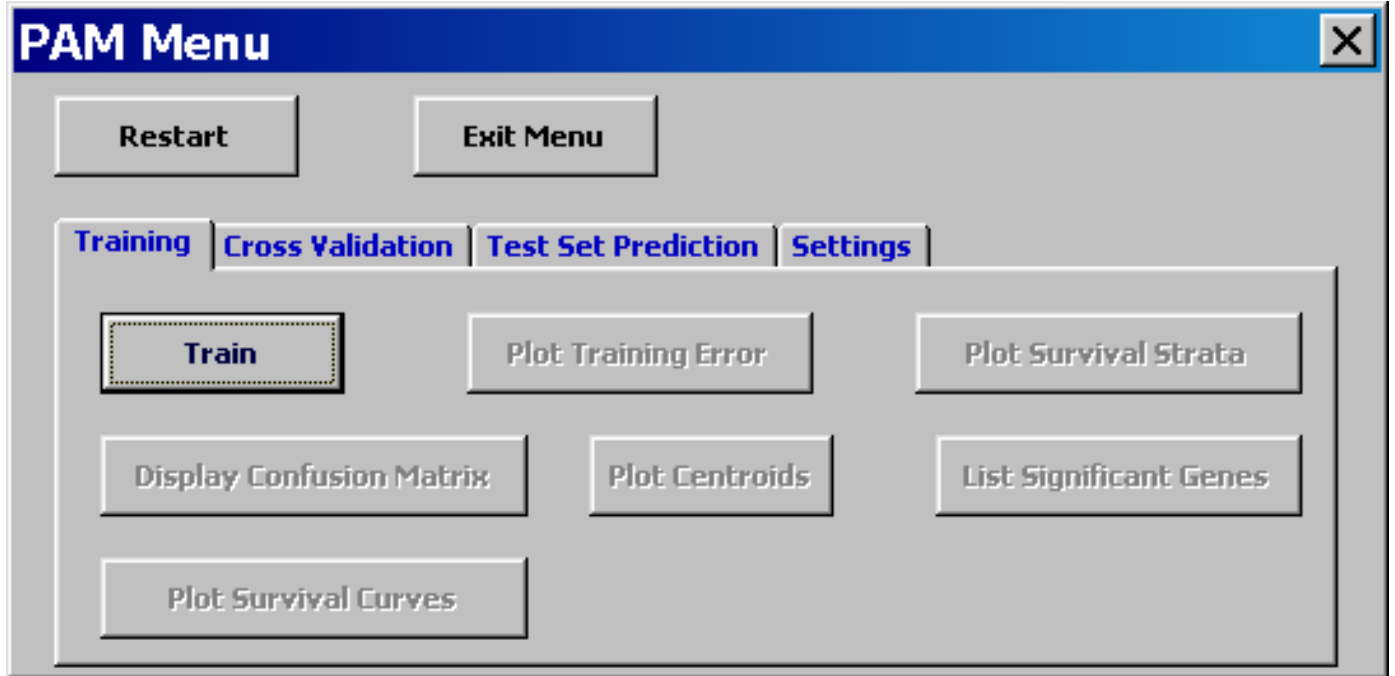


Figure 3: The PAM Controller

## 12.1 Training the Classifier

When the PAM Controller first appears, the only thing one can do is **Train** the classifier. Just click on the train button and the training should be done. Once the training is done, some other buttons in the controller are enabled and a text field appears for setting the **Threshold**. See figure 4.

Here is an explanation of the controls.

**Current Threshold** is a textbox where one enters a threshold value to use for shrinkage and for prediction. Until this is done, many of the other buttons are disabled.

The choice of the threshold is typically made after a judicious examination of training errors and the cross-validation results.

**Train** will train the classifier. This is always the first step.

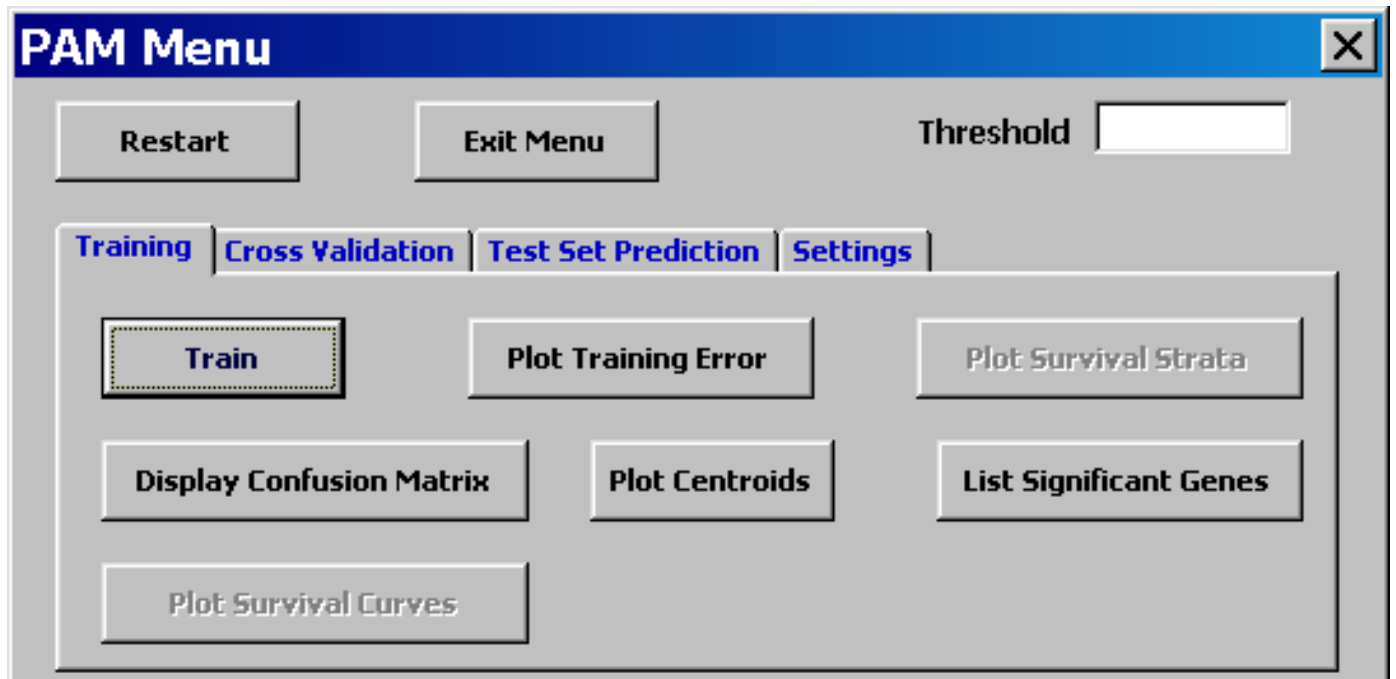


Figure 4: The PAM Controller after training

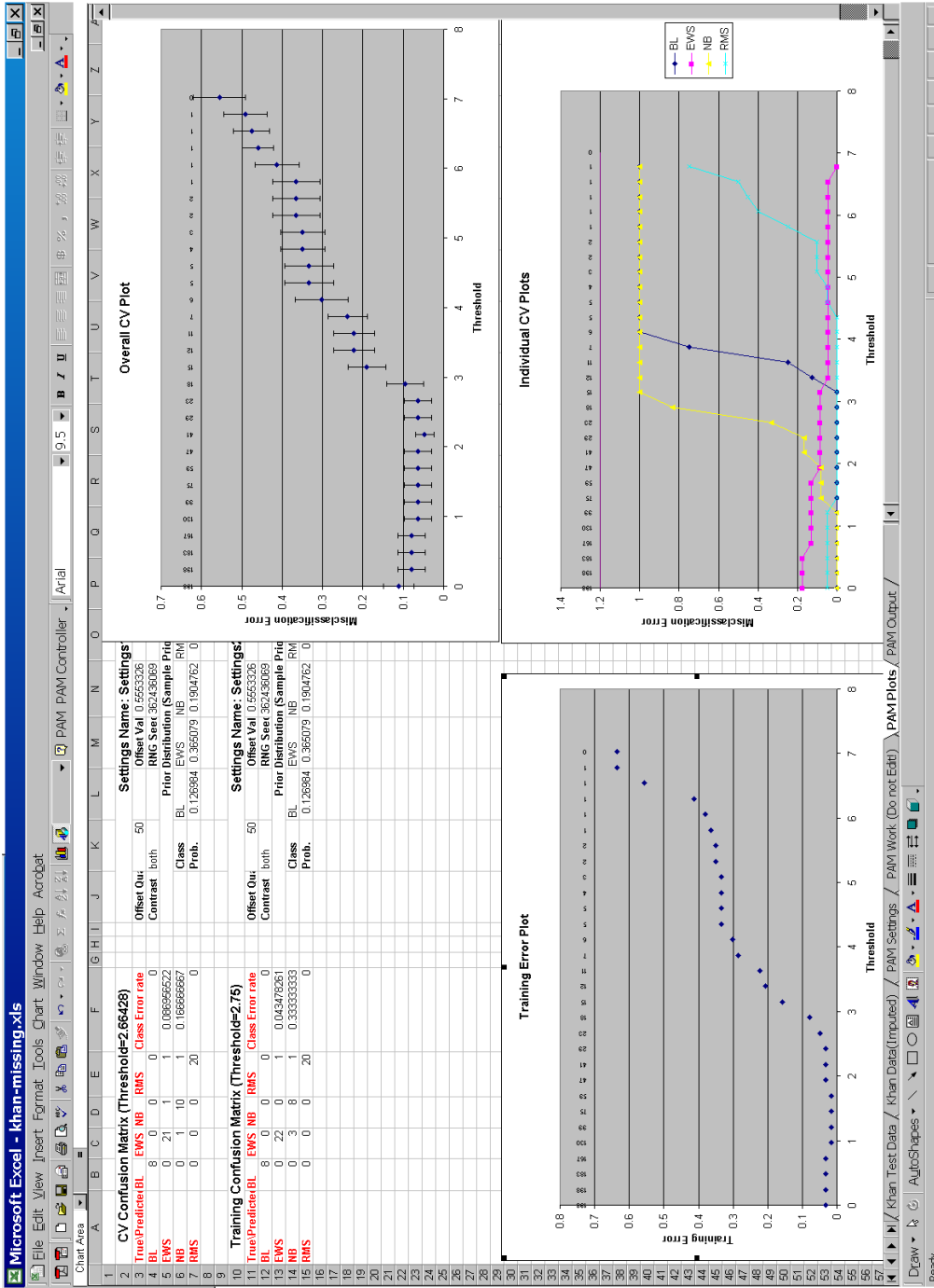


Figure 5: PAM Plots



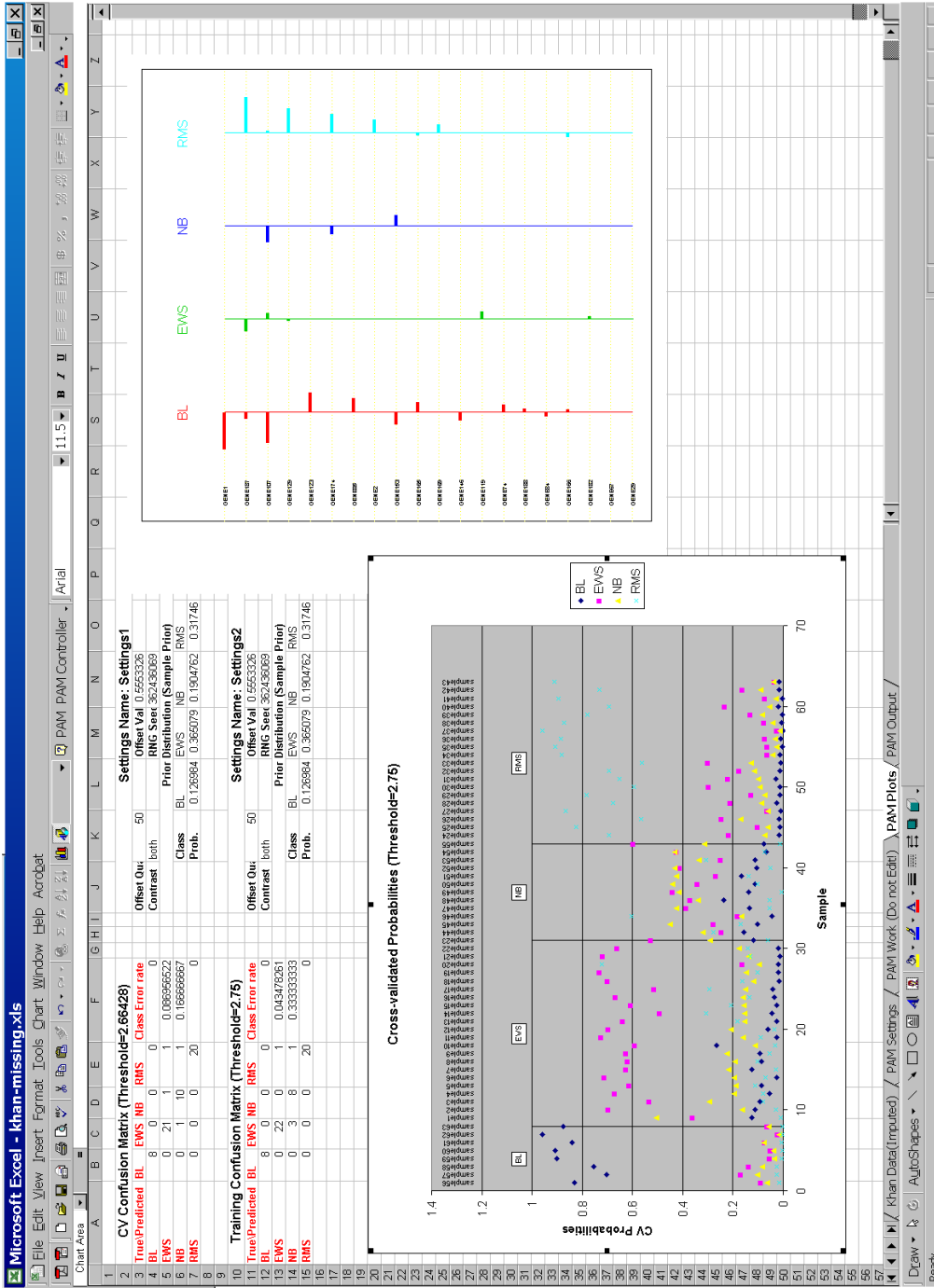


Figure 6: More PAM Plots

**Plot Training Error** will plot the training error and place the plot in the **PAM Plots** sheet.

**Confusion Matrix** button will output a training confusion matrix for a given threshold. The matrix is deposited in the **PAM Plots** sheet as we decided not to fatten the workbook with yet another additional worksheet. (This means that some of the plots could be obscuring the confusion matrix!) If a threshold has not been entered already, then you are asked to enter one.

**Cross Validate** will do a balanced 10-fold cross validation to enable one to choose a threshold that minimizes classification errors.

**Plot CV Curves** will plot the misclassification errors obtained by cross-validation for various values of the threshold.

**Plot CV Probabilities** will plot classification probabilities for a specified threshold.

**Plot Centroids** will plot the shrunken centroids for a specified threshold.

**List Genes** will list the significant genes in the format described in section 12.4.

**Predict Test Set** can be used to predict a test set. More on this in section 12.2.

**Plot Test Error** can be used to plot the prediction errors for a number of values of the threshold.

**Plot Test Probabilities** will plot the class probabilities for each sample in the test set for a specified threshold.

**Show Prediction** will create a worksheet with the prediction confusion matrix, if computable, and a list of actual and predicted class labels along with the prediction probabilities for each class.

The last four buttons are used in prediction and are further described in section 12.2.

We emphasize that almost all of the button controls may appear to be enabled or disabled depending on the stage of the analysis. For example, the button **Plot CV Probabilities** remains disabled until one has performed the cross validation and entered a valid threshold.

Please note that one can produce a number of plots and there is the very real problem of screen real estate. Often one plot can get obscured by another. Furthermore, some generated plots usually are more comprehensible when resized.

All the interactive capabilities that Excel provides for plots are available to the user. Of course, these can only be exploited when one exits the **PAM Controller** menu because all events are directed to the controller when it is active.

Figures 5 and 6 show some of the plots produced by PAM for the khan data example.

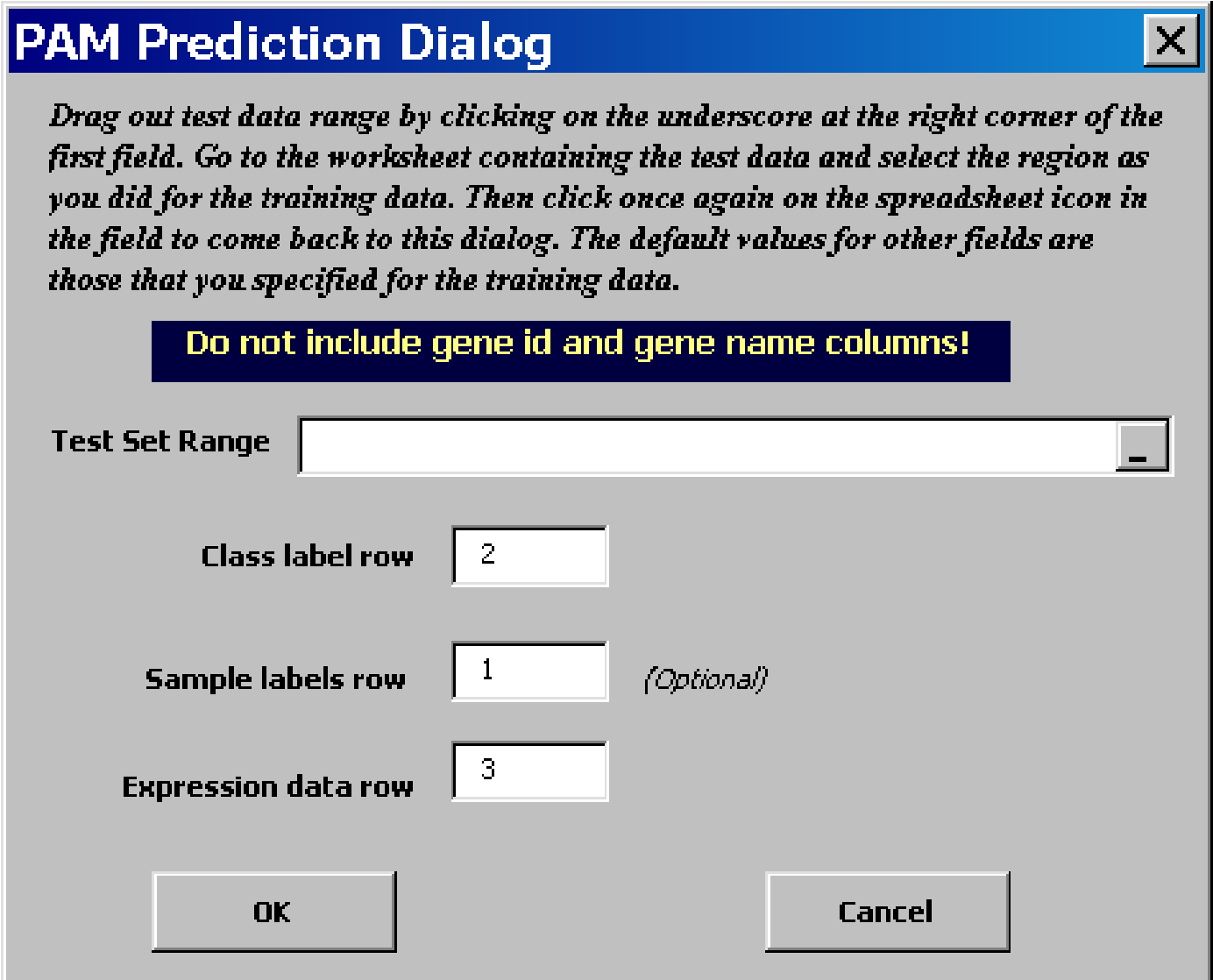


Figure 7: The PAM Prediction Dialog



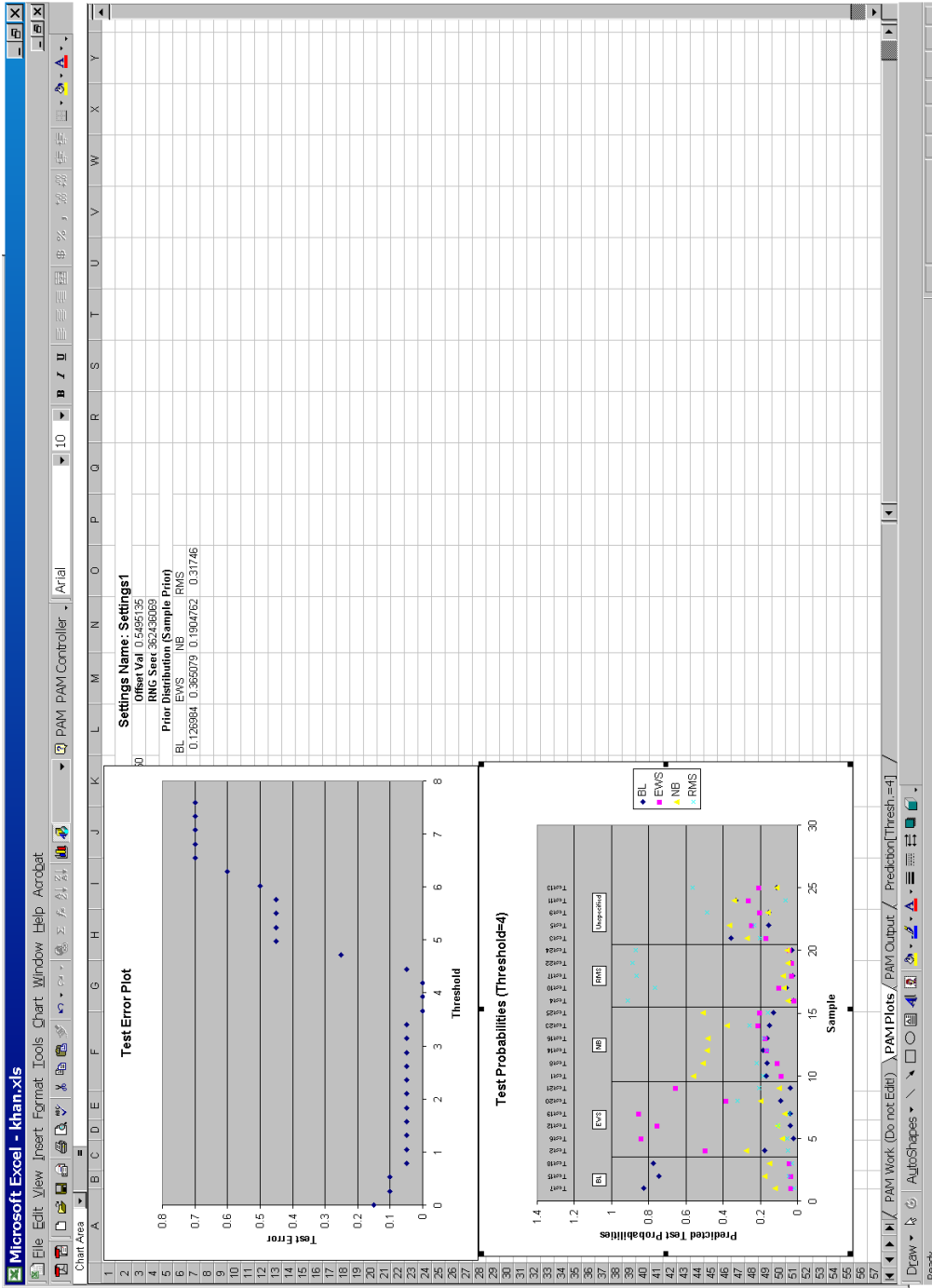


Figure 9: The test error and probability plots for Khan data

## 12.2 Test Set Prediction

Before venturing into prediction, we note that the test data set for prediction is expected to have the same number of genes as the training data set. All or some class labels cells for the test set can be empty but *the class labels row must exist!* Sample labels are once again optional but recommended so that one can identify the samples in prediction output. If specified, none of them should be empty.

To predict class labels for a new test data set, one first clicks on the **Predict Test Set** button. A new dialog—see 7—pops up. To select a test sample, one clicks on the underscore to the right of the **Test Set Range** field. This has the effect of replacing all other dialogs *temporarily* with a new one with a spreadsheet icon at the right of the field. You can now navigate through various sheets and select an area for prediction. If you were following along using the example file **khan-missing.xls**, there is a test data set available in the sheet names **Khan Test Data**. Click on the cell containing `Test1`, scroll down to the right corner and then shift click on the right corner of the data set. It is important that the selected area not contain the gene names or ids. After selecting the area, click on the spreadsheet icon and you will get back to the previous dialog.

The rows containing class labels, sample labels and start of expression data are by default assumed to be what one specified for the training set. They can be edited or changed depending on your test data set.

Clicking on the **OK** button will prepare PAM for predicting the test set. If all goes well, the buttons in the last row are all enabled.

At this point, one can plot the test errors and, if the threshold is set, plot the predicted probabilities for the test set. Clicking on the **Show Prediction** button produces an output sheet with a detailed summary. If the test data set contained all class labels, a confusion matrix is printed at the top of the sheet. If the test data contains a few missing labels, they are highlighted in red color. See 8.

## 12.3 Using data in Multiple Sheets

The maximum number of columns one can have in an Excel worksheet is 256 columns (A through IV). If you have more than 256 samples, you can arrange the data in multiple sheets before invoking PAM.

For example, consider the situation where you have 5000 genes and 300 samples. Assume you don't have sample labels or batch labels. Per the data format required by PAM, this means that the data set would contain  $300 + 2 = 302$  columns and 5001 rows. The extra two columns contain the gene name and identifier and the top row contains the class labels.

One possibility is to put the first 256 columns in one sheet and the remaining 46 in another sheet. Or a 100, 100, 102 split over three, not necessarily contiguous worksheets, is also possible—it is your call. Then, highlight the regions in each sheet as usual by clicking on the top-left corner of the rectangle and shift-clicking on the right-bottom corner. Then switch back to the sheet that

contains the gene names and ids.

*PAM must be invoked from the sheet that contains the gene names and ids.* Failure to do so will result in all kinds of hell breaking loose.

The PAM dialog will offer you the option of choosing the additional sheets. Control-click on the sheets that contain the additional data. Proceed as usual after this point.

If any of the  $n$  input sheets contains missing data, please note that PAM will add  $n$  sheets with the prefix (**Imputed**).

## 12.4 Format of the Gene Listing

The gene listing has the following format:

**ID** The gene ID specified in the second column selected data rectangle. This is for the user's reference and is linked to the SOURCE website if so desired.

**Gene Name** The gene name specified in the first column selected data rectangle. This is for the user's reference.

These columns are followed by the centroid scores for the various classes specified in the data set.

## 13 Running PAM: Survival analysis and regression

We give a briefer description of the survival and regression problems, highlighting the places where they differ from the classification setting.

### 13.1 Basic idea of supervised principal components

Supervised principal components is a generalization of principal components regression. The first (or first few) principal components are the linear combinations of the features that capture the directions of largest variation in a dataset. But these directions may or may not be related to an outcome variable of interest. To find linear combinations that are related to an outcome variable, we compute univariate scores for each gene and then retain only those features whose score exceeds a threshold. A principal components analysis is carried out using only the data from these selected features.

Finally, these “supervised principal components” are used in a regression model to predict the outcome. To summarize, the steps are:

1. Compute (univariate) standard regression coefficients for each feature

2. Form a reduced data matrix consisting of only those features whose univariate coefficient exceeds a threshold  $\theta$  in absolute value ( $\theta$  is estimated by cross-validation)
3. Compute the first (or first few) principal components of the reduced data matrix
4. Use these principal component(s) in a regression model to predict the outcome

This idea can be used in standard regression problems with a quantitative outcome, and also in generalized regression problems such as survival analysis. In the latter problem, the regression coefficients in step (1) are obtained from a proportional hazards model. The PAM handles these two cases: standard regression and survival data.

There is one more important point: the features (e.g. genes) which are important in the prediction are not necessarily the ones that passed the screen in step 2. There are other features that may have as high a correlation with the supervised PC predictor. So we compute an importance score for each feature equal to its correlation with the supervised PC predictor. A reduced predictor is formed by soft-thresholding the importance scores, and using these shrunken scores as weights. The soft-thresholding sets the weight of some features to zero, hence throwing them out of the model. The amount of shrinkage is determined by cross-validation. The reduced predictor often performs as well or better than the supervised PC predictor, and is more interpretable.

Thus the user should think of the analysis in two distinct stages. In the first stage, a `threshold` is chosen, yielding a supervised principal component predictor which is fit to test data. Note that this predictor has non-zero importance scores (loadings) for all genes. In the second stage, a `shrinkage` parameter is chosen, to reduce the number of genes with non-zero loadings. This reduced predictor is the fit to test data. Hopefully, it will perform as well as the full predictor and will use far fewer genes.

This two-stage view provides a useful way to consider the results from a microarray analysis. First it looks for a global predictor that captures the overall prediction information in the genes. Then it looks for a subset of genes that show the strongest correlation with this overall predictor. This process separates out the discrete feature selection step, and acknowledges that there may be many different gene sets that produce similar models over the data.

To summarize:

*Steps in a typical supervised PC analysis*

1. *Training*: compute Cox survival scores for each gene
2. *Cross-validation*: use cross-validation to estimate best score threshold
3. *Testing*: use this threshold to construct the supervised PC predictor and apply it to predict test data. Examine survival curves for the discretized predictor and the output from multivariate model.
4. As a second stage, seek a reduced model by soft-thresholding the loadings. Form the resulting “reduced” supervised PC predictor and see how well it predicts.



## 13.2 Opening screen

For survival analysis problems, you are asked for **Survival Time** and **Censoring Status** instead of **Class Labels**. For regression analysis problems, you will be asked for the **Outcome variable**. Sample labels are not required in general, but are required if a comparison to competing predictors is desired (details below)

## 13.3 Training, Cross-validation and Test Set Prediction

Here is an explanation of the controls.

**Current Threshold** is a textbox where one enters a threshold value to use for shrinkage and for prediction. Until this is done, many of the other buttons are disabled. For survival or regression, this is the threshold for the univariate score that correlates each gene with the outcome. Specifically- the Score statistics from the proportional hazards model for survival data, and the standardized regression coefficient for regression problems.

The choice of the threshold is typically made after a judicious examination of training errors and the cross-validation results.

**Train** will train the classifier. This is always the first step.

**Plot Training Error** will plot the log-likelihood ratio statistic from the proportional hazards or linear regression supervised principal components fit, and place the plot in the **PAM Plots** sheet. Note that high values are better, in contrast to misclassification error.

**List Gene scores** will list the significant genes. The format is geneid, genename, importance score- the loading of each gene on the supervised principal component, and the raw score- the univariate score of each gene with the outcome.

**Princ comp number of gene scores** is where the user must choose which of the first 3 supervised principal components (1,2 or 3) is to be used to compute gene important scores. The scores are the loadings (inner products) of each gene with the supervised principal component.

**Decorrelate with competing** allows the user to first decorrelate the expression data from competing predictors (such as clinical stage, grade etc), before training the model. This encourages the procedure to find a gene expression-based predictor that is independent of the competing predictors. The training data for competing predictors are entered in a separate spreadsheet, with the format shown on the “Get competing predictors” Dialog Box. An example is given in the survival workbook. Note that predictor names must contain only letters and digits, with no spaces. In the Test set prediction menu, Decorrelate with competing decorrelates the test set expression data from the test set competing-predictor data.

**Fit with competing** allows the user to compare the gene expression-based predictor to competing predictors (such as clinical stage, grade etc) on test data. The test data for the competing predictors are entered into a separate spreadsheet, with the format shown in the “Input Competing Predictors” Dialog Box. An example is given in the `survival` workbook. Note that predictor names must contain only letters and digits, with no spaces. The user can choose `Fit with competing`, whether or not they have decorrelated the expression data before training. If they have decorrelated the training data, they will be reminded to also decorrelate the test data, before proceeding with test set predictions.

**Cross Validate** will do two balanced 3-fold cross validations to enable one to choose a threshold that maximizes the log-likelihood ratio statistic. The two cross-validations are then combined.

**Plot CV Curves** will plot the log-likelihood ratio statistic, obtained by cross-validation for various values of the threshold.

**Seek reduced model** applies soft-thresholding of various `shrinkage` amounts to the gene loadings, to yield predictors that use a subset of the genes.

**Plot reduced model** will plot the log-likelihood ratio statistic, obtained by cross-validation for various values of the shrinkage parameter. The `threshold` is fixed at the value entered in the threshold box.

**Predict Test Set** can be used to predict a test set. More on this in section 12.2.

**Plot Test Error** can be used to plot the prediction errors (log-likelihood ratio statistic) for a number of values of the threshold.

**Response Prediction plot** creates a plot and summary for the predictive model. For survival analysis, these are Kaplan-Meier curves for the categorized predictor, discretized at the appropriate percentiles of the training data. For regression, this is a plot of predicted vs actual values. Note that for survival test data, the cutpoints from the training set are used in the test data. When the number of principal components is bigger than 1, the results for individual components are shown, as well as the `combined predictor` that uses a weighted average of all components (weights determined by the predictive model fit to the training data)

**Output Prediction info** Output the predicted value for each test sample, and summaries of the proportional hazards or regression models that have been fit to test data.

**Shrinkage value** This Box allows the user to define a shrinkage value for soft-thresholding the gene loadings. This produces a model that uses fewer genes. This reduced model is then used if any of the `Plot Test Error`, `Response Prediction plot` or `Output`

Prediction info buttons are pressed. The results, including the number of genes in the reduced model, are listed in the prediction worksheet.

**Number of Survival Classes** setting indicates how many survival classes (2, 3, or 4) are to be used in discretizing the supervised PC predictor for the Response Prediction plot.

**Number of Principal components** setting indicates how many supervised principal components (1, 2 or 3) are to be used in the predictive models.

## 13.4 Learning competing predictors from expression data

Since traditional clinical predictors like stage, grade and blood pressure are often powerful and useful, it is of interest to ask whether they can be “learned” from expression data. That is whether a gene expression-based model can accurately predict them. The PAM package has all of the tools needed to do this. For categorical competing predictor (eg. stage) , one would run PAM in classification mode, with the competing predictor as the class label. For a quantitative competing predictor (like blood pressure), one would run PAM in regression mode. Finally the predicted stage and blood pressure can be used as competing predictors in PAM, in for example, a model to predict patient survival. We have only briefly experimented with this idea, but it does seem to have potential.

## 14 Details of the R Excel Connection

A forthcoming document will provide more details on how to build applications using the R DCOM server.

## 15 Frequently Asked Questions

### 15.1 General Questions

1. How is PAM licensed?

PAM is distributed under the GNU Public License 2.0.

2. Is there a version of PAM that works on Macintosh computers?

We have not tried it on Macintoshes. We know that SAM works on a Windows emulator on Macs such as Virtual PC from Connectix Corporation. So the same is probably true for PAM. Let us know if it does and we’ll update this answer.

## 15.2 PAM Registration Questions

1. I registered for PAM and I have still not received an email confirming my registration.

This is most likely due to your email server being down. Hundreds of requests have been successfully sent out to people. Our registration server tries every hour to re-mail the pending requests.

If you do not receive your registration user-id and password within the day, you may always register again and use another email address that works.

## 15.3 Installation, Uninstallation Questions

1. How do I uninstall PAM?

With the current and future versions, the uninstallation process is very simple. Click on **Start**, then **Programs** and choose **Uninstall PAM for Excel** from the menu.

With versions of PAM prior to 1.20, the process is more involved. One pretty much reverses the steps in the install process. However, please make sure you do it in the following order.

- (a) First you must unlink PAM from the list of **Addins** loaded into Excel. The list of addins is available by choosing the **Addins** item from the **Tools** menu.
  - (b) PAM can be uninstalled via the **Control Panel**. Double Click **Add/Remove Programs** and double click on **Prediction Analysis of Microarrays**.
  - (c) To be complete, you should also uninstall the *PAM for R* package by typing in an R command window `remove.packages("pamr")`.
2. How do I install a newly released version of PAM? Do I just install it on top of the old version?  

Installing new software on top of old versions is a good way to hose your Windows machine. If you want to preserve the little sanity that Windows has, you must first uninstall the old version and then install the new version.
  3. When I install PAM, I get an error that a library was not registered. However, at the end, the program says that the installation was successful. Does this mean that PAM is installed correctly?  

No! Anytime an error occurs, it means that PAM is not installed properly. The problem must be fixed before you can rely on PAM working for you. This often happens when the prerequisites are not met. See section 5.
  4. I am using office 97. Where can I download the Service packs for it?

The last time we looked, it was at the following URL: <http://office.microsoft.com/downloads/9798/sr2off97detail.aspx>.

If you don't find it there, search for the words **office97 service release** at the web-site <http://office.microsoft.com>. Beware these things keep being moved around!

## 15.4 PAM Usage Questions

1. PAM generates an error saying that it could not start the R server. What can I do?

For, versions prior to 1.20:

- Please make sure you have installed R!
- Please make sure you have installed the R package `pamr`!

Remember that *PAM for Excel* is a front-end to *PAM for R* and so the latter is needed.

For versions 1.20 and higher, this is probably a bug.

2. PAM generates an error when I run it on my dataset. What should I do?

The Excel R DCOM connection is quite brittle and is easily broken. In general, DCOM errors can be hard to debug.

- Please make sure that your data is formatted exactly as described in section 10. Particular attention needs to be paid to the class labels, sample labels or batch labels as described in section 12. It is easy to misspecify these rows.

For prediction, it is imperative that you not include the gene id and gene names columns—they are implicit.

- Please make sure that you have chosen your data area appropriately as discussed in 12. It is easy to highlight the wrong area or accidentally highlight unwanted cells.
- Have you installed several versions of R? We've seen some problems in such situations.

3. My plots are no longer interactive. Help!

When the PAM controller is active, all input is directed to it. To get interaction with plots, exit the PAM Controller and click on the plot you want to interact with.

4. PAM generates an error saying that an illegal name was specified for a sheet.

Does your workbook have long names for worksheets? You might want to change such names to shorter ones. There is a limit on the length of names for worksheets. The current version is not very intelligent about naming sheets and it will be improved in subsequent versions.

5. PAM generated an error. Now when I get back into Excel, I have several PAM buttons. Help!

This is an annoying bug that we doubt we can ever fix satisfactorily.

For versions below 1.20, fire Excel up again uncheck the PAM Excel addin. Quit excel. Then bring up Excel again and check the addin. That should take care of the problem.

For version 1.20 or higher, just fire up Excel, exit and fire it up again. The multiple buttons will go away.

6. PAM gave an error message:

```
The R server said this:  
There is no package called 'pamr'.
```

This can only happen with versions below 1.20. You did not install the *PAM for R* package. We suggest upgrading to version 1.20 as outlined in section 6.2.

7. How come probabilities don't add to 1 in the PAM Worksheet?

They do. Excel has the bad habit of rounding numbers for presentation purposes. Thus 0.91 can appear as 1 if there isn't enough space to display two digits. Select the cell containing the value and you can see the actual number.

8. I get pound signs (#) where I expect numbers. Why?

Excel will display pound signs in cells where it cannot fit the actual values. Just enlarge the cell width and you should see the values.

9. How do I interpret the probability plots?

Suppose you have three classes A, B and C. For each sample, three numbers plotted: the prediction probability for class A, that for class B and for class C. Thus for a single sample, three  $y$  values will be shown in different colors. The predicted class is the class corresponding to the highest probability.

We also order the probability plot so that there would be three panels for the three-class example. The panels are usually labelled.

When doing prediction, an extra panel is added, containing those samples for which no class labels were specified. This panel is labelled "Unspecified."

10. How large a dataset can PAM handle?

There is really no hard limit *per se* in PAM. Excel itself has some limit on the number of rows and columns it can handle. There are additional overheads involved in marshalling the data between Excel and the core of PAM. Therefore, the practical limit is lower. In general, the more memory you have, the larger problems you can handle.

11. How do I know what PAM is doing? Why don't I see a Progress Bar?

At this point, there is no mechanism to show progress when one is deep in R computation. This should be possible in the near future though, given some new developments. For now, until the computation ends, successfully or in error, one has to simply wait.

12. This document does not answer my questions. Where should I look?

As we get asked new questions, we update this list of frequently asked questions with answers. Please visit the url <http://www-stat.stanford.edu/~tibs/PAM> where you may find further information.

13. Where is the PAM manual?

It should be located in `C:\Program Files\PAMVB\doc` in the default installation. If you used a different directory, then it should be in the analogous place.

In the worst case, search for the file **pam.pdf**.

14. Where are the examples?

They should be located in the `C:\Program Files\PAMVB\Examples` in the default installation. If you used a different directory, then it should be in the analogous place.

In the worst case, search for the file **khan.xls**.

15. What does the gene hyperlink lookup do? Does it mean that my identified genes are snooped by Stanford?

The web lookup facility is provided merely a convenience. By default, it is off. One doesn't have to use it.

Please remember that all websites have logs and surely your query gets recorded somewhere. But as to what happens to it, we cannot answer as we have really no affiliation with that site.

So the bottom line is that if you are really concerned, you should just refrain from using that feature.

16. Where can I go for help if I just cannot get PAM to work?

We are very interested in making PAM work for all users. However, before reporting problems or bugs, we'd really like you to make sure that the problem is really with PAM. The following checklist should help.

- Please make sure you have installed all the prerequisites. See section 5.
- If the problem is with PAM usage, please make sure that you have formatted your data exactly as mentioned in the PAM manual.

- If you are having problem on a particular type of data, please make sure that you have formatted the response labels appropriately and have chosen the correct applicable data type.

If you still cannot get PAM to work, send email to `pam-bug@stat.stanford.edu` with complete details including

- (a) The error message
- (b) The system you are using (Windows 95, Windows 98, Windows ME, Windows NT or Windows 2000)
- (c) Whether you have installed all the prerequisites mentioned in the PAM manual.
- (d) The dataset you used that generated the error.

## 16 Technical Details- Classification

### 16.1 Details of the centroid shrinkage

Let  $x_{ij}$  be the expression for genes  $i = 1, 2, \dots, p$  and samples  $j = 1, 2, \dots, n$ . We have classes  $1, 2, \dots, K$ , and let  $C_k$  be indices of the  $n_k$  samples in class  $k$ . The  $i$ th component of the centroid for class  $k$  is  $\bar{x}_{ik} = \sum_{j \in C_k} x_{ij}/n_k$ , the mean expression value in class  $k$  for gene  $i$ ; the  $i$ th component of the overall centroid is  $\bar{x}_i = \sum_{j=1}^n x_{ij}/n$ .

In words, we shrink the class centroids towards the overall centroids. However, we first normalize by the within class-standard deviation for each gene. Let

$$d_{ik} = \frac{\bar{x}_{ik} - \bar{x}_i}{m_k \cdot s_i}, \quad (16.1)$$

where  $s_i$  is the pooled within-class standard deviation for gene  $i$ :

$$s_i^2 = \frac{1}{n - K} \sum_k \sum_{i \in C_k} (x_{ij} - \bar{x}_{ik})^2. \quad (16.2)$$

and  $m_k = \sqrt{1/n_k - 1/n}$  makes the denominator equal to the standard error of the numerator in  $d_{ik}$ . Thus  $d_{ik}$  is a t-statistic for gene  $i$ , comparing class  $k$  to the other classes.

One important detail—in the denominator of the statistics  $d_{ik}$  in (16.1) we add (the same) positive constant  $s_0$  to each of the  $s_i$  values. This guards against the possibility of large  $d_{ik}$  values arising by chance, from genes at very low expression levels. We set  $s_0$  equal to the median value  $s_i$  over the set of genes. A similar strategy was used in the SAM methodology of Tusher, Tibshirani, and Chu (2001), although instead of the median it chooses the quantile of the  $s_i$  values in an



adaptive fashion. For this reason in the two-class case, the genes scores in SAM and PAM can differ a little.

We can write

$$\bar{x}_{ik} = \bar{x}_i + m_k s_i d_{ik} \quad (16.3)$$

Our proposal shrinks each  $d_{ik}$  towards zero, giving  $d'_{ik}$  and new shrunken centroids or prototypes

$$\bar{x}'_{ik} = \bar{x}_i + m_k s_i d'_{ik} \quad (16.4)$$

The shrinkage we use is called *soft-thresholding*: each  $d_{ik}$  is reduced by an amount  $\Delta$  in absolute value, and is set to zero if its absolute value is less than zero. Algebraically, this is expressed as

$$d'_{ik} = \text{sign}(d_{ik})(|d_{ik}| - \Delta)_+ \quad (16.5)$$

where  $+$  means *positive part* ( $t_+ = t$  if  $t > 0$ , and zero otherwise). Since many of the  $\bar{x}_{ik}$  will be noisy and close to the overall mean  $\bar{x}_i$ , soft-thresholding produces “better” (more reliable) estimates of the true means (Donoho and Johnstone, 1994a; Tibshirani, 1996).

It has the nice property that many of the components (genes) are eliminated as far as class prediction is concerned, if the shrinkage parameter  $\Delta$  is large enough. Specifically if for a gene  $i$ ,  $d_{ik}$  is shrunken to zero for all classes  $k$ , then the centroid for gene  $i$  is  $\bar{x}_i$ , the same for all classes. Thus gene  $i$  does not contribute to the nearest centroid computation. We choose  $\Delta$  by cross-validation.

Note that the standardization by  $s_i$  above has the effect of giving higher weight to genes whose expression is stable within samples of the same class. This same standardization is inherent in other common statistical methods, such as linear discriminant analysis (see section 16.4).

## 16.2 Choosing the amount of shrinkage

The `Plotcv` button plots cross-validation error as the shrinkage parameter  $\Delta$  is varied. Shown along the top of the plot is the number of genes retained (for the training data) at that particular threshold.

We use balanced 10-fold cross-validation, ensuring that the classes were distributed evenly among the 10 folds.

From this we choose a threshold value to select a list of genes. The `Plot centroids` button plots the shrunken differences  $d_{ik}$  for only the genes having at least one non-zero difference.

## 16.3 Class probabilities and discriminant functions

We classify test samples to the closest shrunken centroid, again standardizing by  $s_i$ . We also make a correction for the relative abundance of members of each class.

Suppose we have a test sample (vector) with expression levels  $x^* = (x_1^*, x_2^*, \dots, x_p^*)$ . We define the *discriminant score* for class  $k$

$$\delta_k(x^*) = \sum_{i=1}^p \frac{(x_i^* - \bar{x}'_{ik})^2}{s_i^2} - 2 \log \pi_k \quad (16.6)$$

The first part of (16.6) is simply the standardized squared distance of  $x^*$  to the  $k$ th shrunken centroid. The second part is a correction based on the class *prior probability*  $\pi_k$ , where  $\sum_{k=1}^K \pi_k = 1$ . This prior gives the overall proportion of class  $k$  in the population. The classification rule is then

$$C(x^*) = \ell \text{ if } \delta_\ell(x^*) = \min_k \delta_k(x^*) \quad (16.7)$$

If the smallest distances are close and hence ambiguous, the prior correction gives a preference for larger classes, since they potentially account for more errors. We usually estimate the  $\pi_k$  by the *sample priors*  $\hat{\pi}_k = n_k/n$ . If the sample prior is not representative of the population, then more realistic priors can be used instead, or even uniform priors  $\pi_k = 1/K$ .

We can use the discriminant scores to construct estimates of the class probabilities, by analogy to Gaussian linear discriminant analysis:

$$\hat{p}_k(x^*) = \frac{e^{-\frac{1}{2}\delta_k(x^*)}}{\sum_{\ell=1}^K e^{-\frac{1}{2}\delta_\ell(x^*)}} \quad (16.8)$$

The **Plot CV Probs** button displays these probabilities, for each sample, for each class. At each ascissa value, the class probabilities for a given sample are plotted. The **Predict Test Set** button computes these probabilities for a test set.

## 16.4 Relationship to other approaches

The discriminant scores (16.6) are similar to those used in linear discriminant analysis (LDA), which arise from using the *Mahalanobis* metric in computing distance to centroids:

$$\delta_k^{LDA}(x^*) = (x^* - \bar{x}_k)^T W^{-1} (x^* - \bar{x}_k) - 2 \log \pi_k. \quad (16.9)$$

Here we are using a vector notation, and  $W$  is the pooled, within-class covariance matrix. With thousands of genes and tens of samples ( $p \gg n$ ),  $W$  is huge and any sample estimate will be singular (and hence its inverse is undefined) Our scores can be seen to be a heavily restricted form of LDA, necessary to cope with the large number of variables (genes). The differences are that

- we assume a diagonal within-class covariance matrix for  $W$ ; without this LDA would be ill-conditioned and fail.
- we use shrunken centroids rather than centroids as a prototype for each class.

- as the shrinkage parameter  $\Delta$  increases, an increasing number of genes will have *all* their  $d'_{ik} = 0$ ,  $k = 1, \dots, K$  due to the soft-thresholding in (16.5). Such genes contribute no discriminatory information in (16.6), and in fact cancel in (16.8)

Both our scores (16.6) and (16.9) are effectively *linear* in the  $x_i^*$ . If we expand the square in (16.6), discard the terms involving  $x_i^{*2}$  (since they are independent of the class index  $k$  and hence do not contribute towards class discrimination), and multiply by  $-2$ , we get

$$\tilde{\delta}_k(x^*) = \sum_{i=1}^p \frac{x_i^* \bar{x}'_{ik}}{s_i^2} - \frac{1}{2} \sum_{i=1}^p \frac{\bar{x}'_{ik}{}^2}{s_i^2} + \log \pi_k, \quad (16.10)$$

which is linear in the  $x_i^*$ . Because of the sign change, our rule classifies to the largest  $\tilde{\delta}_k(x^*)$ . Likewise the LDA discriminant scores have the equivalent linear form

$$\tilde{\delta}_k^{LDA}(x^*) = x^{*T} W^{-1} \bar{x}_k - \frac{1}{2} \bar{x}_k{}^T W^{-1} \bar{x}_k + \log \pi_k \quad (16.11)$$

Both Golub et al. (1999) and Hedenfalk et al. (2001) present linear scoring procedures for the two-class problem, which are very similar to each other, and also similar to our procedure and hence LDA in the two-class case. Both methods essentially work by

1. selecting a set of genes based on a two-sample t-statistic, and then
2. forming a linear classifier in the selected  $x_i^*$  using these t-statistics as weights.

We describe each in turn, using our own re-expression and terminology, but remaining faithful to the original in each case.

Hedenfalk et al. (2001) use a standard t-test at level  $\alpha$  to select differentially expressed genes in two breast cancer types. They then construct a “compound covariate” class predictor, which they threshold to classify a sample into one of the two classes. Their compound covariate is

$$h(x^*) = \sum_{i \in S(\alpha)} t_i x_i^* \quad (16.12)$$

where  $x^*$  is the gene expression vector for the sample to be classified,  $t_j$  is the t-statistic for comparing the two classes in the training data at level- $\alpha$ , and  $S(\alpha)$  is the subset of genes with a significant t-statistic at level  $\alpha$ . They use as a threshold the same compound formula applied to  $(\bar{x}_1 + \bar{x}_2)/2$ .  
Now

$$t_i = \frac{\bar{x}_{i1} - \bar{x}_{i2}}{s_i \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}, \quad (16.13)$$

where  $s_i$  is the pooled within-class standard deviation for gene  $i$  as before, and hence (ignoring constants in  $n_1$  and  $n_2$ )

$$\begin{aligned} H(x^*) &= h(x^*) - h\left(\frac{\bar{x}_1 + \bar{x}_2}{2}\right) \\ &= \sum_{i \in S(\alpha)} \frac{(\bar{x}_{i1} - \bar{x}_{i2})x_i^*}{s_i} - \sum_{i \in S(\alpha)} \frac{(\bar{x}_{i1} - \bar{x}_{i2})}{s_i} \frac{(\bar{x}_{i1} + \bar{x}_{i2})}{2} \end{aligned} \quad (16.14)$$

They classify to class 1 if  $H(x^*) > 0$ , else to class 2.

Golub et al. (1999) use a very similar procedure. They screen the genes using the absolute value of a ‘‘correlation measure’’

$$\rho_i = \frac{\bar{x}_{i1} - \bar{x}_{i2}}{s_{i1} + s_{i2}}, \quad (16.15)$$

where  $s_{i1}$  and  $s_{i2}$  are the class-1 and class-2 standard deviations for gene  $i$ . This is very similar to a t-statistic, using a somewhat unorthodox estimate of the common standard deviation. They then form a ‘‘weighted vote’’, which amounts to using the linear function

$$\begin{aligned} G(x^*) &= \sum_{i \in S(m)} \rho_i \left(x_i^* - \frac{\bar{x}_{i1} + \bar{x}_{i2}}{2}\right) \\ &= \sum_{i \in S(m)} \frac{(\bar{x}_{i1} - \bar{x}_{i2})x_i^*}{s_{i1} + s_{i2}} - \sum_{i \in S(m)} \frac{(\bar{x}_{i1} - \bar{x}_{i2})}{s_{i1} + s_{i2}} \frac{(\bar{x}_{i1} + \bar{x}_{i2})}{2} \end{aligned} \quad (16.16)$$

Again, if  $G(x^*) > 0$ , we assign to class 1, else class 2. The actual screening procedure used by Golub et al. (1999) is fairly complex, and depends on a parameter  $m$  (the number of genes retained per class);  $m$  is determined by cross-validation.

We can re-express our linear discriminant scores for the two class case. Rather than picking the maximum of  $\tilde{\delta}_1(x^*)$  and  $\tilde{\delta}_2(x^*)$ , we can form the difference and threshold at zero. These differences, after a little manipulation, can be seen to be

$$\begin{aligned} \ell(x^*) &= \tilde{\delta}_1(x^*) - \tilde{\delta}_2(x^*) \\ &= \sum_{i \in S(\Delta)} \frac{(\bar{x}'_{i1} - \bar{x}'_{i2})x_i^*}{s_i^2} - \sum_{i \in S(\Delta)} \frac{(\bar{x}'_{i1} - \bar{x}'_{i2})}{s_i^2} \frac{(\bar{x}'_{i1} + \bar{x}'_{i2})}{2} + \log \frac{\pi_1}{\pi_2} \end{aligned} \quad (16.17)$$

The sums are over the set  $S(\Delta)$  of genes remaining after soft-thresholding. Here it is clear that if both genes are shrunk to the overall mean, they drop out of the sum.

The procedures are similar in form but differ in the details.

- The LDA score uses a pooled variance  $s_i^2$  in the denominator, whereas the other two use standard deviations. The variance is the natural quantity to use since it arises from the squared standardized distance.
- The selection methods are different. Both Golub et al. (1999) and Hedenfalk et al. (2001) use *hard* thresholding of the t-statistics as the basis of selection (hard thresholding selects only those genes whose t-statistics falls above a threshold). Our selection is based on soft thresholding of t-statistics. Soft thresholding is known to perform better in other settings, see e.g. Donoho and Johnstone (1994b).
- We use cross-validation to select  $\Delta$ , and Golub et al. (1999) use it to select their  $m$ . We could not determine whether Hedenfalk et al. (2001) use cross-validation to select  $\alpha$  (it would be reasonable to do so).
- Our procedure can adjust the classification according to the class prior probabilities.
- Our procedure produces probabilities (16.8) for each of the classes.
- Our nearest prototype procedure works for more than two classes.

With more than two classes, nearest shrunken centroids soft-thresholds all of the differences between the class centroids and the overall centroid. In the process, it chooses different sets of genes for characterizing each class.

We saw in Section 16.3 that our nearest prototype scores (16.6) are a restricted form of the LDA scores (16.9)

$$\delta_k^{LDA}(x^*) = (x^* - \bar{x}_k)^T W^{-1} (x^* - \bar{x}_k).$$

Since  $p \gg n$ , the within-class covariance  $W$  is singular, and so this solution is undefined. Nearest shrunken centroids makes  $W$  diagonal which solves the singularity problem; in addition it shrinks the  $\bar{x}_k$ .

## 17 Technical details- supervised principal components

The main references for this technique are Bair and Tibshirani (2004) and Bair et al. (2004).

### 17.1 Extracting supervised principal components

We assume there are  $p$  features measured on  $N$  observations (e.g. patients). Let  $\mathbf{X}$  be an  $N \times p$  matrix of feature measurements (e.g. genes), and  $y$  the  $N$ -vector of outcome measurements. We assume that the outcome is a quantitative variable; below we discuss other types of outcomes such as censored survival times. Here in a nutshell is the supervised principal component proposal:

*Supervised principal components*

1. Compute (univariate) standard regression coefficients for each feature
2. Form a reduced data matrix consisting of only those features whose univariate coefficient exceeds a threshold  $\theta$  in absolute value ( $\theta$  is estimated by cross-validation)
3. Compute the first (or first few) principal components of the reduced data matrix
4. Use these principal component(s) in a regression model to predict the outcome

We now give details of the method. Assume that the columns of  $\mathbf{X}$  (variables) have been centered to have mean zero. Write the singular value decomposition of  $\mathbf{X}$  as

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T \quad (17.1)$$

where  $\mathbf{U}$ ,  $\mathbf{D}$ ,  $\mathbf{V}$  are  $N \times m$ ,  $m \times m$  and  $m \times p$  respectively, and  $m = \min(N - 1, p)$  is the rank of  $\mathbf{X}$ .  $\mathbf{D}$  is a diagonal matrix containing the singular values  $d_j$ ; the columns of  $\mathbf{U}$  are the principal components  $u_1, u_2, \dots, u_m$ ; these are assumed to be ordered so that  $d_1 \geq d_2 \geq \dots, d_m \geq 0$ .

Let  $s$  be the  $p$ -vector of standardized regression coefficients for measuring the univariate effect of each gene separately on  $y$ :

$$s_j = \frac{x_j^T y}{\|x_j\|}, \quad (17.2)$$

with  $\|x_j\| = \sqrt{x_j^T x_j}$ . Actually, a scale estimate  $\hat{\sigma}$  is missing in each of the  $s_j$ , but since it is common to all, we can omit it. Let  $C_\theta$  be the collection of indices such that  $|s_j| > \theta$ . We denote by  $\mathbf{X}_\theta$  the matrix consisting of the columns of  $\mathbf{X}$  corresponding to  $C_\theta$ . The SVD of  $\mathbf{X}_\theta$  is

$$\mathbf{X}_\theta = \mathbf{U}_\theta \mathbf{D}_\theta \mathbf{V}_\theta^T \quad (17.3)$$

Letting  $\mathbf{U}_\theta = (u_{\theta,1}, u_{\theta,2}, \dots, u_{\theta,m})$ , we call  $u_{\theta,1}$  the first supervised principal component of  $\mathbf{X}$ , and so on. We now fit a univariate linear regression model with response  $y$  and predictor  $u_{\theta,1}$ ,

$$\hat{y}^{\text{spc},\theta} = \bar{y} + \hat{\gamma} \cdot u_{\theta,1}. \quad (17.4)$$

Note that since  $u_{\theta,1}$  is a left singular vector of  $\mathbf{X}_\theta$ , it has mean zero and unit norm. Hence  $\hat{\gamma} = u_{\theta,1}^T y$ , and the intercept is  $\bar{y}$ , the mean of  $y$  (expanded here as a vector of such means).

We use cross-validation of the log-likelihood (or log partial-likelihood) ratio statistic to estimate the best value of  $\theta$ .

Note that from (17.3),

$$\begin{aligned} \mathbf{U}_\theta &= \mathbf{X}_\theta \mathbf{V}_\theta \mathbf{D}_\theta^{-1} \\ &= \mathbf{X}_\theta \mathbf{W}_\theta. \end{aligned} \tag{17.5}$$

So, for example,  $u_{\theta,1}$  is a linear combination of the columns of  $\mathbf{X}_\theta$ :  $u_{\theta,1} = \mathbf{X}_\theta w_{\theta,1}$ . Hence our linear regression model estimate can be viewed as a restricted linear model estimate using *all* the predictors in  $\mathbf{X}_\theta$ :

$$\hat{y}^{\text{spc},\theta} = \bar{y} + \hat{\gamma} \cdot \mathbf{X}_\theta w_{\theta,1} \tag{17.6}$$

$$= \bar{y} + \mathbf{X}_\theta \hat{\beta}_\theta, \tag{17.7}$$

where  $\hat{\beta}_\theta = \hat{\gamma} w_{\theta,1}$ . In fact, by padding  $w_{\theta,1}$  with zeros (corresponding to the genes excluded by  $C_\theta$ ), our estimate is linear in all  $p$  genes.

Given a test feature vector  $x^*$ , we can make predictions from our regression model as follows:

1. We center each component of  $x^*$  using the means we derived on the training data:  $x_j^* \leftarrow x_j^* - \bar{x}_j$ .
2.  $\hat{y}^* = \bar{y} + \hat{\gamma} \cdot x_\theta^{*T} w_{\theta,1} = \bar{y} + x_\theta^{*T} \hat{\beta}_\theta$ ,

where  $x_\theta^*$  is the appropriate sub-vector of  $x^*$ .

In the case of uncorrelated predictors, it is easy to verify that the supervised principal components procedure has the desired behavior: it yields all predictors whose standardized univariate coefficients exceed  $\theta$  in absolute value.

Our proposal is also applicable to generalized regression settings, for example survival data, classification problems, or data typically analyzed by a generalized linear model. In these cases we use a score statistic in place of the standardized regression coefficients in (17.2) and use a proportional hazards or appropriate generalized regression in (17.4). Let  $\ell_j(\beta)$  be the log-likelihood or partial likelihood relating the data for a single predictor  $X_j$  and the outcome  $y$ , and let  $U_j(\beta_0) = d\ell/d\beta|_{\beta=\beta_0}$ ,  $I_j(\beta_0) = -d^2\ell_j/d\beta^2|_{\beta=\beta_0}$ . Then the score statistic for predictor  $j$  has the form

$$s_j = \frac{U_j(0)^2}{I_j(0)}. \tag{17.8}$$

Of course for the Gaussian log-likelihood, this quantity is equivalent to the standardized regression coefficient (17.2).

## 17.2 Importance scores and a reduced predictor

Having derived the predictor  $u_{\theta,1}$ , how do we assess the contributions of the  $p$  individual features? We compute the *importance score* as the inner product between each centered feature that has been selected and  $u_{\theta,1}$ :

$$\text{imp}_j = (x_j - \bar{x}_j)^T, u_{\theta,1} \quad (17.9)$$

This is proportional to the loading for that gene. Features  $j$  with large values of  $|\text{imp}_j|$  contribute most to the prediction of  $y$ .

In some applications we would like to have a model that uses only a small number of features. For example, a predictor that requires expression measurements for a few thousand genes is not likely to be useful in a everyday clinical settings: microarrays are too expensive and complex for everyday use, and simpler assays like RT-PCR can only measure 50 or 100 genes at a time. In addition, isolation of a smaller gene set could aid in biological understanding of the disease.

There are a number of ways to obtain a series of reduced models. One would be to apply the lasso Tibshirani (1996) to the data  $(\mathbf{X}, \hat{y}^{\text{spc}})$ . The LAR algorithm Efron et al. (2004) provides a convenient method for computing the lasso solutions. One drawback of this approach is that the series of models will typically involve different sets of features, and this can be difficult for a scientist to assimilate.

Here we take a simpler approach. We define

$$\hat{u}_{\text{red}} = \sum_{|\text{imp}_j| > \gamma} \ell_j \cdot x_j \quad (17.10)$$

where  $\ell_j = u_{\theta,1}^T x_j / d_1$ , the loading for the  $j$ th feature ( $d_1$  is the first singular value from the SVD (17.3)). This predictor keeps only features with importance scores  $\gamma$  or larger, and weights these features by their loadings. We illustrate this idea in the next section.

## References

- Bair, E., Hattis, T., Paul, D., and Tibshirani, R. (2004), “Prediction by supervised principal components,” Tech. rep., Stanford University.
- Bair, E. and Tibshirani, R. (2004), “Semi-supervised methods to predict patient survival from gene expression data,” *PLOS Biology*, 2, 511–522.
- Donoho, D. and Johnstone, I. (1994a), “Ideal Spatial Adaptation By Wavelet Shrinkage,” *Biometrika*, 81, 425–455.
- (1994b), “Ideal spatial adaptation by wavelet shrinkage,” *Biometrika*, 81, 425–55.



- Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004), “Least angle regression,” *Annals of Statistics*, 407–409.
- Golub, T., Slonim, D., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J., Coller, H., Loh, M., Downing, J., Caligiuri, M., Bloomfield, C., and Lander, E. (1999), “Molecular classification of cancer: class discovery and class prediction by gene expression monitoring,” *Science*, 286, 531–536.
- Hedenfalk, I., Duggan, D., Chen, Y., Radmacher, M., Bittner, M., Simon, R., Meltzer, P., Gusterson, B., Esteller, M., Raffeld, M., Yakhini, Z., Ben-Dor, A., Dougherty, E., Kononen, J., Buben-dorf, L., Fehrle, W., Pittaluga, S., Gruvberggerm, S., Loman, N., Johannsson, O., Olsson, H., Wilfond, B., Bor, A., and Trent, J. (2001), “Gene-expression profiles in hereditary breast cancer,” *N. Engl. J. Med.*, 344, 539–548.
- Ihaka, R. and Gentleman, R. (1996), “R: A Language for Data Analysis and Graphics,” *Journal of Computational and Graphical Statistics*, 5, 299–314.
- Tibshirani, R. (1996), “Regression shrinkage and selection via the lasso,” *J. Royal. Statist. Soc. B.*, 58, 267–288.
- Tibshirani, R. J., Hastie, T. J., Narasimhan, B., and Chu, G. (2002), “Diagnosis of multiple cancer types by shrunken centroids of gene expression,” *Proceedings of the National Academy of Sciences*, 99, 6567–6572.
- Tusher, V., Tibshirani, R., and Chu, C. (2001), “Significance analysis of microarrays applied to ionizing radiation response,” *Proceedings of the National Academy of Sciences*, first published April 17, 2001, 10.1073/pnas.091062498.