Paper 292-2010

Integrate JMP® with SAS® to create an automated workflow application for analyzing, processing and reporting of 96 well plate format fluorescence microplate reader data

Stanley P. Koprowski, Jr., Lang Peng, Jacquelyn Dwyer,
sanofi-aventis US, Bridgewater, NJ 08807 USA

ABSTRACT

One of the many challenges facing drug discovery and high throughput biology within the pharmaceutical research and development groups is the effective management of repetitive and time consuming tasks. These everyday jobs are often involved in opening files, copying and pasting information from various instrument outputs, such as plate reader instruments. Additionally, integration of various sources of meta data, including, multiple MS Excel workbooks and text files must be completed in order to accomplish additional post processing calculations.   Successfully completing all of the previous steps in the analysis subsequently allow one to interact with their data through guided analysis, and to create statistics and plots, and final study reports. These series of actions are prone to quality errors, force manual reviews of summary facts, double data entry, and may lend to copying the wrong cell, utilize dissimilar unit abbreviations or arise in slightly singular variations of formatting, including font sizes, font type, cell spacing, and document margins of the final study report.

The Molecular and Cellular Toxicology (MCT) group within sanofi aventis US currently produces data, statistical output and reports using primarily manual processes. This paper will discuss the business opportunity for the MCT department and demonstrate the various techniques that were used to develop an automated approach for importing the raw 96 well plate instrument data, how scientists analyze and interact with and at the end of the day generate their final reports. The result is a powerful package that is easy to use, produces high quality output, and can also fulfill the cyclical needs of quantitative data analysis.

The application uses the following technologies: JMP Scripting Language (JSL), SAS data step programming, and MS Visual Basic.

INTRODUCTION

Pharmaceutical companies typically screen many thousands of compounds in high-throughput biology to identify lead compounds, or candidates for further development. High throughput biology makes use of automation equipment in tandem with standard cell biology methods in an attempt to tackle biological questions that are otherwise impossible using normal processes. It utilizes biology and chemistry along with analytics to feed rapid research. The goal is to not lose quality while applying simple bench methods on a large scale. These lead compounds are then subjected to subsequent safety and efficacy testing before reaching approvable status.

The vision of the present work was to extend the usage of software tools our scientists were already familiar with and which we currently licensed while developing a workflow solution that enabled the scientists to be more efficient and effective. Repetitive and time consuming tasks are required to open, copy and paste data from multiple Excel workbooks (text files and various instrument outputs) into worksheets where there is a need to place data into one Excel workbook to complete additional post processing calculations. This process is prone to data quality errors, such as, copying the wrong cell. Once the worksheet is constructed a data report is produced that contains additional descriptive statistics based on derived data. Scientists typically produce slightly different variations of the data report. Further post-processing of the data using various statistical methods is required.

The business opportunity presented by the MCT department was to request an automated approach for copying the specific cell or range of cells from each worksheet, from within an Excel workbook that is contained in a single directory path to a single file. Derive additional parameters, generate descriptive statistics, determine if the data is statistically significant, and produce a final study report in Microsoft Word format. The automated workflow should be able to be run by any scientist using data generated from a variety of instrument outputs.

REQUESTED FEATURES OR REQUIREMENTS

The single most important goal desired by MCT management was a solution that would improve data quality and could be implemented with no additional budget. The key features discussed and implemented in the initial release of the application are detailed in the table below. The requested functionality can be grouped into either general usage requirements or constraints. Constraints can be thought of as anything that limits or impacts how the software is designed.

Type	Description
Feature	Ability to use workflow on any study data
Feature	Ability to use workflow by any scientist
Feature	Ability to parse all Excel workbooks within a user defined directory path
Feature	Ability to parse all text files within a user defined directory path
Feature	Ability to derive data or perform additional calculations based on existing data
Feature	Ability to store output as SAS data set in user selected directory location
Feature	Ability to store protein content output as a SAS data set in user defined location
Feature	Ability to store study metadata output as a SAS data set in user defined location
Feature	Ability to generate specific activity reports as MS Word format
Feature	Ability to perform distribution analysis
Feature	Ability to perform x-y data fit
Feature	Ability to perform one-way analysis of variance (ANOVA)
Feature	Ability to compare means with Control group using Dunnett's Method
Feature	Ability to perform Tukey-Kramer HSD (Honestly Significant Difference) test
Feature	Ability to create final study report in MS Word format
Constraint	Ability to use JMP 7.02 or newer
Constraint	Ability to use SAS 9.1.3 or newer

Table 1 Requested features and constraints

OVERVIEW

Market evaluation has determined that no commercially available solution currently exists to automate the workflow or that would be able to be purchased at no cost. The group currently utilizes both JMP and SAS software in their routine work and therefore a custom in-house solution comprising of SAS 9.1.3 or newer and JMP 7.0.2 or newer was developed.

The basic workflow is depicted in the figure below. The workflow has a number of steps including, launching the application, creation and import of animal design data, creation and import of study design data, creation and import of the well plate design information, importing of B-OX 96 well plate data to determine the reaction rate, statistical analysis, and creation of final report output. A JMP journal forms the backbone of the application and provided the foundation to layer on the various modules required to build an end-to-end solution for importing raw data to final report generation. The paper will attempt to highlight specific areas of the workflow through coding examples and provide the rationale as to why certain design constraints required actions to be performed in SAS. General comparisons using JSL to SAS data steps will be provided to illustrate syntax between the two languages.

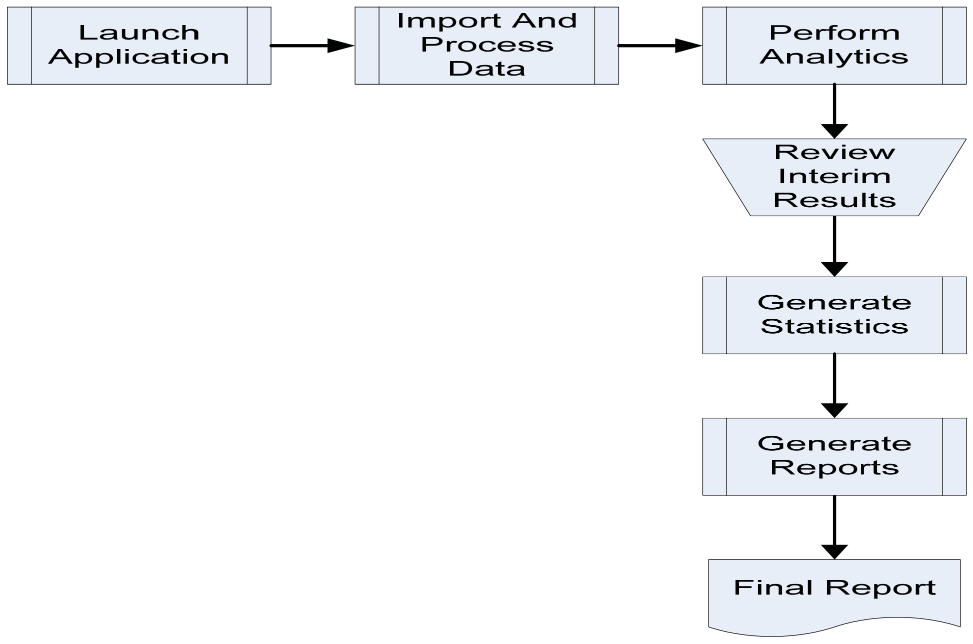


Figure 1 Basic workflow

TECHNOLOGIES USED

The application employed a number of software packages including JMP Scripting Language, SAS data step programming, and MS Visual Basic. The software application and versions used to create the MCT β-PEROXISOMAL ANALYSIS application are detailed in the table below.

Software and Version	Design Usage
JMP 7.0.2	Analytics, User Interface, and workflow control
Microsoft Visual Basic 6.3	Reporting
MS Excel 2003 SP2	Raw data files
SAS 9.1.3	Data import, Analytics, and Reporting
MS Word 2003	Final Study Report
Adobe Acrobat Standard 9.1.3	User guide

Table 2 Application software and version and corresponding design use

CREATING CUSTOM MENUS IN JMP®

The application implemented a variety of custom menus, including a menu for organizing the workflows into a centralized location and a Help menu (see figure 2 and figure 3 below) for providing additional details on the usage of each of the workflow applications. The advantages that the menus provided in terms of general understanding of required data specifications, usage of the application, and a glossary was worth the additional effort to create. Creating the menu from within JMP is straight forward. The basic process is to add the new menu item and to tell the menu item what action to perform.

Begin by selecting **Edit** ⇨ **Customize** ⇨ Menus and Toolbars. This will open a new browser allowing you to manage the menu and toolbar definitions. Clicking on the ⊞ icon expands the selection. A right-click on the well plate layout allows you the option to Insert Before or Insert After the currently selected object. After creating menu options you need to program the menu item so that it can actually perform an action. A right-click on the newly inserted object in the Menu browser will open the properties window for that item.

In this example, the menu was programmed to open a PDF document stored on a remote file share. The JSL OPEN Operator OPEN( ) is used as follows:

```
Open ("\\File location\Name of file to open");
```

Additional options can be included for example, to limit the display or to filter only certain file type extensions. The basic syntax for filtering the list is:

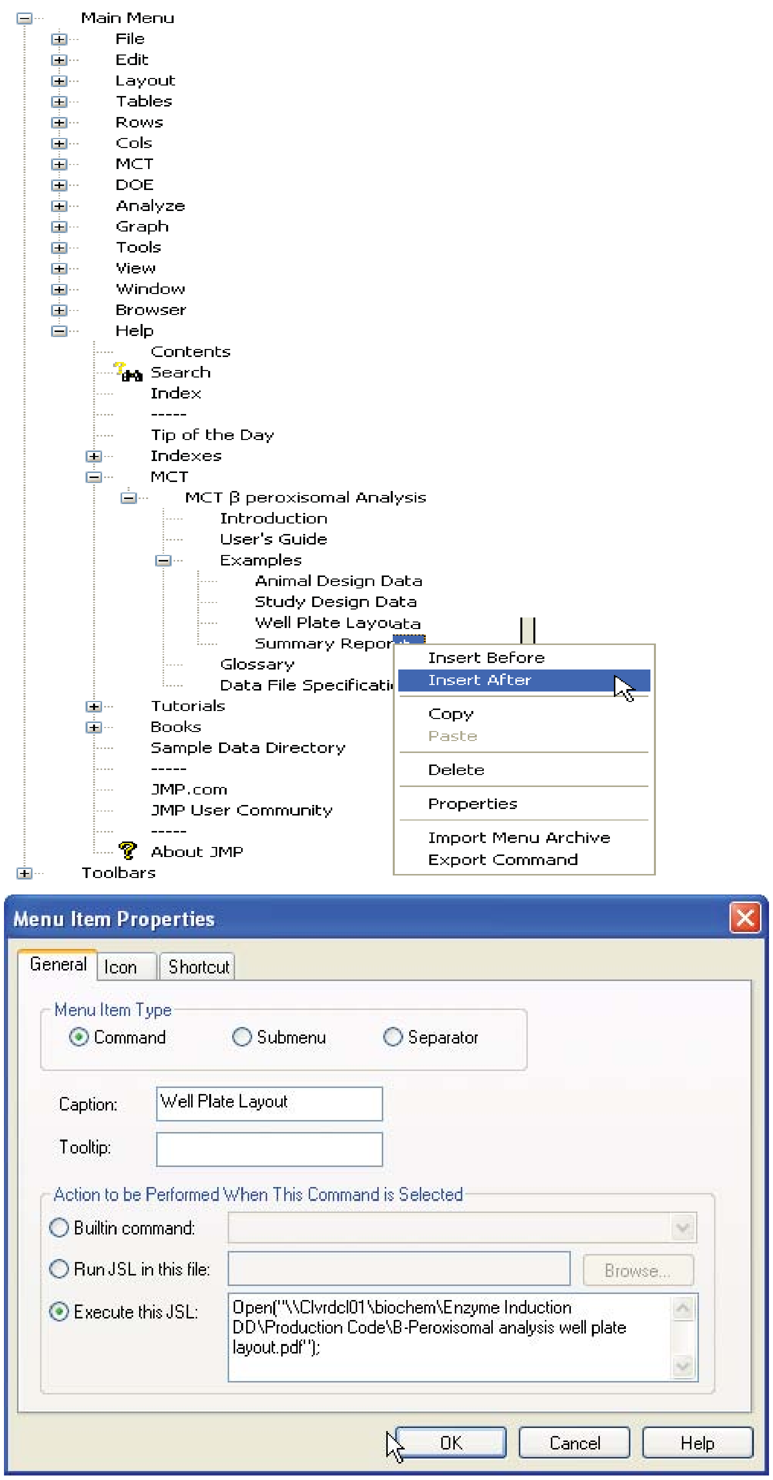
HOT TIP

```
{"Label1|suffix1; suffix2; suffix3", "Label2|suffix4; suffix5"}.
```

To display only PDF and JMP journal files include the following code:

```
{"PDF or JMP Files | pdf;jrn"} or

{"PDF | pdf", "JMP Files | jrn"}
```



```
Main Menu
    File
    Edit
    Layout
    Tables
    Rows
    Cols
    MCT
    DOE
    Analyze
    Graph
    Tools
    View
    Window
    Browser
    Help
        Contents
        Search
        Index
        -----
        Tip of the Day
        Indexes
        MCT
            MCT β peroxisomal Analysis
                Introduction
                User's Guide
                Examples
                    Animal Design Data
                    Study Design Data
                    Well Plate Layouata
                    Summary Repor
                Glossary
                Data File Specificati
        Tutorials
        Books
        Sample Data Directory
        -----
        JMP.com
        JMP User Community
        -----
        About JMP
Toolbars
```

Menu Item Type

[✓ Command] Submenu Separator

Caption: Well Plate Layout

Tooltip:

Action to be Performed When This Command is Selected

Builtin command:

Run JSL in this file:

[✓ Execute this JSL:]

```
Open("\\Clvrdcl01\biochem\Enzyme Induction
DD\Production Code\B-Peroxisomal analysis well plate
layout.pdf");
```

Figure 2 Customizing the JMP menu and editing the menu item properties

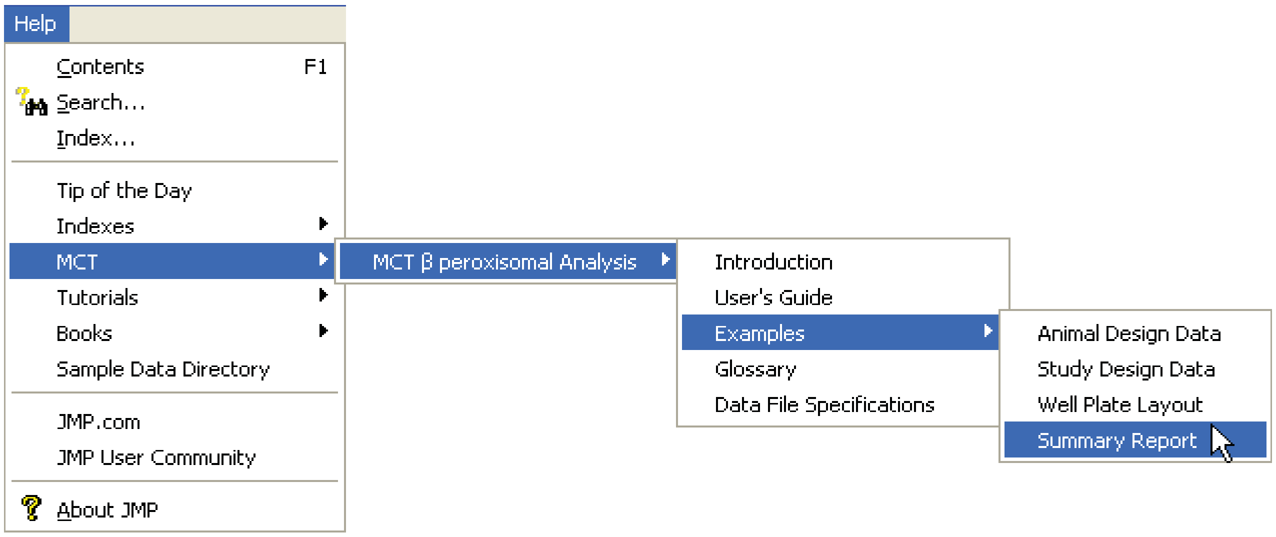


Figure 3 Accessing the applications help menu from within JMP

HOW DO WE COMMUNICATE?

Communication by the application was important because it allowed the developer to work together with the scientist and permitted the integration between JMP and SAS to be transparent to the user. A user refers the intended audience of who might operate the application. How we communicate depends on what or with whom we need to communicate. There were three types of communication implemented in the application are described as USER, SAS and JMP. User or interactive communication occurs between the JMP application and the individual users who are running it. SAS communication takes place by either passing in succession to SAS or returning consecutively from SAS. An example of passing information to SAS is a macro variable parameter or libname statement. JMP communication can be in response to a user action or just simply an update as to what is happening during a particular step in the workflow.

COMMUNICATING WITH THE USER

To communicate with the user a caption window or a custom dialog box was used. A caption window can be used to notify the user that an action is in-process or when an action has been completed. Use the JMP Caption ( ) operator to present an on-screen message for the user.

Line 1 instructs JMP to create a caption window
Line 2 instructs JMP where to place the caption window on the screen based on the X-Y coordinates
Line 3 is the message to be displayed in the caption window
Line 4 instructs JMP to not wait before continuing to next caption, if existing
Line 5 instructs JMP on whether or not to speak the message (0: don't speak, 1: speak)
Line 6 completes the caption window operator
Line 7 instructs JMP to wait for set time, in this example 3 seconds before continuing with execution of script
Line 8 instructs JMP to remove the displayed message from the list

```
Caption(
{620, 100},
"SAVE data directory location successfully defined!",
delayed( 0 ),
spoken( 0 )
);
Wait( 3 );
Caption( remove );
```

SAS® COMMUNICATION--CALLING SAS® FROM JMP®

To call SAS from JMP use the following syntax: SAS Connect ( ). Once a connection to the SAS server has been established you can submit your code to instruct SAS on what to do by using SAS Sumbit ( ). In this example, we submit a Char string called NADHRPT with additional options. The additional options instruct JMP to not display the SAS Output window or the SAS Log window. To submit second instructions to SAS insert a semicolon, to let JMP know that there is more to follow. The semicolon in JMP is not a terminator of an action as used within a SAS data step program.

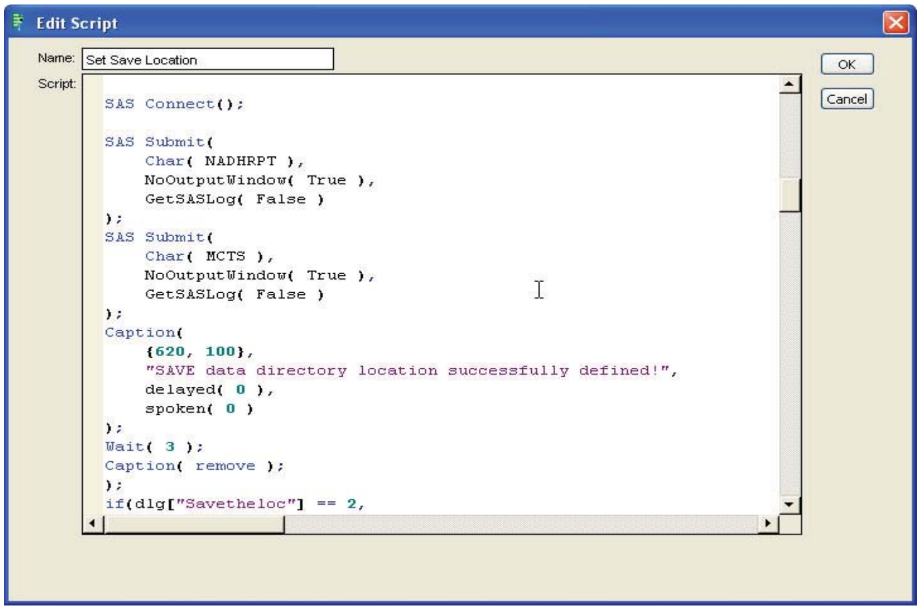
```
SAS Connect( );
SAS Submit(Char( NADHRPT ),
NoOutputWindow( True ),
GetSASLog( False ));
SAS Submit(Char( MCTS ),
NoOutputWindow( True ),
GetSASLog( False ));
```

For a complete list of JSL syntax you can select:

Help ⇨ Indexes ⇨ JSL operators or
Help ⇨ books ⇨ JMP Scritping Guide

CREATING AN ACTION BUTTON IN JMP®

To enable the application to interact with the user custom action buttons were created as shown in the figure below. In this example, the button is given a descriptive name called, Set Save Location. Once the button is named a series of one or more actions are assigned to the button, such as, submitting data step code to SAS or communicating with user.



```
SAS Connect();

SAS Submit(
    Char( NADHRPT ),
    NoOutputWindow( True ),
    GetSASLog( False )
);
SAS Submit(
    Char( MCTS ),
    NoOutputWindow( True ),
    GetSASLog( False )
);
Caption(
    {620, 100},
    "SAVE data directory location successfully defined!",
    delayed( 0 ),
    spoken( 0 )
);
Wait( 3 );
Caption( remove );
);
if(dlg["Savetheloc"] == 2,
```

Figure 4 Configure Action Button

SAS® DATA STEP CODE VERSUS JMP® SCRIPTING LANGUAGE

If you are familiar with SAS data step programming then it would not be too difficult to transition to programming using JSL. Examples of table joins and bar graphs are described below with a comparison of the syntax using SAS data step and JSL. In the first example we will join two tables (table1 and table2). Both tables have a variable in common, commonid1. In the second example a bar graph is created using the mean as the statistic. Parts of the workflow were done in SAS based on necessity others based on familiarity and comfort level with data step programming compared to JSL.

To stack data sets or add observations from one SAS data set to another SAS data set with a common variable using the SAS data step one could use the set option as follows:

```
data combined;
set table1 table2;
by commonid1;
run;
```

To stack or to combine rows of several tables top to bottom in JSL using a table reference one would use either of the following:

```
joinTbl = Data Table( "table1" ) << Join(
With( Data Table( "table2" ) ),
By Matching Columns( :commonid1 = :commonid1 ),
output table name( "combined" ));
```

Or

```
joinTbl = table1 << join(
With( table2), By Matching Columns( :commonid1 = :commonid1 ),
output table name( "combined" ));
```

To create a bar graph using a SAS data step one could use the following code:

```
        title1 "Effect of SGF2010 On The Rate of Knowledge Consumption in IT Professionals
        Treated for Seven Days";
        axis1 label= (f='SANS SERIF/Bold' j=r "Mean Rate (nmol/min/mg)" a=-90)
origin=(30.0,);
        axis2 label= (f='SANS SERIF/Bold'  "Dose Group (mg/kg/day)" ) minor=(number=9)
offset=(,0);
        proc gchart data=table1;
        hbar d / sumvar=commonid1 type = mean errorbar = top clm =95
        maxis=axis1 raxis=axis2 legend=legend1 noframe;
        run;
        quit;
```

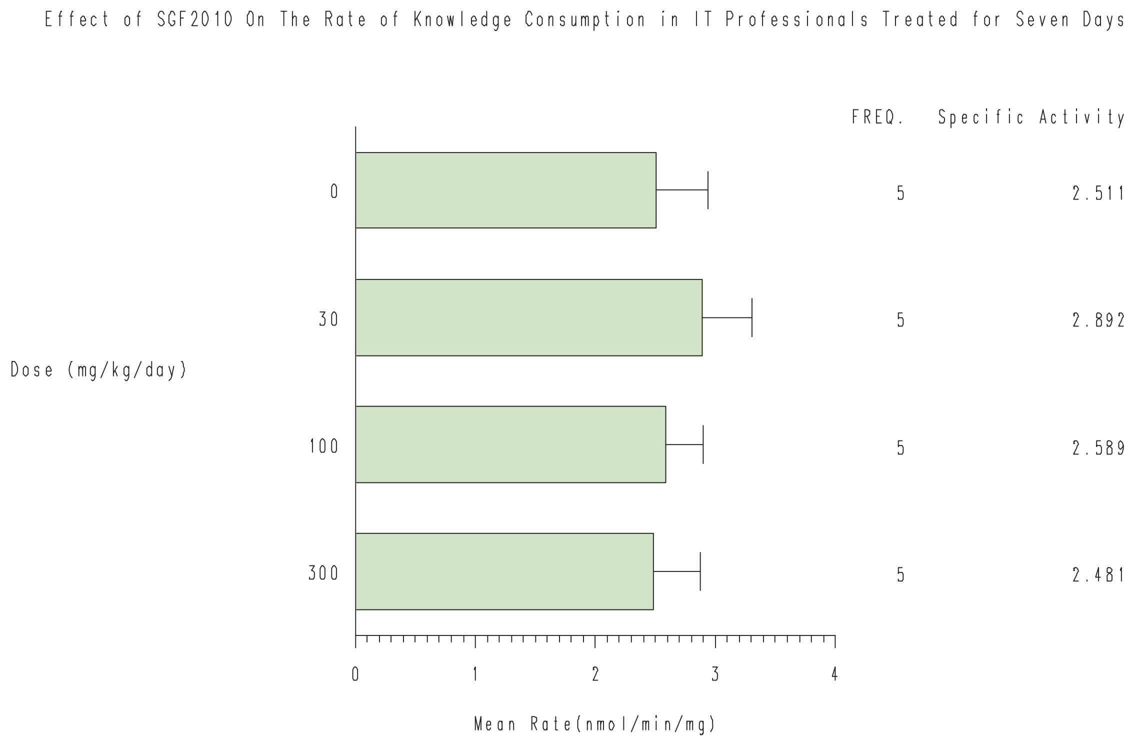
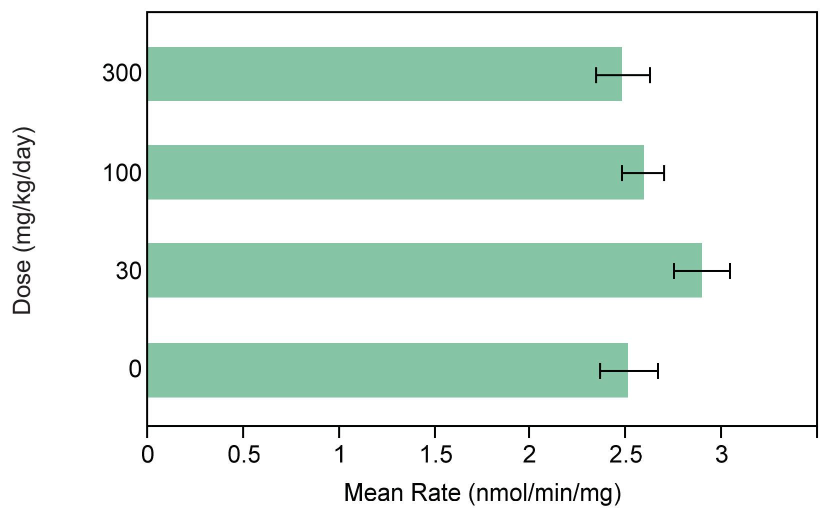


Figure 5 Bar graph output produced from SAS proc gchart

To create a bar graph using JSL use the following syntax:

```
Chart(
X( :d ),
Y( Mean( :Specific Activity ) ),
Horizontal( 1 ),
Category Axis << {Axis Name( Name( "Dose (mg/kg/day)" ) ),
Label( Level( 1 ) )},
Level[1] << Colors( -8832164 ),
Level[2] << Colors( -8832164 ),
Level[3] << Colors( -8832164 ),
Level[4] << Colors( -8832164 ),
Std Error Bars( 1 ),
Bar Chart( 1 ),
SendToReport(
Dispatch( {}, "", AxisBox, Add Axis Label(
"Mean Rate (nmol/min/mg)" ) ),
Dispatch({},"Mean(Specific Activity)",
TextEditBox,
Set Text( "Mean Rate (nmol/min/mg)" )
)
)
);
```

Chart



Each error bar is constructed using 1 standard error from the mean.

ANALYTICS

Analytics included linear regression, distribution analysis, goodness of fit, one-way Analysis of Variance (ANOVA), Means comparisons using Dunnett's and Tukey-Kramer HSD (Honestly Significant Difference). Specific rationale for why each analytics were chosen and interpretation of the results of each analytic is beyond the scope of this paper. Design considerations were given to automate the analytics in order to determine whether or not data was statistically significant. Results produced from the individual analytics were used in either summary tables or custom reports. The review modules provided the analysts with the facility to review raw or untransformed data, calculated or derived data in both tabular and graphical formats. The JSL syntax for creating the ANOVA for the current data table is shown below. Box Plots and Mean Diamonds were excluded through the use of the 0 option.

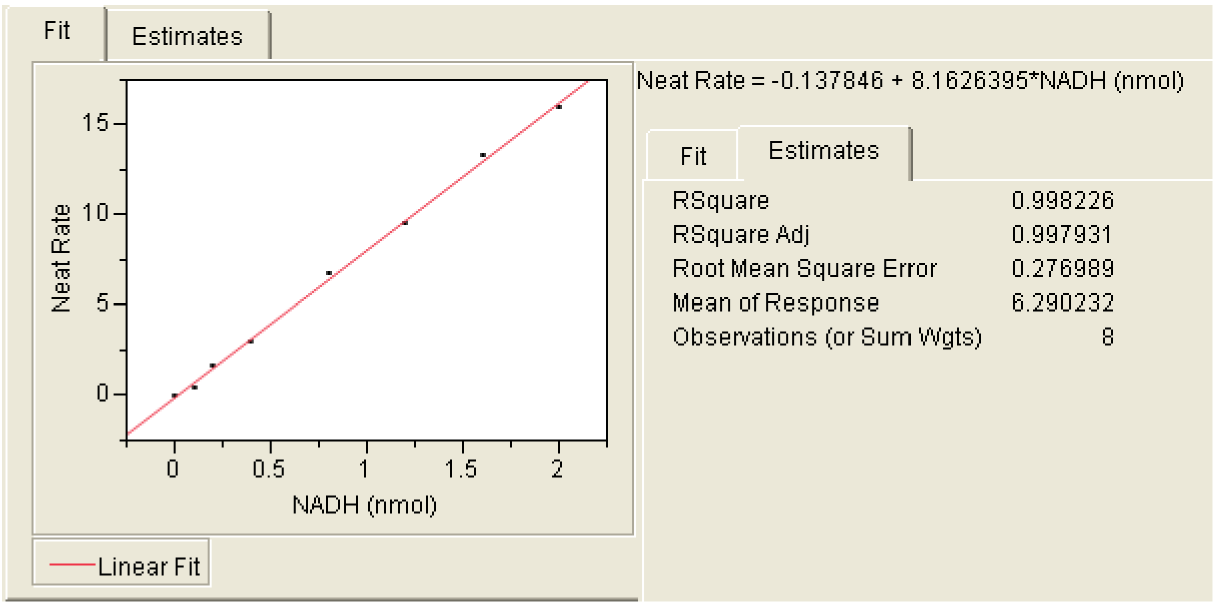
```
ow2 = Send (Current Data Table() ,Oneway
(Y( :Specific Activity ),
X( :Dose ),
UnEqual Variances( 1 ),Box Plots( 0 ),
Mean Diamonds( 0 ),Std Dev Lines( 1 ),
SendToReport(Dispatch( {}, "", NomAxisBox, Rotated Tick Labels( 1 ) ),
Dispatch({"Tests that the Variances are Equal"},"",NomAxisBox,
Rotated Tick Labels( 1 ))));
```

REPORTING

Various reports were created using JSL and SAS's data _null_ report writing. The data _null_ or Output Delivery System (ODS) Object Oriented (OO) approach allowed complete control of report attributes. During this step in the workflow the user has the ability to review the processed data for both individual and summarized data attributes pertaining to quality control samples, standards and unknowns. Summarized data tables included derived values obtained *via* regression analysis, fit estimates, distribution analysis, etc. The final study report is created after all analytics and statistical significance have been determined. The preferred output format of the final study report was Microsoft Word. Because JSL does not provide an out-of-the–box mechanism to produce this type of output SAS was chosen for this task. The approach used combines aspects contained in the SAS ODS OO demonstrated by Daniel O'Connor (O'Connor, SAS Global Forum 2009, 2009), William W. Viergever and Koen Vyverman (Viergever, SUGI28 Proceedings, 2004) and Mark Stetz (Stetz, SUGI25 Proceedings, 2000). The main RTF document is created using ODS OO, details added using DDE and final post-processing formatting using Visual Basic. Partial SAS data step code used for generating the main sections of the RTF report are shown below. Full code syntax details, as well as, other coding aspects of the final study report were omitted from the paper for conciseness.

```
aincount = 0;
do until ((last.group) & (last.animalnumber));
set NADHRPT.&summary end=eof;
by studynumber group animalnumber;
if (first.studynumber = 1) then do;
declare odsout obj();
obj.layout_gridded(width: "7.5in", height: "1in", columns:1, column_gutter: "0.0in",
row_gutter: "0.0in");
obj.region();
obj.head_start();
obj.region();
obj.format_text(text: studynumber, overrides: &overrideH);
—MORE Code—
obj.head_end();
obj.layout_end();
obj.table_start(name: put(ReportTitle, $200.),
 Label: put(studynumber, $20.),
overrides: &overrideT);
obj.row_start(type: "Header", overrides: &overrideN);
obj.format_cell(text: " ",overrides: &overrideH);
obj.format_cell(text: put(NDUnit, $3.) || " " || put(ND, 2.), column_span: 2, overrides:
&overrideB );
obj.row_end();
—MORE Code—
end; /* End First.studynumber */
obj.row_start(overrides: &overrideH);
if aincount not in (1 2) then do;
obj.format_cell(text: " " , overrides: &overrideN);
end; /* aincout ne 1 or 2 */
—MORE Code—
obj.row_end();
aincount + 1;
end; /* End until loop */
if (last.animalnumber=1) then do;
obj.row_start(overrides: &overrideN);
obj.format_cell(text: "Mean:",overrides: &overrideN);
obj.format_cell(text: " ",overrides: &overrideH);
obj.format_cell(text: put(grpmean, 6.3),overrides: &overrideB);
obj.row_end();
—MORE Code—
end; /* end last.animalnumber  */
if eof =1 then do;
obj.table_end();
end;
obj.close_dir();
run;
```

Custom reporting using JSL was used to simplify reporting of the regression analysis by minimizing the need for scrolling through several screen outputs to identify the equation of the line and the R-Square value. The report summarizes the results under two tabs entitled “Fit” and “Estimates” as shown below.



Neat Rate = -0.137846 + 8.1626395*NADH (nmol)

RSquare	0.998226
RSquare Adj	0.997931
Root Mean Square Error	0.276989
Mean of Response	6.290232
Observations (or Sum Wgts)	8

Figure 6 Custom JSL report output window showing Fit and Estimates tabs

The abbreviated syntax used to create the custom JSL report is captured below. The code prompts the user to make a file selection. Once the data table is selected the application uses the Bivariate platform to generate linear regression, equation of the line, and estimates for R-Square, etc. The Clone Box makes a new copy of what is displayed on screen. The variable stdplot is used to store the value of Outline Box (1) [1] so that it can be referenced later. The new window option creates a new window entitled “Standards Summary”. The horizontal list box or H List Box groups display boxes together and arranges them in a horizontal layout.

```
DT3 = open (PICKFILE);
ow = Send (current data table() , Bivariate
( Y( :Neat Rate ),
X( :Name( "NADH (nmol)" ) ),
Fit Line( {Line Color( "Red" )} )));
ob = Report(ow)[Outline Box (1)];
stdplot = ob[Outline Box( 1 )] [1] << Clone Box;
fiteq =ob[Outline Box (2)] [1] [1] << Clone Box;
esttab = ob[Outline Box( 3 )] [1]  << Clone Box;
Send (ow, close window);
nw = New Window ( "Standards Summary",
Tab Box("Fit",
H List Box(stdplot,fiteq),"Estimates",
Table Box(esttab)));
```

CONCLUSION

The workflow application combines presentation quality scientific graphs with comprehensive data fitting. The result is a powerful package that is easy to use, produces stunning output, and can also fulfill the needs of quantitative data analytics. Distinct advantages were realized by combining the user interface capabilities offered out of the box with JMP. The user interface options allowed communication to the user in the form of dialogue boxes and status update messages. Feedback received regarding status update messages was well received. Users indicated that the application was intuitive and easy to use. No more did they have to rely on brute force to transform their data into final study reports. While it was not possible to create Microsoft Word documents directly in JMP, it was possible to do so with SAS. Creation of Microsoft Word documents through SAS ODS OO bridged the gap. The additional post-processing of the RTF document was required possibly because of being used with SAS version 9.1.3 and certain aspects of the SAS ODS OO could not be realized with that version of SAS. SAS ODS OO while currently experimental with the version of SAS we used was quite stable for implementation in the current release of our application. The addition of menus to the application was well received although they did have some maintenance issues. Menus had to be installed on each of the scientist's workstations and if any changes were made to the menus the changes had to be reapplied to each workstation. The software engineering approach taken was to design the application in modules so that value could be realized as each module was test complete. This not only allowed the scientist to simplify their work by using the modules that were completed but provided an opportunity for refinement in a timelier manner.

The application solution provided the following benefits that previously did not exist, including 1) elimination of data transcription errors, 2) solution saved time, 3) allowing scientist to focus time and energy on the data, 4) solution did not require any additional budget for programming or purchasing of new software, and 5) the solution improved overall data consistency and data quality. The application has been tested for compatibility with JMP versions 7.0.2, 8.0.1, and beta 9 releases and to date all coding is forward and backward compatible. Based on the success of this application, further developments around additional workflows are planned for the MCT group. Other departments are currently investigating whether or not the solution as designed might be appropriate for their data analytics and study reports.

REFERENCES

- Cody, Ronald P. and Pass Raymond, SAS Programming by Example, Cary NC: SAS Institute Inc., 1995.

- Gravely, Archer R., Achieving Graphical Excellence with SAS/Graph Software, SUGI24 Proceedings, Cary NC: SAS Institute, 2000.

- Haworth, Lauren E., Output Delivery System: The Basics, Cary, NC: SAS Institute, 2001.

- JMP Software: Introduction to the JMP Scripting Language Course Notes, Cary NC: SAS Institute, 2009.

- Microsoft Corporation, Word 2003 Rich Text Format (RTF) Specification, Version 1.8, http://www.microsoft.com/downloadS/details.aspx?FamilyID=ac57de32-17f0-4b46-9e4e-467ef9bc5540&displaylang=en, 2009.

- O'Connor, Daniel, The Power to Show: Ad Hoc Reporting, Custom Invoices, and Form Letters, SAS Global Forum 2009 Proceedings, Cary NC: SAS Institute, 2009.

- O'Connor, Daniel, Next Generation Data _NULL_ Report Writing Using ODS OO Features, SUGI28 Proceedings, Cary NC: SAS Institute, 2003.

- Parsons, Lori S., Enhancing RTF Output with RTF Control Words and In-Line Formatting, PharmaSUG 2006 Proceedings, Cary NC: SAS Institute, 2006

- SAS Institute, Method Documentation, Appendix 2 Base Software, Cary NC: SAS Institute, http://support.sas.com/rnd/base/datastep/dsobject/Power_to_show_documentation.pdf, 2009.

- SAS Macro Language Reference, First Edition. Cary NC: SAS Institute, 1997.

- Sevick, Carter, "Table 1" In Scientific Manuscripts; Using PROC REPORT and the ODS System, WUSS 2006 Proceedings, Cary NC: SAS Institute, 2006.

- Stetz, Mark, Using SAS® Software and Visual Basic for Applications to Automate Tasks in Microsoft Word: An Alternative to Dynamic Data Exchange, SUGI25 Proceedings, Cary NC: SAS Institute, 2000.

- Tufte, Edward R., The Visual Display of Quantitative Information, Cheshire, Conn: Graphics Press, 1983.

- Usage Note 15179: Executing the SYSTEM function and X Command via the SAS Stored Process Server, Http://Support.SAS.com/kb/15/179.html.

- Viergever, William W. and Vyverman, Koen, Fancy MS Word Reports Made Easy: Harnessing the Power of Dynamic Data Exchange—Against All ODS, Part II--, SUGI28 Proceedings, Cary NC: SAS Institute, 2004.

- Zender, Cynthia L., The Power of TABLE Templates and DATA _NULL_, SUGI30 Proceedings, Cary NC: SAS Institutue, 2005.

ACKNOWLEDGEMENTS

I would like to acknowledge the help and support of Lang Peng and Jackie Dwyer who provided the requirements and testing of the application.

RECOMMENDED READING

- JMP Software: Introduction to the JMP Scripting Language Course Notes, Cary NC: SAS Institute, 2009.
- Cody, Ronald P. and Pass Raymond, SAS Programming by Example, Cary NC: SAS Institute Inc., 1995.

CONTACT INFORMATION

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

Stanley Koprowski

sanofi-aventis US
Mailstop: M-203B
Routes 202-206
Bridgewater, NJ 08807-0800

Work Phone: (908) 541-5251
Fax: (908) 541-5251

Stanley.Koprowski@sanofi-aventis.com

http://www.sanofi-aventis.com