Paper SP07-2011

JMP® as an Analytic Hub: Using JMP to Build Custom Applications via SAS® and R

Kelci Miclaus, Clay Barker, Jun Ge, SAS Institute, Cary, NC

ABSTRACT

In today's data-rich era, statisticians and analysts have a number of tools available for data analysis. This paper presents examples and explores applications of using JMP® 9 as a data analysis hub. The integration of JMP with both SAS and R—coupled with a new add-in architecture that makes it easy to create, distribute, and install custom applications—provides unlimited options for data exploration and analysis. Examples presented highlight the incorporation of R packages to perform novel analysis techniques as well as SAS procedures to implement advanced statistical methods within the JMP graphical user interface. Such applications allow you to take advantage of the dynamically interactive JMP visualizations, the depth of SAS statistical procedures, and the breadth of R packages available. Addressing these examples, tips, and techniques on building custom dialog-based applications via JMP Scripting Language (JSL) with SAS and R programming is highlighted, as well as how to create a JMP add-in.

INTRODUCTION

JMP is a comprehensive interactive statistical and visual tool that allows you to perform a variety of analyses through a friendly graphical user interface. That, however, is just the surface of JMP. One of its best and most often overlooked features is extensibility. Once you become familiar with JMP, there are dozens of ways that you can add your own customized features to JMP, as well as extend the use of JMP beyond just the platforms that are surfaced to you in the menu. In fact, you can extend JMP even further through integrations with other software tools. JMP has offered integration capabilities with SAS since JMP 7. The current release of JMP 9 now adds the ability to integrate with the R project (http://www.r-project.org/) as well as making continual advances in the seamless integration to SAS. Both integrations can be used through the JMP Scripting Language (JSL). JSL is a powerful tool provided by JMP that allows you to develop custom analyses. Through JSL you can launch platforms, perform matrix computations and data set manipulation, and create custom dialogs, output windows, and graphics. Several JSL commands also exist for facilitating the integrations with SAS and R, the most basic of which allow you to connect, import/export data, and submit native code to the external software tool. With such functionality, it becomes easy to call to SAS or R to perform a statistical technique that is not covered by the JMP platforms.

Another feature new in JMP 9 is the add-in architecture; you can create JMP add-in files that package JSL scripts, menu customizations, data tables, and so on. These files contain all that is needed to share a custom analysis tool that has been created in JMP. An add-in can extend JMP with something as small as a new JSL function or as large as a full menu containing hundreds of custom dialogs and processes. In this paper, we present several examples of custom JMP applications that use the integration tools (SAS or R) available in JMP. These examples range from formal commercial products that SAS offers to individually written scripts that enhance a specific analysis.

VERTICAL PRODUCT OFFERINGS FROM JMP

The JMP family of products includes two vertical application tools in the life sciences realm. These tools are JMP Genomics and JMP Clinical. JMP Genomics 5.0 has been said to be the "mother of all add-ins." This is an apt name since the product is actually a software bundle that uses both JMP and SAS to build literally hundreds of custom analytical processes specific for analysis of genomic data. Genomic data are massive; a typical analysis of variation in DNA and how it relates to disease can easily contain over a million columns where each column contains data from locations in our genetic code (called single nucleotide polymorphisms [SNPs]) where we differ. Therefore, analysis is best carried out in the file-based SAS system to avoid memory constraints. JMP, on the other hand, is the perfect complement for visualizing the results of genomic analyses. The JMP Genomics system surfaces processes as JMP dialogs under the Genomics menu and each process works by firing off a SAS job to perform analysis in SAS (and on SAS data sets). When SAS is finished, a JSL script launches the results back to JMP for interactive visualization of results. The seamless integration of the product in its latest release is facilitated by the add-in architecture to create the most comprehensive genomics analysis tool on the market for analyzing genome-wide data, gene expression data, copy number, and data produced by modern sequencing tools. Figure 1 is a screenshot showing just a small portion of some of the analysis and visualization mechanisms available in JMP Genomics.

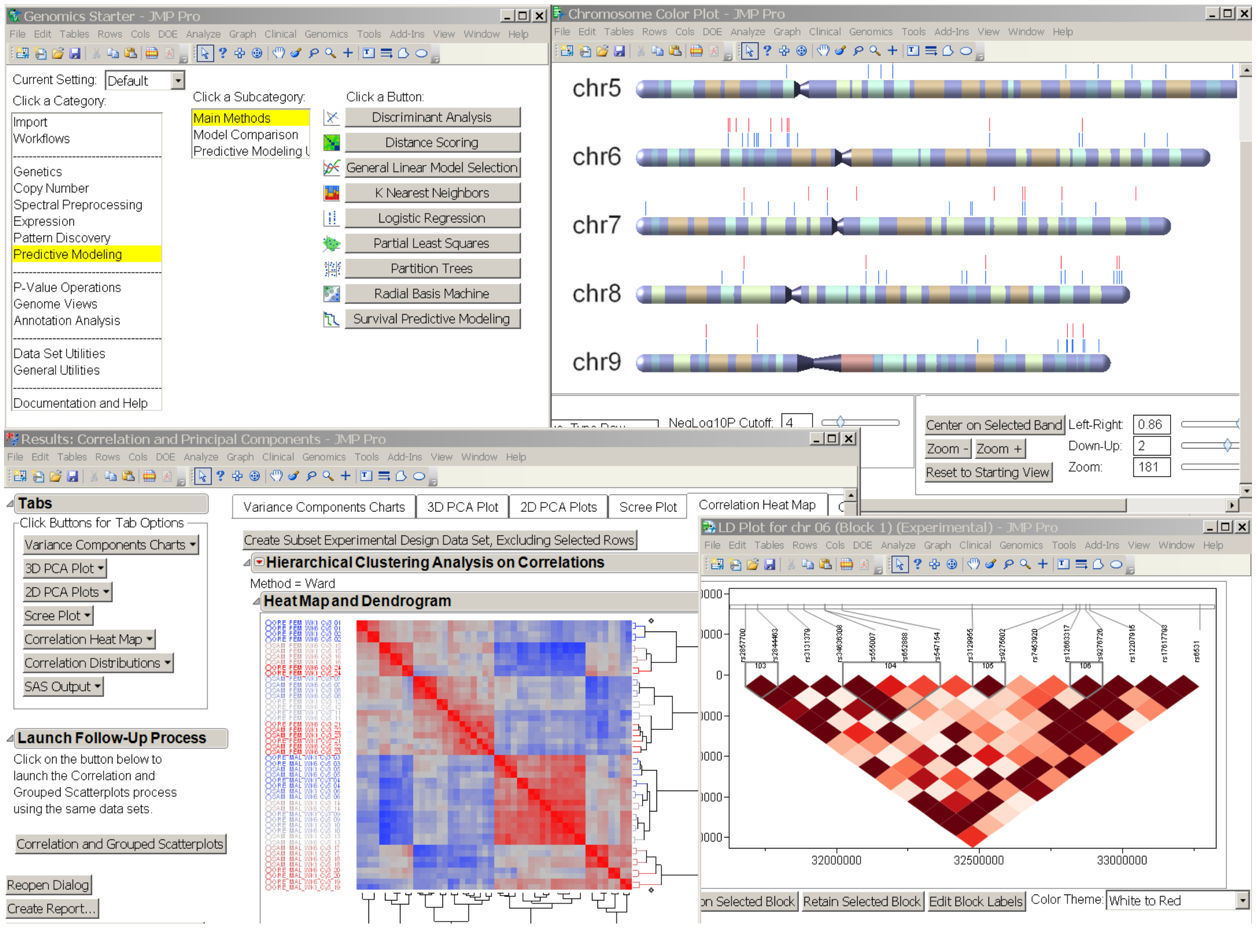
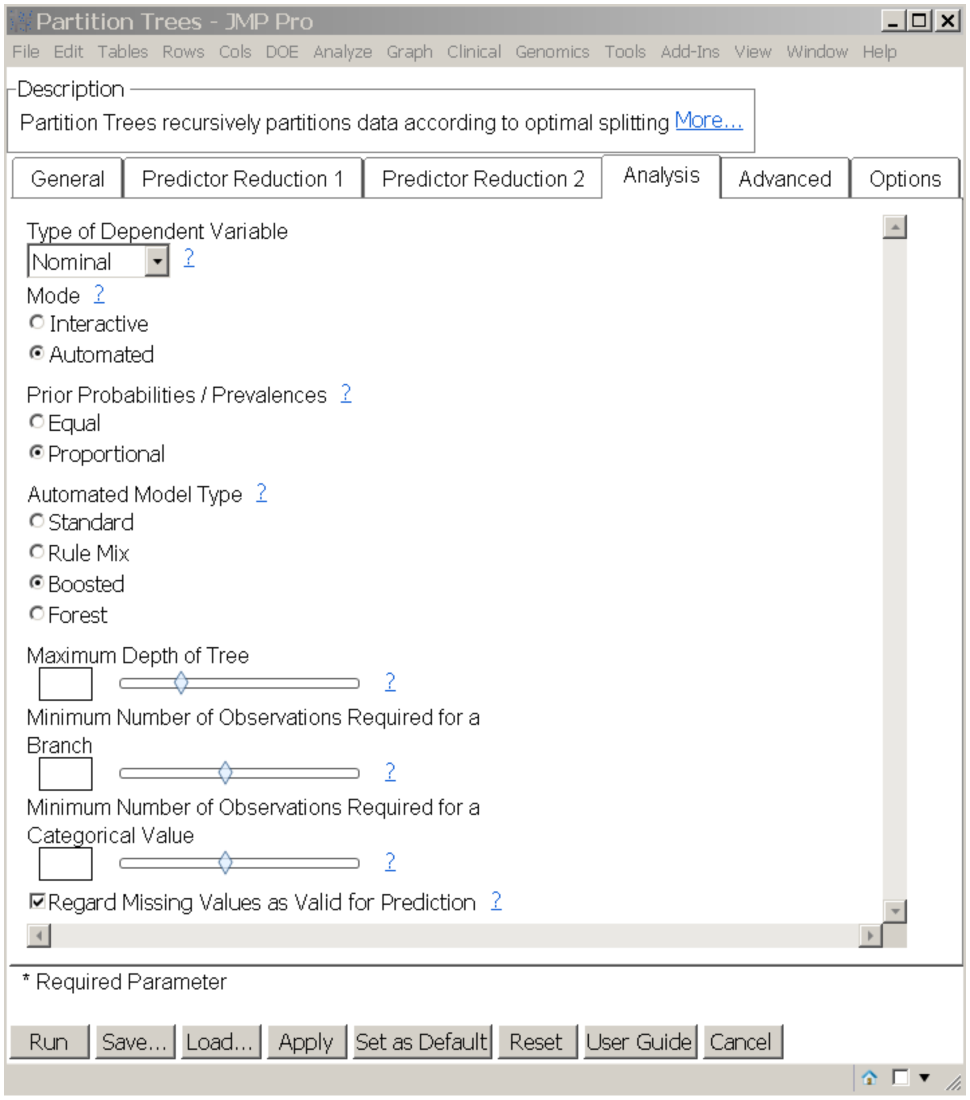


Figure 1. A Screenshot of JMP Genomics 5.0 That Highlights the Custom Starter Window, the Genome Browser Displaying Statistical Results along Chromosomes, a View of Correlation Structure across Genetic Markers, and Hierarchical Clustering Results.

In the top left corner of Figure 1 you might notice that the current view on the JMP Genomics starter (a window generated with a script launched by the JMP Genomics add-in) is on the Predictive Modeling submenu. In addition to genomic-specific analysis tools, a full set of predictive modeling and cross validation routines is available in the software. Each of the processes runs the corresponding SAS procedure to build models for prediction. Figure 2 is an example of the dialog that surfaces the option to control the parameters passed to the SAS procedure. In Figure 2, the dialog shown has a parameter option called Mode. This highlights a cool feature in JMP Genomics where you can choose to run the analysis natively in JMP if the platform is available. In this case, instead of connecting to SAS, you could launch the JMP interactive partition platform.


Partition Trees - JMP Pro

File Edit Tables Rows Cols DOE Analyze Graph Clinical Genomics Tools Add-Ins View Window Help

Description

Partition Trees recursively partitions data according to optimal splitting More...

General | Predictor Reduction 1 | Predictor Reduction 2 | Analysis | Advanced | Options

Type of Dependent Variable: Nominal

Mode
- Interactive
- [✓ Automated]

Prior Probabilities / Prevalences
- Equal
- [✓ Proportional]

Automated Model Type
- Standard
- Rule Mix
- [✓ Boosted]
- Forest

Maximum Depth of Tree

Minimum Number of Observations Required for a Branch

Minimum Number of Observations Required for a Categorical Value

[✓ Regard Missing Values as Valid for Prediction]

* Required Parameter

Figure 2. The Analysis tab for the JMP Genomics Dialog Used to Build a Predictive Model via Partition Trees.

The suite of predictive modeling processes is a powerful feature of JMP Genomics and is a shared feature with JMP Clinical. If JMP Genomics is the “mother” of JMP add-ins, then JMP Clinical, which was released as a new product in the summer of 2010, could be thought of as her first born. JMP Clinical follows the same integrated architecture of combining visual exploration of JMP with the analytics of SAS and shares a large code base of SAS utility and analysis macros with JMP Genomics.

JMP Clinical is another vertical product offering from SAS that is quickly becoming the premier tool used by medical reviewers, clinicians, and biostatisticians for analyzing safety data in clinical trials. The product relies on data standards as defined by the CDISC data model for evaluating drug safety with regard to events, interventions, and findings. A screenshot of JMP Clinical is shown in Figure 3, which highlights the use of the JMP bubble plot platform for visualizing results of lab tests as well as the JMP tree map and volcano plot to screen for adverse events.

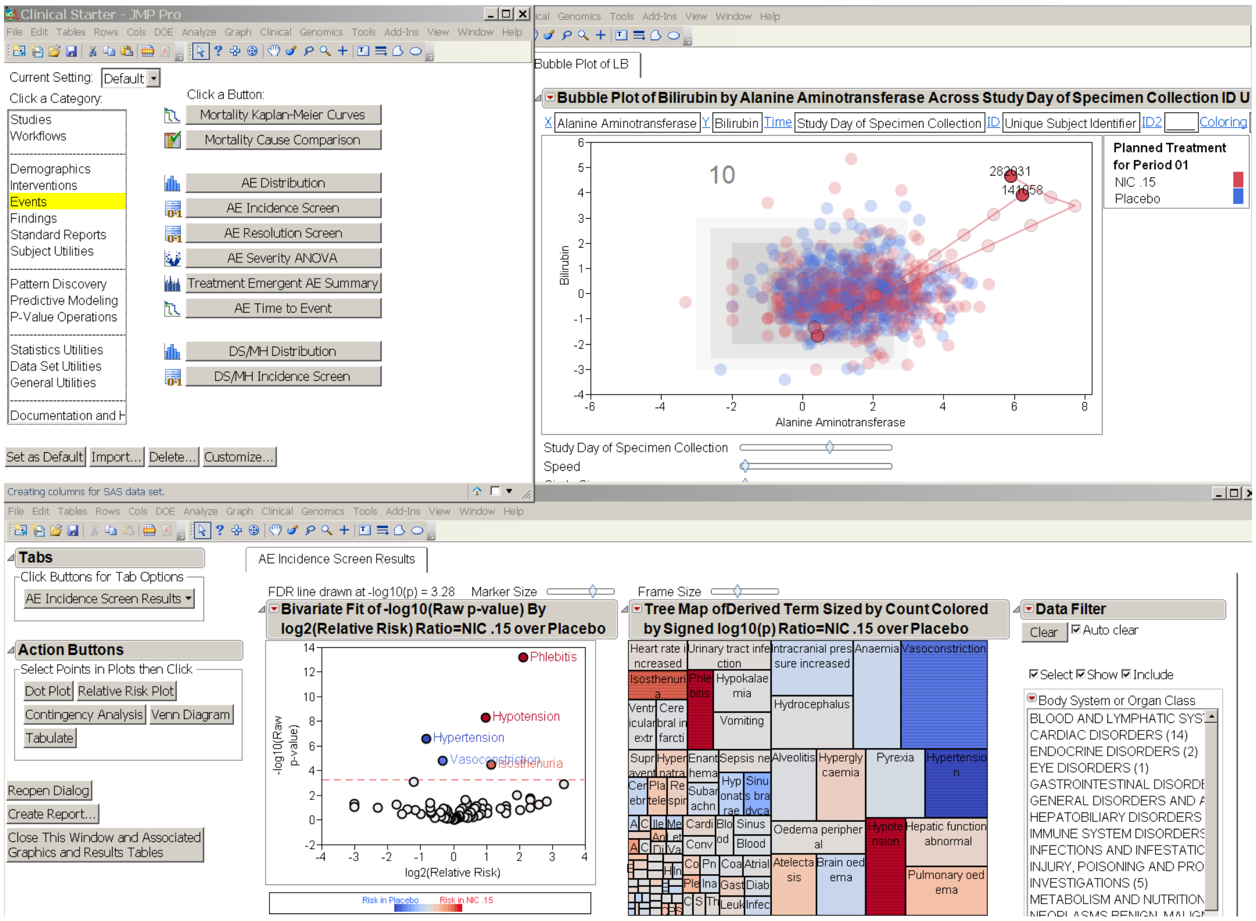
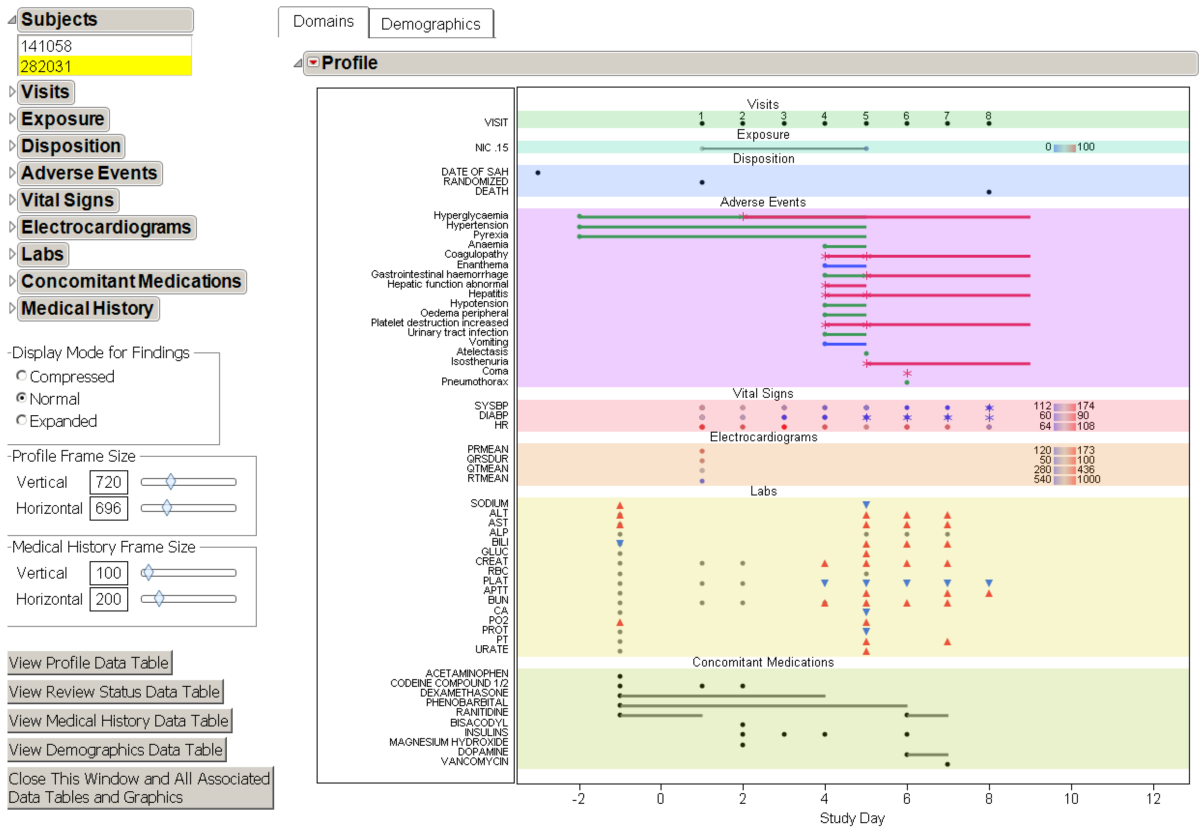


Figure 3. A Screenshot of the JMP Clinical Starter Window, a Bubble Plot Display of Lab Test Results, and a Dashboard Presentation of Screening for Adverse Events in Drug Treatment.

A special feature of JMP Clinical is the subject profiler, shown in Figure 4. This display is a testament to what can be created through JSL scripting. Advanced graphical scripting allows you to truly visualize the data collected to understand how a drug has affected a specific patient. For more information about JMP Clinical, be sure to read the SAS Global Forum paper 201-2011, titled "Driving Clinical Safety Reviews with Data Standards," by the JMP Clinical product manager Geoffrey Mann.

JMP ADD-INS PROVIDE INTERFACES TO SAS COMPONENTS

Other development groups within SAS are learning to harness the power of JMP to provide easy-to-use interfaces and dynamic results to SAS analytical processes. The SAS/ETS component is one such example. While it will not be distributed as a formal bundle product, in the future SAS customers who buy the SAS/ETS component will also receive a JMP add-in file. If they happen to also buy JMP, the add-in can be installed to offer custom interfaces to SAS/ETS procedures such as PROC AUTOREG, PROC PANEL, and PROC UCM to be used for dynamic, interactive analysis of time series and econometric data. This add-in has been developed so that the dialogs for entering variables and parameters to send to the procedure and the output have the look and feel of any other native JMP platform. Familiar constructs such as the JMP red triangle have customized options for producing certain SAS output, building new models from the procedure, and comparing models that have been built.



Display Mode for Findings
Compressed [✓ Normal] Expanded

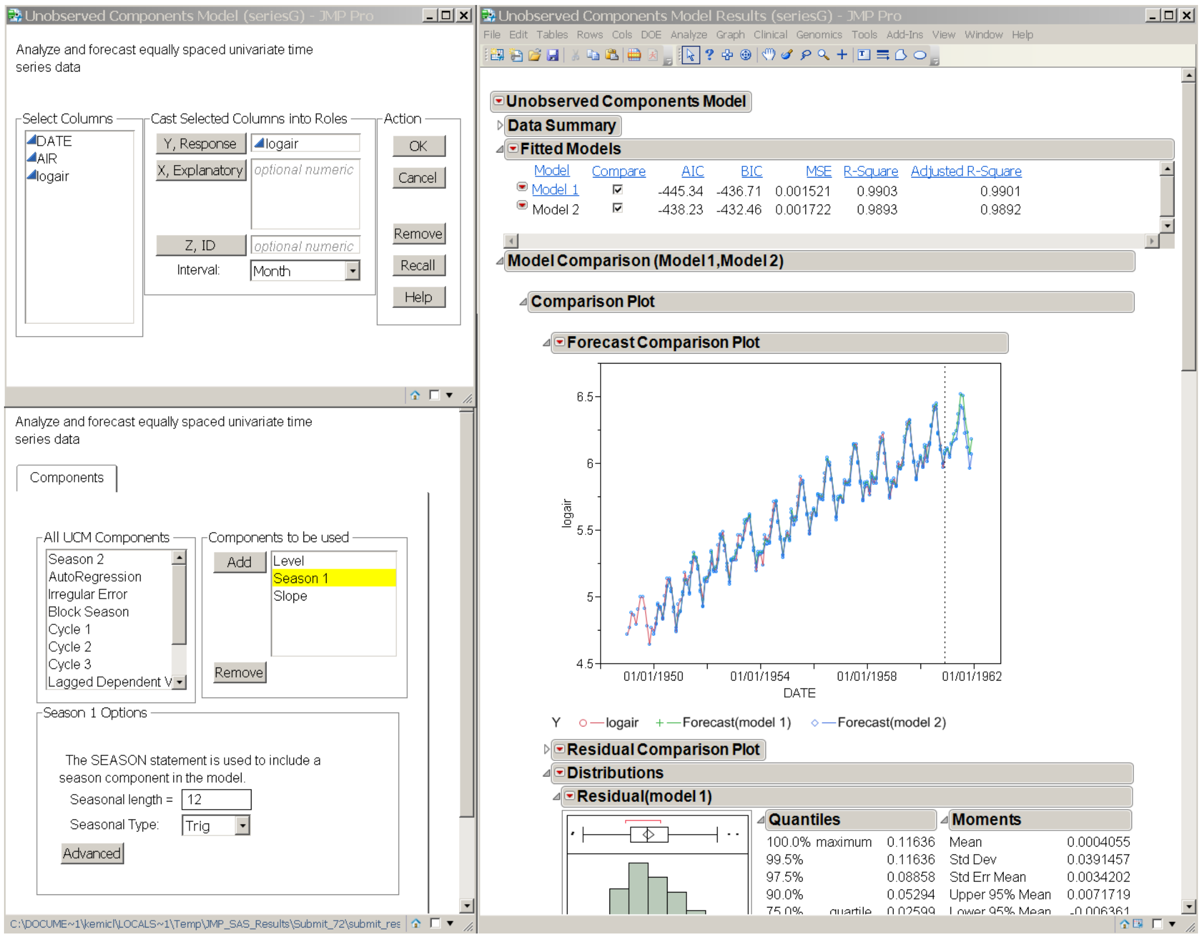
Profile Frame Size
Vertical 720
Horizontal 696

Medical History Frame Size
Vertical 100
Horizontal 200

Figure 4. The JMP Clinical Subject Profiler Is an Impressive Example of Custom JSL Scripting to Produce a Dynamic and Interactive Display of Clinical Trials Safety Data.

The SAS/ETS components offer a suite of advanced procedures for analyzing econometric data, time series forecasting, and financial modeling. These procedures offer analysis options through dozens of procedure statements that are so in depth that it can become daunting for a business analyst to apply these methods via the SAS system. For example, the UCM procedure offers 18 different statement options for specifying a time series model, some of which are for advanced specialized analyses such as BLOCKSEASON, FORECAST, IRREGULAR, RANDOMREG, and SPLINEREG, to name a few. The UCM procedure fits an unobserved components model (or structural model) for analysis and forecasting of univariate time series data. This is done by decomposing the observed series data into components corresponding to trend, seasons, cycles, and regression effects due to other predictors, which allows you to model and predict behavior from data containing complex patterns due to multiple sources.

The application of the procedure can clearly become complex. Enter the JMP add-in to provide an interface to PROC UCM. The initial dialog for fitting the unobserved components model starts simple, where all you need to enter is the response, any covariates, and the ID/time variable. An initial model is built with default trend and seasonal components as well as any covariates. When you click **OK**, JMP connects to SAS and sends the data and commands for the analysis. The results are brought to JMP in the form of any other native JMP platform, with some added bells and whistles under the red triangles to build further models, produce the SAS ODS graphics from the procedure, and compare models. The screenshot in Figure 5 shows the initial dialog in the top left corner, and the results to the right. Models can be added from the red triangle that will launch the dialog shown in the bottom left of Figure 5. From this dialog, you can choose from the available statements to build various models to fit the data. Advance options pertaining to the UCM component chosen are surfaced and can be set or changed.



Unobserved Components Model (seriesG) - JMP Pro

Analyze and forecast equally spaced univariate time series data

Select Columns: DATE, AIR, logair

Cast Selected Columns into Roles:
- Y, Response: logair
- X, Explanatory: optional numeric
- Z, ID: optional numeric
- Interval: Month

Action: OK, Cancel, Remove, Recall, Help

Analyze and forecast equally spaced univariate time series data

Components

All UCM Components: Season 2, AutoRegression, Irregular Error, Block Season, Cycle 1, Cycle 2, Cycle 3, Lagged Dependent V

Components to be used: Level, Season 1, Slope

Season 1 Options

The SEASON statement is used to include a season component in the model.

Seasonal length = 12

Seasonal Type: Trig

Advanced

C:\DOCUME~1\kemicl\LOCALS~1\Temp\JMP_SAS_Results\Submit_72\submit_res

Unobserved Components Model Results (seriesG) - JMP Pro

Unobserved Components Model

Data Summary

Fitted Models

Model	Compare	AIC	BIC	MSE	R-Square	Adjusted R-Square
Model 1	☑	-445.34	-436.71	0.001521	0.9903	0.9901
Model 2	☑	-438.23	-432.46	0.001722	0.9893	0.9892

Model Comparison (Model 1,Model 2)

Comparison Plot

Forecast Comparison Plot

Residual Comparison Plot

Distributions

Residual(model 1)

Quantiles

100.0%	maximum	0.11636
99.5%		0.11636
97.5%		0.08858
90.0%		0.05294
75.0%	quartile	0.02599

Moments

Mean	0.0004055
Std Dev	0.0391457
Std Err Mean	0.0034202
Upper 95% Mean	0.0071719
Lower 95% Mean	-0.006361

Figure 5. A Screenshot of the Dialogs and SAS Output Results in JMP for a Time Series Analysis Using the Add-In to SAS/ETS UCM Procedure.

The benefit of using JMP to enable the time series analysis with the UCM procedure is not limited to simplifying the model fitting and procedure use through a user interface. You also gain from the interactivity of JMP for selecting model outputs to display (by clicking on the model results you want to see under the “Fitted Models” results section) for selecting models that you want to compare results for, and for modifying (adding, deleting, changing) models that have already been fit. By clicking on the column names, the fitted models can be sorted by criteria such as MSE, AIC, and BIC. It can even help you learn more about using the SAS procedures by producing the SAS code used to generate the models as seen below.

```
/*   ============================================================
      The Unobserved Component Model:     Generated SAS Code
    ============================================================ */
ods graphics on;
proc ucm data=work.seriesG plots=all;
id DATE interval=Month;
model logair = ;
level;
slope;
season length=12 type=trig;
estimate;
forecast skipfirst=0 skiplast=0 lead=12 alpha=0.05 outfor=outForecastTable;
run;
ods graphics off;
```

Thus the add-in that combines the power of SAS analytics with JMP not only can get you to your analysis results faster, it also will allow you to assimilate the results and learn more about how to perform future analyses in SAS.

Like the SAS/ETS add-in, other SAS add-ins are available in JMP or will be soon. Add-ins for running loess regression, quantile regression, and robust regression are available to any current user of JMP that also has SAS/STAT. These can be found in the SAS submenu under the JMP File menu. Another comprehensive JMP add-in for structural equation modeling via the SAS/STAT CALIS procedure is also on the horizon. See SAS Global Forum paper 356-2011 by Wayne Watson called “Introducing SAS® Structural Equation Modeling: A New User Interface That Brings the Power of SAS/STAT® Software to JMP® Software” for more information about that particular marriage of JMP and SAS.

CREATION OF CUSTOM ANALYSIS TOOLS VIA SAS AND R

The functionality used to create the products mentioned in the previous sections is available to all JMP users, assuming some knowledge of JSL scripting as well as SAS or R programming. If you are not a programmer, you can still take advantage of custom analyses provided by an add-in that others have developed. Already several JMP Add-Ins are available for download on the JMP file exchange (http://www.jmp.com/community/), many of which use SAS or R to add functionality to JMP. For further information about JMP add-ins, read SAS Global Forum paper 040-2011 by Eric Hill, titled “JMP 9 Add-Ins: Taking Visualization of SAS Data to New Heights.”

An add-in is typically made up of at least three files: an addin.def file that contains the ID of the add-in, an addin.jmpcust file that specifies the JMP menu customization for adding something to the JMP menu, and a JSL file that contains the script that adds the new functionality. The files are packaged as a zip file and the extension needs to be changed from .zip to .jmpaddin. Then, *voilà*, that file is ready to be distributed to allow others to take advantage of a custom analysis tool. To install an add-in to JMP, just double-click or open the .jmpaddin file in JMP. The new JMP add-in architecture, in essence, offers the same extensibility as R packages. Combined with R or SAS integration and tools for building interactive graphical user interfaces and dynamic plots, JMP truly has become a tool that is primed to be used as an analytic hub.

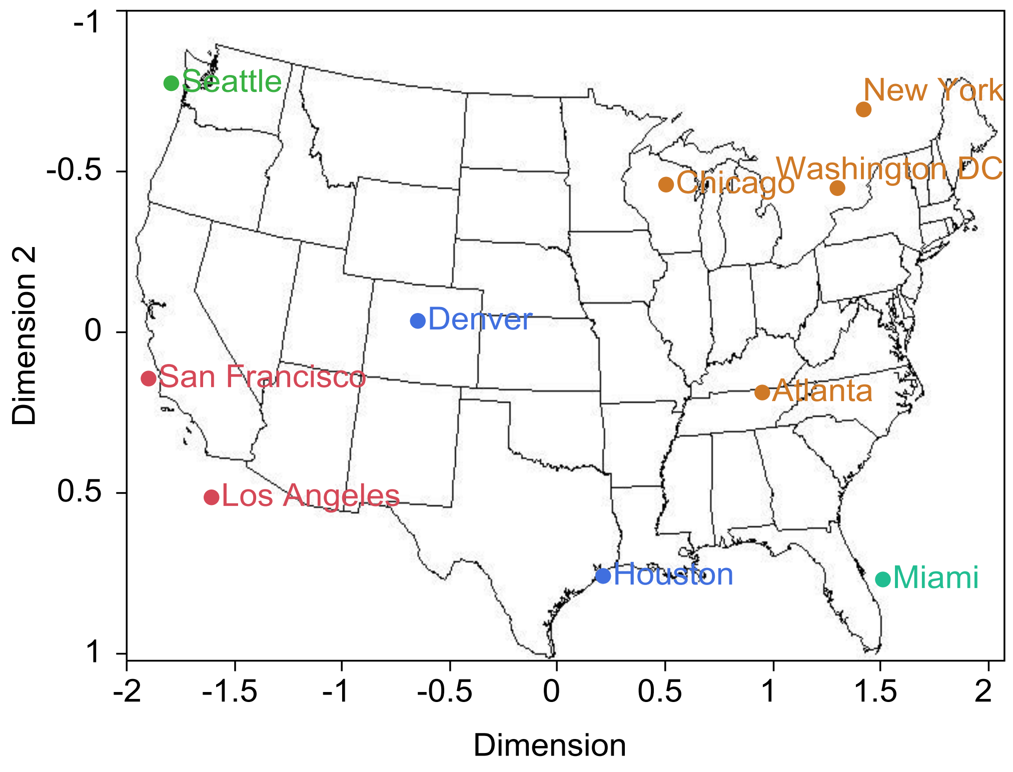
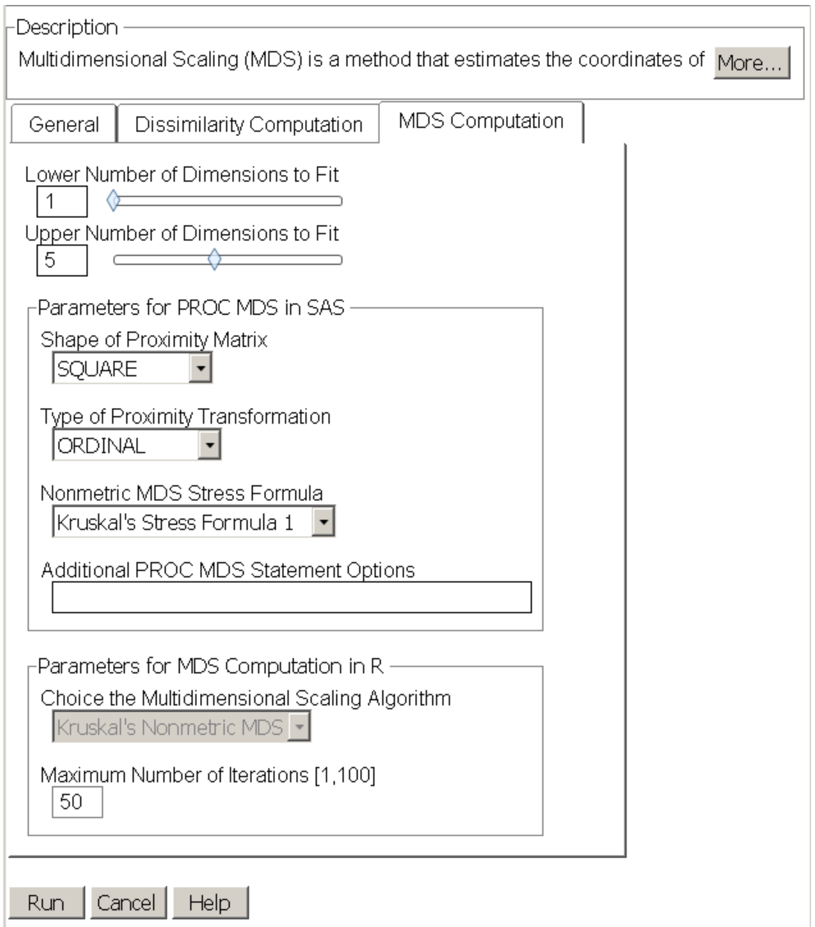


Figure 6. MDS Results in Two-Dimensional Space from Pairwise City Distances in the US.

MULTIDIMENSIONAL SCALING VIA SAS OR R

Multidimensional scaling (MDS) is a multivariate technique for dimension reduction, similar in some ways to principal components analysis. The idea is to take a matrix (of dimensions N by N) that contains pairwise dissimilarities or distances between N objects and try to find a set of (Euclidean) coordinates in typically two or three dimensions that best represents how all the objects are related to each other. The key differences from principal components analysis (PCA) is that MDS will find coordinates that are as close as possible to the original pairwise dissimilarities, whereas PCA decomposes the correlation or covariance matrix to find dimensions that explain maximal variability. Figure 6 highlights this difference by showing the MDS dimensions computed from a matrix containing pairwise distances between major US cities. PCA would have returned a solution with little regard to where the cities actually lie in relation to each other, while MDS produces coordinates that closely match their actual position on a map (as you can see from the overlaid image of the US map in the JMP plot). For this reason, MDS is a popular method used in a variety of fields such as psychology, ecology, and genomics, to name a few.

The plot shown in Figure 6 was made with a multidimensional scaling add-in. MDS is not a statistical method available natively in JMP, but both R and SAS have packages and procedures for it. The add-in generates a dialog where the user can specify the input variables and set parameters for the analysis. One of those parameters is a radio button where you can specify whether SAS or R should be used for computing the distance matrix (if the data are not already formed as one) and MDS dimensions. The portion of the dialog shown in Figure 7 shows how parameters for computing MDS via SAS or R are surfaced to the JMP user. Using JSL, we have set dependencies that if SAS has been chosen for analysis, then only SAS options are enabled (the R options are grayed out).



Multidimensional Scaling (MDS) is a method that estimates the coordinates of

Lower Number of Dimensions to Fit: 1

Upper Number of Dimensions to Fit: 5

Parameters for PROC MDS in SAS

Shape of Proximity Matrix: SQUARE

Type of Proximity Transformation: ORDINAL

Nonmetric MDS Stress Formula: Kruskal's Stress Formula 1

Additional PROC MDS Statement Options:

Parameters for MDS Computation in R

Choice the Multidimensional Scaling Algorithm: Kruskal's Nonmetric MDS

Maximum Number of Iterations [1,100]: 50

Figure 7. A Snapshot of the Tab on the MDS Dialog That Surfaces Options for Computing Multidimensional Scaling Either in SAS or in R.

Once the analysis has been set up on the dialog, clicking the **Run** button will trigger the JSL script that will create R or SAS code and send the data to the tool for computation. The parameter values will be extracted from the dialog, the data will be formatted and sent to the external analysis tool, and the code to perform multidimensional scaling will be executed. Below is a snippet of the JSL commands for running the MDS procedure from the SAS/STAT component.

```
//create SAS code for running PROC MDS;
MDSsascode = "\[
 %macro MDSrun;
    < Code Removed >
    ods graphics on;
    Proc MDS data=&mdsindat DIM=&MDS_Ldim TO &MDS_Udim SHAPE=&ShapeMat
    PFINAL PFIT OCONFIG LEVEL=&sasMDS_Level &mdsfit &sasMDS_Options
    out=mdsoutdims(drop=_label_ ) outfit=MDS_fit;
       var &mdsinvars;
       %if %length(&MatVar_sas) %then %do;
          matrix &MatVar_sas;
       %end;
    run;
    ods graphics off;
    < Code Removed >
 %mend;
 %MDSrun;
]\";

SAS Export Data(MDS_dt, "WORK", "sasMDS_dt", PreserveSASColumnNames(0));
SAS Submit(MDSSAScode, ODSGraphics(1),
 DeclareMacros(
     CopyVars_jmp, CopyVars_sas, Dvars_sas, sasMDS_Shape, sasMDS_Level,
     sasMDS_SForm, MDS_Ldim, MDS_Udim, sasMDS_Options, MatVar_sas
 )
);
stress_dt=SAS Import Data("WORK", "MDS_fit",invisible);
```

The data set is first sent to SAS using the **SAS Export Data()** command, which also allows you to store JMP object values as SAS macro variables to be used in the analysis (these are all objects that were defined by the user in the dialog). SAS code is created and saved in a JSL string, which is submitted to SAS via the **SAS Submit()** JSL function. The call to PROC MDS will produce output SAS data sets, which can be brought back to JMP with the **SAS Import Data()** call.

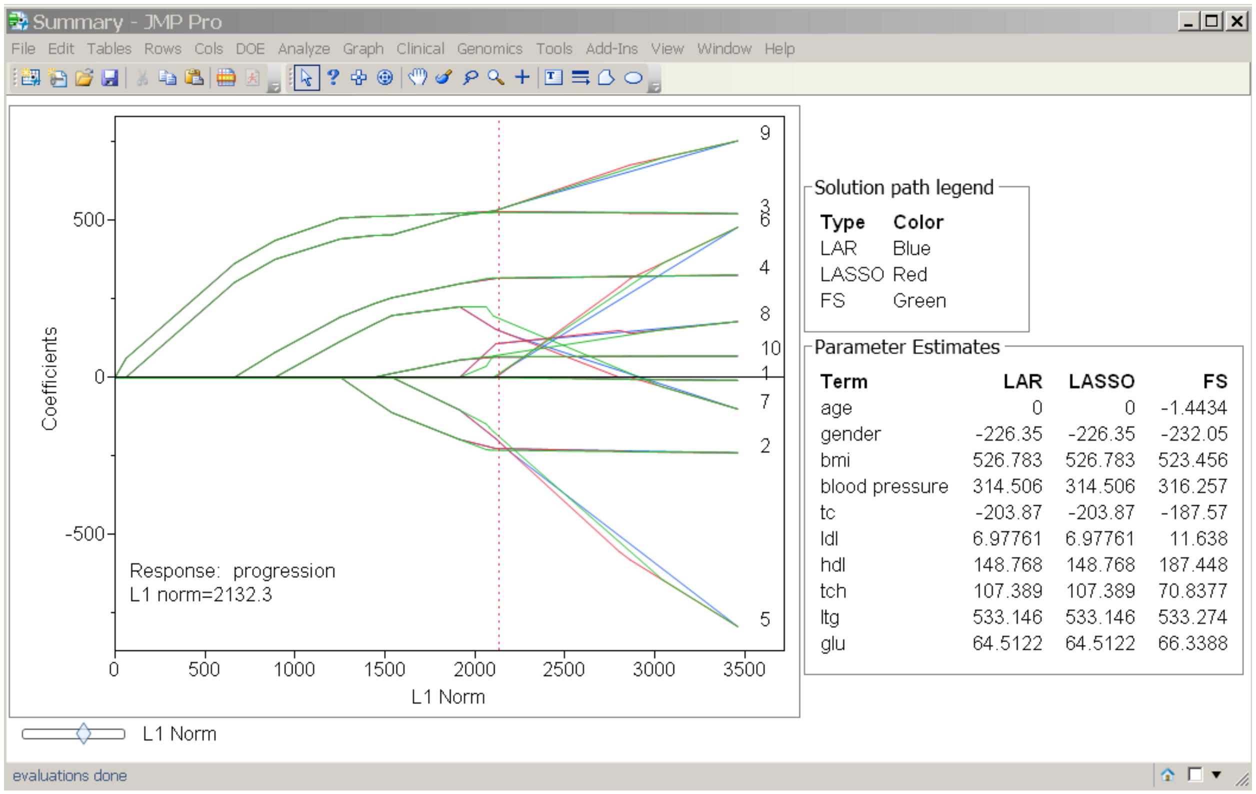
If instead of SAS, you wish to use the R language for MDS computation, the code below highlights the creation of the R commands and JSL functions that allow you to integrate with R.

```
RMDScode = "library(MASS)";
For( i = MDS_Ldim, i <= MDS_Udim, i++,
   j = j + 1;
   RMDScode ||= Eval Insert("\[
      Rnmds.^j^ <- isoMDS(mds.dist, y=cmdscale(mds.dist,^i^), k=^i^, maxit =
      ^MDS_Miter^, tol = 1e-4)
   ]\");
   RGetstmts ||= Eval Insert("\[
      stress_mat[^j^] = R Get(Rnmds.^j^$stress);
      points_mat^i^ = R Get(Rnmds.^j^$points);
   ]\");
);

// submit R code for distance and MDS calculations;
R Init();
R Send(mds_inmat);
R Submit(Rdistcode);                        // Submit R code for distance
calculation;
if(MDS_GetDist ==1, mds_dmat= R Get(mds.dist));   // get back distance matrix;
mdscode = R Submit(RMDScode);                    // do MDS calculatiops;

Eval(Parse(RGetstmts));  // get back stress values and matrices for dimensions;
R Term();
```

For R computation, again a JSL string is created containing R code that will be submitted with the **R Submit()** command. First JMP has to connect to R via the **R Init()** function (note that these JSL commands can also be sent as messages to an object that refers to an R connection) and send the data with **R Send()**. The R objects necessary for displaying the results in JMP are created in JMP with the **R Get()** call (in the example above, this call is also contained in a JSL string that gets parsed to execute the actions). In both snippets of code above, a JMP data table will be produced containing measures of how well the data fit for each computed dimension. From this table, further JSL graphical scripting commands create a scree plot of the goodness-of-fit values. This plot is interactive so that if you select a point corresponding to the number of fitted dimensions, a scripted button box will grab the corresponding data (either from SAS or R) and launch results such as the plot shown in Figure 6. Add-ins like the one described here are available on the JMP file exchange (http://www.jmp.com/community/), although the currently available add-ins for MDS provide a SAS interface for PROC MDS in one add-in and a separate add-in for a dialog to perform R computation of MDS.

MODERN PENALIZED REGRESSION MODELS VIA R

Our last example of building a custom application in JMP is of an add-in (currently available on the JMP file exchange by author Clay Barker) for performing modern penalized regression methods. In regression, it is a common predicament where you need to choose the best selection of predictors from a set of variables. There is a two-fold goal of finding the most meaningful variables while also estimating the coefficients that provide the best prediction, commonly referred to as variable selection and shrinkage. Classic methods such as forward, backward, and stepwise regression are commonly available. Newer, more promising approaches include Least Angle Regression (LAR), Least Absolute Shrinkage and Selection Operator (LASSO), and Elastic Net. Methods such as the LASSO (one of the most popular at the moment) add penalties to the least squares regression solution that can be used as tuning parameters for finding a set of good predictors. The penalized regression add-in contains scripts to create three different dialogs for implementing such methods.



Solution path legend

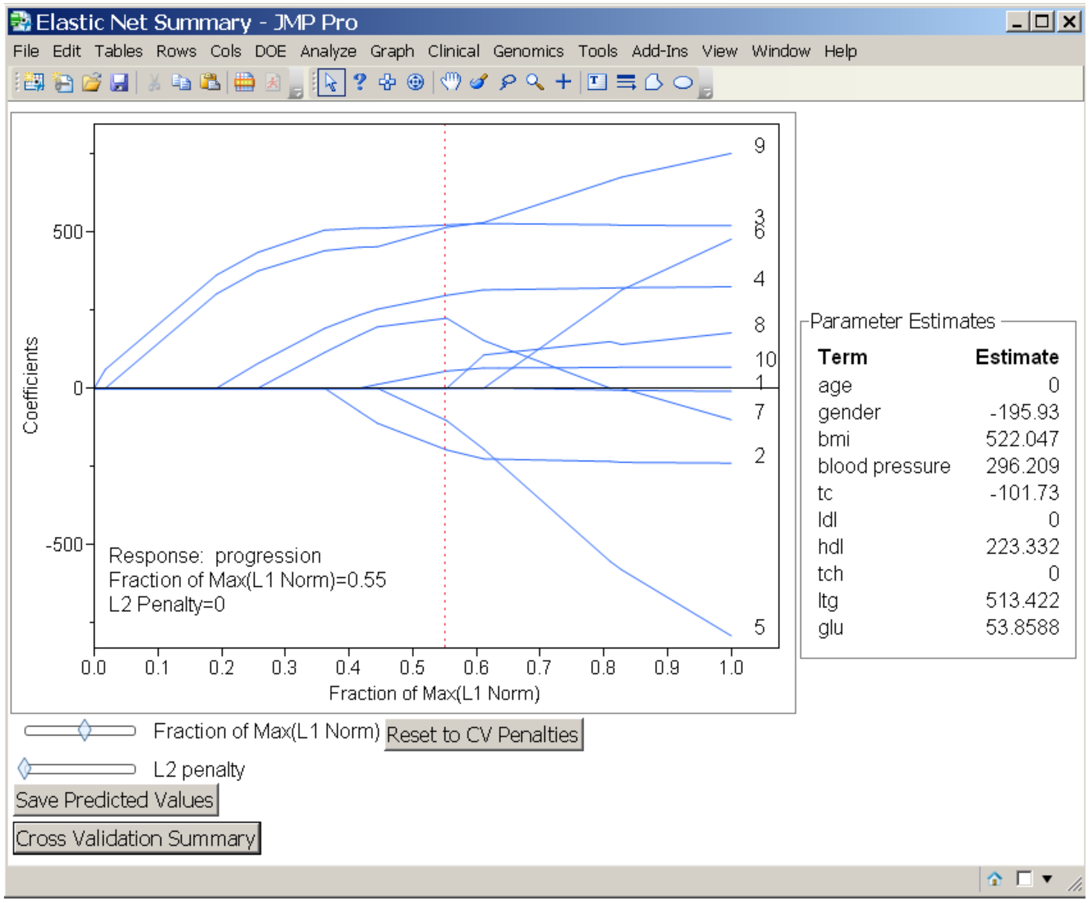
Type	Color
LAR	Blue
LASSO	Red
FS	Green

Parameter Estimates

Term	LAR	LASSO	FS
age	0	0	-1.4434
gender	-226.35	-226.35	-232.05
bmi	526.783	526.783	523.456
blood pressure	314.506	314.506	316.257
tc	-203.87	-203.87	-187.57
ldl	6.97761	6.97761	11.638
hdl	148.768	148.768	187.448
tch	107.389	107.389	70.8377
ltg	533.146	533.146	533.274
glu	64.5122	64.5122	66.3388

Figure 8. Results of the JMP Add-In That Compare Penalized Regression Methods Computed Using R.

Figure 8 shows the results of a custom process for comparing the parameter solution paths for LAR, LASSO, and another similar method called Forward Stagewise regression. Using JMP to implement these R packages provides a unique analysis tool to evaluate how the methods perform and compare. A custom slide bar at the bottom of the plot allows you to change the tuning parameter, which will automatically update the parameter estimates for each method. Through the GLMSELECT procedure in SAS, both LARS and LASSO methods can be implemented; but the elasticnet R package can also perform Elastic Net. Elastic Net is an even more recent method that adds another tuning parameter and has been found to outperform in cases when there are correlated variables. Another script in the add-in creates a dialog for performing Elastic Net with cross validation to find the tuning parameter values that return the most predictive set of variables and estimates. Figure 9 below shows the results of the output for this process.



Response: progression
Fraction of Max(L1 Norm)=0.55
L2 Penalty=0

Parameter Estimates

Term	Estimate
age	0
gender	-195.93
bmi	522.047
blood pressure	296.209
tc	-101.73
ldl	0
hdl	223.332
tch	0
ltg	513.422
glu	53.8588

Figure 9. Results of the Elastic Net JMP Add-In.

One of the most useful scripting features with JMP is how easy it is to create display objects that will execute JSL scripts. In Figure 9, the changing the tuning parameter for the "L2 penalty" requires a new variable solution path to be computed, which means we need to connect back to R. The script for the slider bar that allows you to do that is shown below (portions removed for brevity).

```
Slider Box(low, high, lambda,
   R Init();
   R Send(xmat);
   R Send(yvec);
   R Submit(evalinsert("\[
      library(elasticnet)
      obj1 <- enet(xmat, yvec, lambda = ^lambda^)
      res <- predict.enet(obj1, type="coefficients", mode="step")
   ]\"));
   new_coef = R Get(res$coefficients);
   R Term();
   < Code Removed >
   clty << reshow; p1box << delete; // redraw solution path and parameters;
);
```

The script above, when executed, connects back to R with a new value for the tuning parameter and brings the results back to JMP. By applying the **<<reshow** message, the new solution paths will replace those shown in the graphic. The script attached to the **Save Predicted Values** button box also triggers a connection back to R to use the current model to produce predicted values, which will be attached to the original data set. This code is shown below.

```
// button to save predicted values for a particular combination of tuning
parameters
// involves a connection to R.

Button Box("Save Predicted Values",
   R Init();
   R Send(xmat);
   R Send(yvec);
   R Submit(eval insert("\[
      library(elasticnet)
      obj1 <- enet(xmat, yvec, lambda = ^lambda^)
      pnet <- predict.enet(obj1, type="fit", mode="fraction", newx=xmat, s=^frac^)
   ]\"));
   pred = R Get(pnet$fit);
   R Term();
   pcol = dt << new column("ENET (frac="|| char(round(frac,2)) || ", L2=" ||
   char(round(lambda,2)) || " ) Predicted");
   pcol << set values(pred);
);
```

Along with the script for comparing methods and the Elastic Net interface, the add-in also contains an interface to R to perform the LASSO method with cross validation as well. Through the dialogs built by the penalized regression add-in, any JMP user can easily apply modern variable selection models to build predictive models as if it was a native feature in JMP. Not only can they use these methods, but using JMP adds a level of interactivity that would not be possible in R (or SAS) alone.

CONCLUSION

JMP is a rapidly evolving statistical and visualization tool for the modern analyst. JMP offers a wide array of tested and validated statistical and graphical platforms, a continually evolving scripting language, integration to other analytical tools, and an architecture that allows JMP to be extended to fit your needs. In a data-centric era where a variety of tools can and should be used to accomplish data analysis, JMP is becoming the perfect analytic hub to allow you to make informed, data-driven decisions. Through several examples ranging from full-scale SAS product offerings to individually written custom analytical processes, we have shown that combining JMP with the depth SAS analytics and the breadth of available R packages can provide an extremely powerful tool for any analyst in any application.

REFERENCES

Eric Hill. 2011. “JMP 9 Add-Ins: Taking Visualization of SAS Data to New Heights.” Proceedings of the SAS Global Forum 2011 Conference. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings11/TOC.html.

Wayne Watson. 2011. “Introducing SAS Structural Equation Modeling: A New User Interface That Brings the Power of SAS/STAT Software to JMP Software.” Proceedings of the SAS Global Forum 2011 Conference. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings11/TOC.html.

Geoffrey Mann. 2011. “Driving Clinical Safety Reviews with Data Standards.” Proceedings of the SAS Global Forum 2011 Conference. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings11/TOC.html.

CONTACT INFORMATION

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

Kelci Miclaus
SAS Campus Drive.
SAS Institute Inc.
Cary, NC 27513
Work Phone: (919) 531-2843
Fax: (919) 677-4444
E-mail: Kelci.Miclaus@sas.com