Paper 283-2009

A SAS®/JMP® Integration for Implementation of a Clustering Algorithm for High Dimensional Low Sample Size Data

George von Borries, Universidade de Brasília, Brasília-DF, Brasil

ABSTRACT

A SAS macro solution is presented for clustering of high dimensional low sample size (HDLLSS) data using a new algorithm based o p-values as similarity measure. The algorithm PPCLUST was developed by von Borries (2008) and implemented using SAS macro language with the macro autocall facility and window macro command for friendly interface. The SAS interface to JMP was used to run a SAS macro inside JMP and automatically produce graphs using adaptable JMP scripts. An example with partial data from microarray study is presented.

INTRODUCTION

Clustering is a multivariate problem, also known as unsupervised learning technique, where the goal is to find patterns among a set of input measures without the knowledge of an outcome measure. In multivariate problems with High Dimensional Low Sample Size (HDLSS) Data, the number of variables (p) is large and the information (n_p) available about each variable is very small, such as problems with $p > 15000$ variables and $n_p = 3$ samples. Clustering of HDLSS data is a problem where some assumptions like homocedasticity, normality or balanced data are difficult to ascertain and traditional techniques usually fail in the objective to obtain groups of similar objects. Applications using high dimensional low sample size data are common in current research problems due to the improvements in data collection technologies that allow the obtaining of information from a large number of objects (variables) at once but at a cost that restricts data replication. Examples of areas of such application are:

- Genomics: microarray experimentation with data collected for thousands of genes with few replications (tissues) per genes, which allows scientists to study complex disorders through the monitoring of expression of thousands of genes from a single DNA chip, known as DNA microarray. Tamayo and Ramaswamy (2002), Parmigiani, et al. (2003), McLachlan, Do and Ambroise (2004) are some of the references about the subject.
- Chemometrics: small population of high dimensional spectra as in Parizzi (2005).
- Pattern recognition: small number of images represented by thousands of voxels as in Reese et al. (Unpublished).
- Broadband Sonar: detection and clustering of undersea mines by U.S. Navy. Signals are high dimensional (about 1500) and data acquisition is time-consuming and expensive, resulting in small data sets.
- Agriculture Screening Trials: large number of treatments (cultivars, pesticides) in a complete block design with the limitation of 3 or 4 blocks (Brownie and Boos, 1995).

THE PROBLEM

In the context of HDLSS data, clustering has been one of the most important learning applications used to identify groups of variables, especially in microarray gene expression analysis. The problem is that most of current clustering algorithms for HDLSS data are adaptations from traditional multivariate analysis, where the number of factor levels is not high and sample sizes are relatively large. Those algorithms have some disadvantages as they cannot deal with high dimensionality of data, requiring some dimension reduction prior to the analysis, or requiring working with comparison of pairs of factor levels for clustering. Dimension reduction is one solution criticized by authors like Yeung and Ruzzo (2001), and multiple comparisons result in drastic reduction of Type I error rate of the test and requiring the use of alternative solutions as suggested by Benjamini and Hochberg (1995).

Some exceptions to traditional algorithms are gene-shaving (Hastie et al., 2001), density-based hierarchical clustering (Jiang et al., 2003), clustering via iterative feature filtering or CLIFF (Xing and Karp, 2001), plaid models (Lazzeroni and Owen, 2002), subspace clustering (Parsons et al., 2004), coupled two-way clustering analysis (Getz et al., 2000). Problems with those algorithms arise from documentation, lack of flexibility in implemented software (no code access) and/or incompatibility with different operational systems platforms.von Borries (2008) gives some details about traditional and modern algorithms used in clustering of HDLSS data.

THE SOLUTION

PPCLUST (P-values based Partitional Clustering) is a new algorithm developed by von Borries (2008) for partition clustering of large number of variables with few observations per variables. The procedure is based on ANOVA methods and uses the p-value of the test statistic from Wang and Akritas (2004) as a measure of similarity between groups. The objective is to test if independent observations X_{ij}, $1 \le j \le n_i$, from different variables are from the same distribution, i.e.,

$$H_0 : F_1(x) = \cdots = F_a(x) \qquad i = 1, \cdots, a.$$

The test is based on the statistic $\sqrt{a}(F_R - 1)$ that can be used to obtain a p-value for the test and compare it to some specified significance level α, such that the p-value works as a similarity measure in the algorithm. More details about the test, the calculus of F_R and it's convergence distribution are found in von Borries (2008).

PPCLUST was initially implemented as a macro in SAS 9.1.3 and simulations showed many advantages to some traditional algorithms, as

- **Invariance to monotone transformations**: many clustering algorithms can produce different results before and after monotone transformations on the same data. This is because monotone transformations change distances between variables and therefore modify similarity matrices used in clustering.
- **Automatic specification of number of groups**: PPCLUST SAS macro does not require the specification of number of groups in advance. It determines the number of groups automatically by specification of significance levels that work as thresholds that will be compared with the p-values for testing the hypothesis of no group effect. Usually, the number of groups is found by testing different significance levels and looking for the p-values that produce equal (or almost equal) number of groups. The higher of those p-values should be used as the final choice in the procedure.
- **The algorithm was fast and easy of use**: the use of SAS macro language made the specification of necessary information for the program very easy. Also, SAS showed a very high performance when dealing with large number of variables (sometimes more than 15000) and very small number of observations (3 or 4) when compared with alternative algorithms in commercial and noncommercial statistical packages.
- **No memory allocation problems**: the macro takes advantage of SAS memory allocation and does not have problem in handling as many factors levels as necessary. Many alternative clustering algorithms did not run in tested statistical packages when increasing the number of variables to be grouped.
- **No need of dimension reduction**: since the algorithm and its implementation in SAS does not have limitation in handling as many variables as necessary (the only limitation is its own PC), it is not necessary to apply any dimension-reduction technique, such as principal components (see Johnson, 1998), before clustering data. Studies from Yeung and Ruzzo (2001) show that clustering principal components instead of original data produce different results on many algorithms using different similarity metrics. Instead, PPCLUST relies in high dimension to provide power to give good similarity measure. This is especially appealing with very small number of replications.
- **No requirement of balanced data**: the algorithm works with both balanced and unbalanced data, requiring only that each variable have at least 2 replications.

THE IMPLEMENTED SOLUTION

PPCLUST was initially programmed using SAS/STAT® and SAS/IML® software. In order to facilitate the use of the program, the code was implemented in SAS Macro Language with the alternative of a user "friendly" interface developed using the *%Window* macro command. The macro was compiled using the SAS *Macro Autocall Facility* as explained in Carpenter (1998).

Usually, when grouping HDLSS data, a large number of variables appear in one group that represents variables with no importance in the study. As an example, in microarray studies, a large group of genes usually appears representing genes with same expression in normal or cancer tissues. Those genes are of no importance to the researcher trying to find genes that suffer some kind of modification in expression level due to the development of a cancer. For applications of this kind, PPCLUST was modified in order to produce a file without variables classified in the group with highest number of elements. The modified macro is now called PPCLUSTg.

Finally, it was written a JMP script that uses SAS interface to JMP (version 7 or 8). The script calls macro PPCLUSTg from JMP and generate a cell plot of original data and data grouped by the SAS macro. A detailed description of the implementations is presented bellow.

SAS AND JMP IMPLEMENTATIONS

Before running the SAS macros described bellow a temporary or permanent SAS data set should be ready for analysis. The SAS data set for the implemented macros should be in the format presented in Table 1, where column ID is optional and has the label of each variable to be clustered. Note that the data has variables in lines and observations in columns, as is common in microarray data. For example, a microarray data would have a column (ID) corresponding to each gene name (variables). The columns *X1, X2,…, Xn* would represent the replications (observations for each gene) and should be named with a prefix name plus numbers in sequence. A data set with three replications could replications named by *R1, R2, R3* or *d1, d2, d3* or *Var1, Var2, Var3,* among other options.

Table 1: High dimensional replicated data set layout. Here $a \to \infty$ and $n_i \geq 2$.

ID	$X1$	$X2$	$\cdots$	Xn
1	X_{11}	X_{11}	$\cdots$	X_{1n}
2	X_{21}	X_{22}	$\cdots$	X_{2n}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
a	X_{a1}	X_{a2}	$\cdots$	X_{an}

- **Implementation 1**: PPCLUSTg is a SAS macro for partitioning clustering of HDLSS data as described in previous sections. The use of PPCLUSTg macro is described in the following steps:

Step 1: If the SAS data file is a permanent file, then create a libname indicating the directory where the file is located. Example:

```
libname study 'c:\myfiles\studydata';
```

Step 2: The use of the *Macro Autocall Facility*, avoid submitting PPCLUST every time a new SAS session is started. The only requirement is to include the following commands in the start of your code before execution of ppclust:

```
libname sasmac 'c:\sasmacros';
options mstored sasmstore=sasmac;
```

The command indicates the existence of compiled macros in the directory c:\sasmacros. The macros are saved in the SAS catalog sasmacr. You can also include the command in your local SAS configuration file to run it every time SAS is started.

Step 3: The macro execution. You can start the macro using a command line or using a simple window interface.

Command line:

```
%ppclustg(dataset,obsmin,obsmax,alpha,saslib);
```

In this case **Dataset** is the SAS data name, **obsmin** is the name of the variable indicating the first replication, **obsmax** is the name of the variable indicating the last replication, **alpha** is the threshold parameter to be compared with p-values in the clustering algorithm, as explained before, and saslib is a pre-defined library that indicates where the output files should be saved. If no library is defined then the output files are saved in work library. PPCLUSTg macro produces three temporary SAS data sets that can be saved or exported using common SAS commands. The data set **grfr** has the number of factor levels in each created group, while the data set **datanew** has the original data with the information in data **groupclass** added to it. Finally, **datanewclean** has the same content of datanew but with variables in the largest group deleted from the file, facilitating graphical visualization of clustering results as will be discussed in next section.

Window interface:

```
%ppclustgw;
```

After executing the previous command, a window will appear requiring the dataset, obsmin, obsmax, alpha and additional information to be included in the respective fields. The macro window option also allows you to save the output data from PPCLUST in a directory in the system. Each output file can be saved in one of the following formats: pdf (portable document format), rtf (rich text format), cvs (comma separated value) and/or html. Figure 1 shows the window interface read for data input.

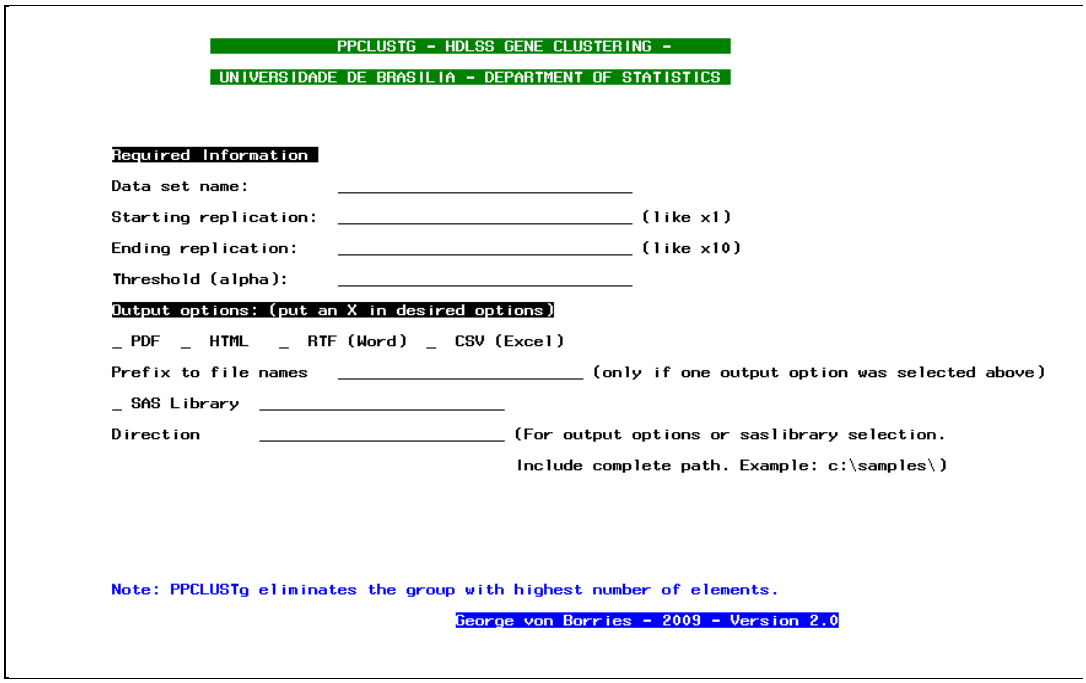


Figure 1. PPCLUSTgw window interface for data input.

- **Implementation 2**: PPCLUSTj is a JMP script that combines SAS commands and JMP commands. A small modification in the script allows the use of results from PPCLUSTg and production of graphs from JMP automatically. The graphs produced by PPCLUSTj are cell plots of original and grouped data ordered by groups. With small modifications, one can improve the script in order to add any other graph usefull for posterior analysis. PPCLUSTj script is reproduced bellow with comments about adaptations for different analysis. The steps discussed bellow use JMP version 8 for windows.

Step 1: Open JMP and choose the SAS category in JMP Starter.

Step 2: Choose the server connection option and in the window indicate if the SAS is used remotely or not. Figure 3 shows an example where is selected SAS on local machine, as if a SAS is installed in a personal desktop, for example.

Step 3: After selection of the location where SAS is running, include the macro PPCLUSTj using the Open Script option in the File category of JMP Starter. Specific commands are indicated in bold blue, between < > symbols. Details will be given in the next section with an example in microarray data analysis.

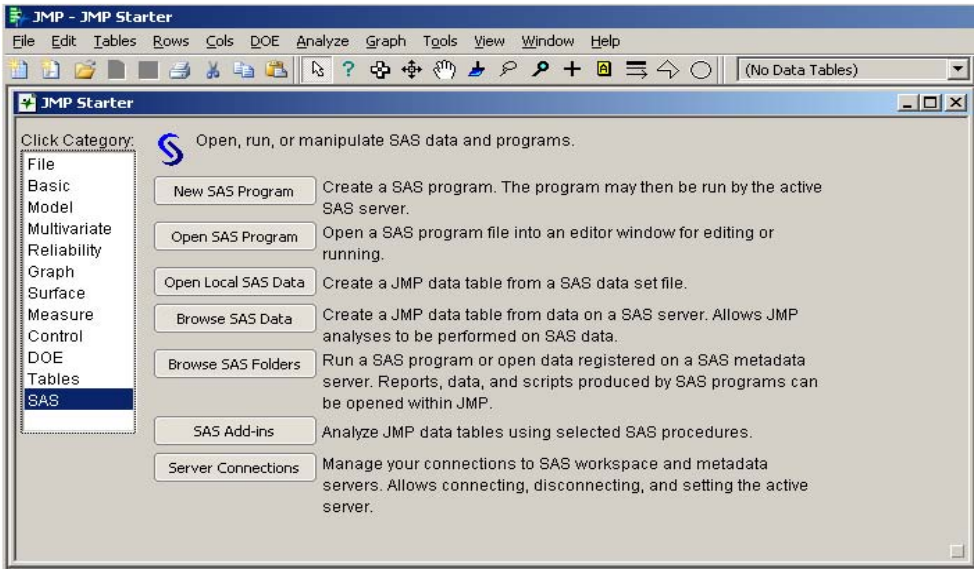


Figure 2. SAS options in JMP Starter Windows.

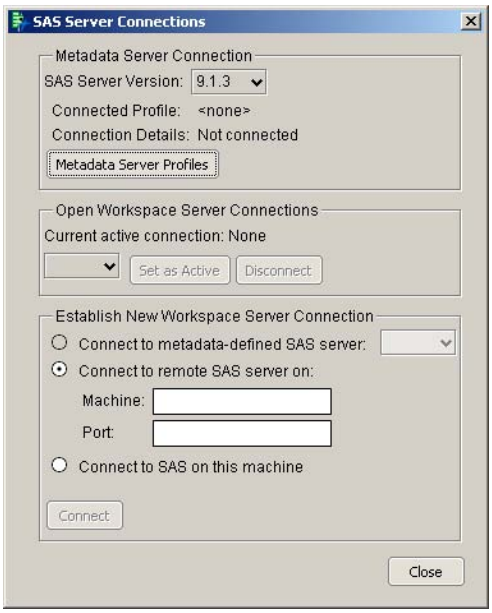


Figure 3: Connecting to Local SAS using JMP SAS Server Connections.

PPCLUSTj (Script in JMP):

```
SAS Submit("libname ppclustg '<local of ppclustg macro>';
options mstored sasmstore=ppclustg;
libname <name> '<local where results will be stored>';
%ppclustg(<ppclustg required parameters>);
run;");

Open("<location where datanew was stored>\datanew.sas7bdat");
Cell Plot(
    Y(
           :<observation 1 name>,
           :<observation 2 name>,
           :<…>,
           :<observation n name>
    ),
    Center at zero,
    SendToReport(
           Dispatch( {}, "", NomAxisBox, Rotated Tick Labels( 1 ) )
    )
);

Open("<location where datanewclean was stored>\datanewclean.sas7bdat");
Cell Plot(
           Y(
           :<observation 1 name>,
           :<observation 2 name>,
           :<…>,
           :<observation n name>
    ),
    Center at zero,
    SendToReport(
           Dispatch( {}, "", NomAxisBox, Rotated Tick Labels( 1 ) )
    )
);
```

DATA ANALYSIS EXAMPLE

In order to illustrate the use of PPCLUSTg, PPCLUSTgw and PPCLUSTj, it was used a partial data from microarray expression profiles of paired normal and carcinoma tissues in colorectal cancer. The original data was studied in Notterman et al. (2001) and von Borries (2008). Here it is used only the expression levels of 10 paired tissues and not 18 as in the original study. Details about how the data was collected can be found in Notterman et al. (2001) and complete data set study using PPCLUST algorithm can be found in von Borries (2008).

A permanent data set SAS, called difca, was created with a code number and 10 observations (d1...d10) for each of 4234 variables. The data was stored in a local directory named study and the results will be saved in a subdirectory named results. It is important to emphasize that the data has many missing values, i.e., it is unbalanced, but this is not a problem for the implemented algorithm.

ANALYSIS 1: PPCLUSTG

In the first analysis one wants to cluster the 4234 variables in data difca using a ppclustg command line. The code necessary to run the macro[1] is

1 The compiled sas macro was created with macro autocall facility and is available in the directory study as a SAS catalog named SASMACR.

```
libname ppclust 'c:\study';
libname results 'c:\study\results';
options mstored sasmstore=ppclust;
%ppclustg(ppclust.difca,d1,d10,1e-8,results);
```

Note that the directory for libname ppclust should be changed if the macro code and permanent sas data set are stored in a different place. The results are stored in the library results. As explained before, three data sets are produced, **grfr** (Figure 4) with number of variable in each group, **datanew** with original data plus variables idobs and group indicating position of original variable in the data set and the group it was allocated, respectively. The last created data set is **datanewclean** that reproduces the content of datanew without the variables in the largest obtained group, i.e, in the previous example all variables in group 4 (see Figure 4) are deleted from the data set datanew and the remaining ones saved in datanewclean.

	GROUP	Frequency Count	Percent of Total Frequency
1	0	2	0.0472366556
2	1	18	0.4251299008
3	2	469	11.076995749
4	3	2	0.0472366556
5	4	3289	77.680680208
6	5	454	10.722720831

Figure 4: File grfr obtained from clustering with PPCLUST for artificial data set.

ANALYSIS 2: PPCLUSTGW

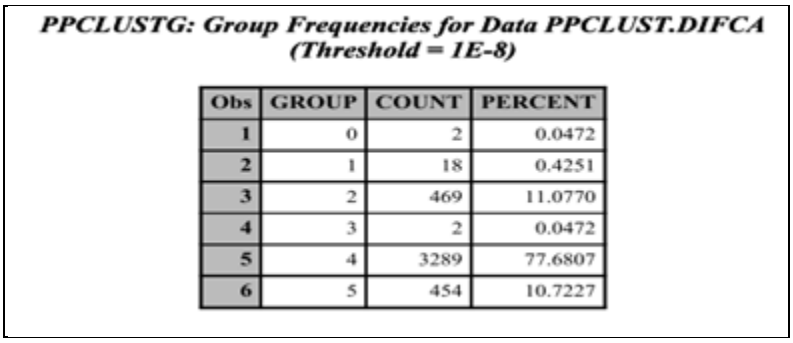
Here, the previous analysis is repeated but with the macro window resource used as a friendly interface. In this case the program allows the creation of pdf, html, csv and rtf versions of each output file. In order to call the macro window, it is necessary the following code:

```
libname ppclust 'c:\study';
options mstored sasmstore=ppclust;
%ppclustgw;
```

PPCLUSTgw macro will call a window interface that will receive the information necessary before running PPCLUSTg. When Figure 1 appears you should complete each field according to the following explanation.

- **Data set name**: name of data set to be analysed. Example: PPCLUST.DIFCA.
- **Starting replication**: first variable indicating replication (observation) of each variable. Example: D1.
- **Ending replication**: last variable indicating replication (observation) of each variable. Example: D10.
- **Threshold (alpha)**: significance level in testing if a group of variables are from same group. Example: 1E-8.
- X **PDF**: selection of PDF output option for results. Note: You can put an X in each field at the same time.
- **Prefix to File Names**: label to be added as a prefix to each output file (pdf, rtf, csv or html). Only necessary if one output file option was selected. Example: GVB.
- **SAS Library**: name of sas library to store results. Example: RESULT.
- **Direction**: direction of created SAS library. Example: C:\STUDY\RESULTS\.

The previous sequence of examples will produce a pdf file labeled GVBgrfr.pdf in the directory c:\study\results.



PPCLUSTG: Group Frequencies for Data PPCLUST.DIFCA (Threshold = 1E-8)

Obs	GROUP	COUNT	PERCENT
1	0	2	0.0472
2	1	18	0.4251
3	2	469	11.0770
4	3	2	0.0472
5	4	3289	77.6807
6	5	454	10.7227

Figure 5: Reproduction of contents of file GVBgrfr.pdf produced in previous example.

ANALYSIS 3: PPCLUSTJ

Since the two previous analysis produces SAS data sets as results, one can explore the graphical resources of SAS to visualize the groups obtained in more detail. One option is to use Proc SGPLOT in SAS 9.2 to produce heatmaps (See technical report in http://support.sas.com/rnd/papers/sgf2008/butterfly_handout.pdf). Heatmap is a matrix that maps the expression levels of each gene to a color intensity value. A very close option to heatmaps is the Cell Plot produced by JMP software. The objective of PPCLUSTj is to integrate the SAS macro command with the flexibility and facility of JMP in other to produce visual representations of the results obtained in the study. JMP has complete integration with SAS data sets. PPCLUSTj is a JMP script file with the objective to automatically call the SAS macro **ppclustg** and generate cell plots from the output files in the study.

The first step before running the JMP script is to indicate the location of SAS System as shown in Figures 2 and 3. For the previous study PPCLUSTj code is reproduced bellow with comments in boldface font.

```
SAS Submit("libname ppclustg 'c:\study';
options mstored sasmstore=ppclustg;
libname results 'c:\study\results';
%ppclustg(ppclustg.difca,d1,d10,1e-8,results);
run;");
```

PREVIOUS SAS COMMANDS USING SAS JMP INTERFACE.

```
Open("c:\study\results\datanew.sas7bdat");
Cell Plot(
       Y( :d1,   :d2,   :d3,   :d4,   :d5,
          :d6,   :d7,   :d8,   :d9,   :d10
       ),
       Center at zero,
       SendToReport(
           Dispatch( {}, "", NomAxisBox, Rotated Tick Labels( 1 ) )
       )
);
```

OPENING ORIGINAL DATA.

VARIABLES INDICATED HERE.

```
Open("c:\study\results\datanewclean.sas7bdat");
Cell Plot(
       Y( :d1,   :d2,   :d3,   :d4,   :d5,
          :d6,   :d7,   :d8,   :d9,   :d10
       ),
```

OPENING DATA FROM PPCLUSTG WITHOUT LARGEST GROUP.

VARIABLES INDICATED HERE.

```
        Center at zero,
        SendToReport(
              Dispatch( {}, "", NomAxisBox, Rotated Tick Labels( 1 ) )
        )
);
```

The Script above automatically executes the macro **ppclustg** in SAS and produces two cell plots using JMP. The first one with original variables not ordered, and the second plot with the variables ordered by group and without the group with largest number of variables, i.e., only extreme groups are considered. Figure 6 has the representation of both cell plots.

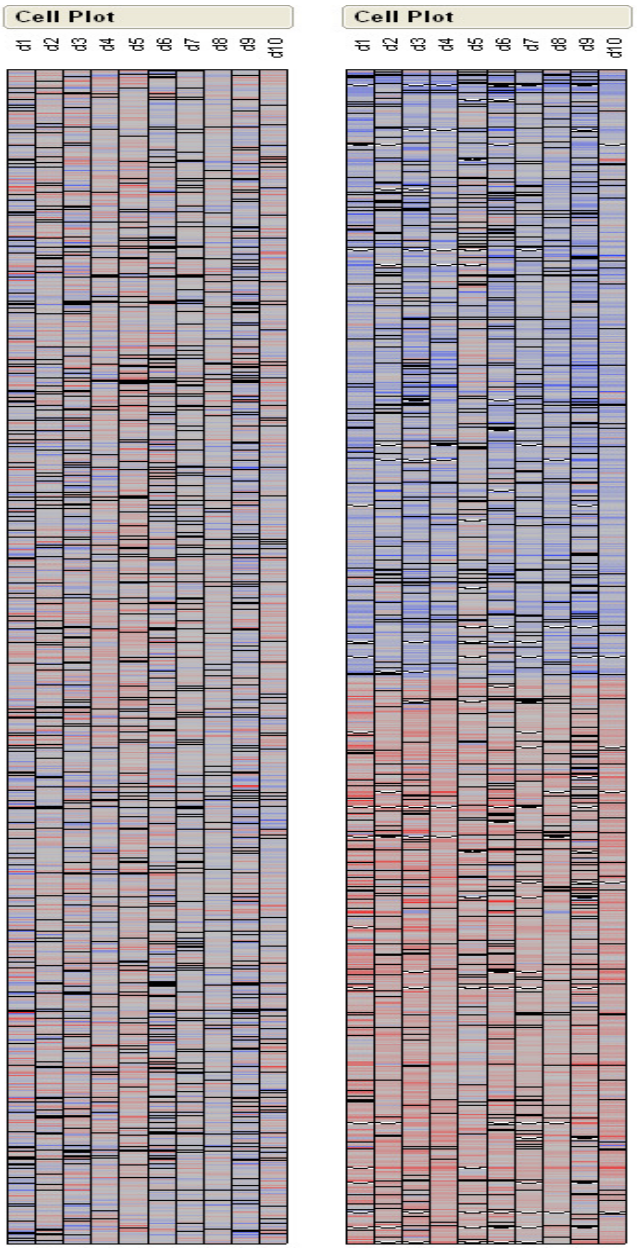


Figure 7: Cell Plot for original data (left) and Cell Plot for grouped data without the variables in largest group. Note: Both cell plots are represented in equal size, but left plot has 4234 variables (lines) and right plot has 945 variables (lines).

After clustering and eliminating the group with largest number of variables, we can observe the evident difference between the extreme groups. In Figure 7, groups in blue represent the ones with highest values and groups in red represent the groups with lowest values. In gene expression studies it could mean expressed genes in some type of cancer, for example. The identification of expressed genes would help molecular biologists to identify characteristics of genes that are highly expressed (positively or negatively) in cancer cells and help to conduct treatments.

CONCLUSION

SAS Macro language has a simple tool that allows the production of good user-friendly programs. The new integration of SAS with JMP allows one to automatically execute macros using SAS Macro Autocall Facility and include all graphical capabilities of JMP in the final results in a simple way. Actually, any JMP resource can be integrated with SAS commands through the SAS interface to JMP, increasing the power of SAS and JMP as analytical tools.

REFERENCES

[1] Benjamini, Y. and Hochberg, *Controling the false discovery rate: a practical and powerful approach to multiple testing,*JRSSB, 57, 289-300, 1995.

[2] Boos, D.D. and Brownie, C., *Anova and rank tests when the number of treatment is large,* Statistics & Probability Letters, 23, 183-191, 1995.

[3] Carpenter, A. *Carpenter's Complete Guide to the SAS Macro Language.* Cary, NC: SAS Institute Inc., 1998.

[4] Getz, G., Levine, E., and Domany, E., *Coupled two-way clustering analysis of gene microarray data,* PNAS, 97, 12079-12084, 2000.

[5] Hastie, T., Tibshirani, R., and Friedman, J., *The elements of statistical learning (data mining, inference and prediction),* 2001.

[6] Jiang, D., Pei, J., and Zhang, A., *DHC: a density-based hierarchical clustering method for time-series gene expression data,* Proc. BIBE2003: Third IEEE int'l Symp. Bioinformatics and Bioeng., 2003.

[7] Johnson, D.E., *Applied multivariate methods for data analysts,* Duxbury, 1998.

[8] Lazzeroni, L. and Owen, A., *Plaid models for gene expression data,* Statistica Sinica, 12, 61-86, 2002.

[9] McLachlan, G.J. and Do, K.A., and Ambroise, C., *Analysing microarray gene expression data,* Wiley-Interscience, 2004.

[10] Notterman, D.A., Alon, U., Sierk, A.J., and Levine, A.J., *Transcriptional gene expression profiles of colorectal adenoma, adenocarcinoma, and normal tissue examined by oligonucleotide arrays,* Cancer Research, 61, 3124-3130, 2001.

[11] Parizzi, F.C., *Espectroscopia de infra-vermelho próximo na identificação de defeitos e fungos em grãos de café*, . Tese de Doutorado, UFV, 2005.

[12] Parmagiani, G., Garrett, E.S., Irzarry, R.A., and Zeger, S.L. (eds.), *The analysis of gene expression data: methods and software,* Springer, 2003.

[13] Parsons, L., Haque, E., and Liu, H., *Subspace clustering for high dimensional data: a review,* Tech. Report, Arizona State University, 2004.

[14] Reese, S., Sukthankar, G., and Sukthankar, R., *An efficient recognition technique for minelike objects using nearest-neighbor classification,* Tech. Report, Intel Corporation, 2003.

[15] Tamayo, P. and Ramaswamyh, S., *Cancer genomics and molecular pattern recognition,* Tech. Report, MIT, Cambridge and Harvard, 2002.

[16] von Borries, G.F., *Partition Clustering of High Dimensional Low Sample Size Data Based on P-values,* PhD Dissertation, Kansas State University, 2008.

[17] Wang, H. and Akritas, M.G., *Rank tests for ANOVA with large number of factor levels,* Nonparametric Statistics, 16, 563-589, 2004.

[18] Xing, E.P. and Karp, R.M., *CLIFF: clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts,* Bioinformatics, 17, 306-315, 2001.

[19] Yeung, K.Y. and Ruzzo, W.L., *Principal component analysis for clustering gene expression data,* Bioinformatics, 9, 763-774, 2001.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name: George Freitas von Borries
Enterprise: Universidade de Brasília
Address: Campus Universitário Darcy Ribeiro, Instituto de Ciências Exatas, Departamento de Estatística, Asa Norte, Brasília, Distrito Federal, 70910900.
Work Phone: 55 61 32736317
Fax: 55 61 32736317
E-mail: gborries@unb.br