Paper 324-2012

# Using SAS® and LaTeX to Create Documents with Reproducible Results

Tim Arnold and Warren F. Kuhfeld
SAS Institute Inc., Cary, NC

## Abstract

Reproducible research is an increasingly important paradigm, and tools that support it are essential. Documentation for many SAS analytical products has long been created from a single-source system that embeds SAS® code in LaTeX files and generates statistical results from those files. This system is now available to SAS users as an open-source package, which is similar in spirit to Sweave (Leisch 2002) and SASweave (Lenth 2007). The system automatically generates the SAS program file, which includes SAS macros that use the ODS document for capturing the output as external files. Listing and Graphic tags display the captured tabular and graphical output. This paper describes how to access and implement the package, and it illustrates typical usage with several examples.

## Introduction

At the end of a research project, one of the most difficult tasks remains: documentation. The task is especially difficult with computational research because you must ensure that the displayed program code works as expected and exactly produces the displayed output.

Documentation for SAS analytical software is not unlike documentation for computational research. Several years ago, documenting SAS analytics software was a time-consuming and tedious task. Developers maintained two separate sources of their data and code. One set existed in the documentation itself, and the other set was in a separate SAS program file that, when executed, produced the results that were then displayed in the document.

Keeping two separate code sources in agreement is prone to error. When software changes or a new analytical idea comes up, it is easy to make a change in one code set and forget to make the change in the other. For SAS documentation, just as with any research paper, the result of such forgetfulness can have extensive ramifications. Baggerly and Coombes (2009) state in their forensic bioinformatics investigation "Unfortunately, poor documentation can shift from an inconvenience to an active danger when it obscures not just methods but errors."

Because of the consequences of incorrect documentation and the effort needed to maintain two sources of code, the SAS Advanced Analytics Division developed a single-source system in which the code exists in only one location; any changes made to the code automatically appear both in the document and in the executable SAS program that generates the results.

This single-source document system, the StatRep package, is now an open-source software project that you can use for your own research documentation to guarantee that the results you display can easily be reproduced by your readers. The StatRep package is based on the LaTeX typesetting system. You write your paper using both the usual LaTeX markup and the customizations and SAS macros that this package provides. The system reads the code and markup from the single source (your document) and creates an executable SAS program. This automatically generated SAS program produces the results that are displayed in your document.

Comparable projects such as Sweave (Leisch 2002) and SASweave (Lenth 2007) address the problem of reproducibility through the use of a special intermediate language. Although similar in spirit to those systems, StatRep differs in that it is a normal LaTeX package; no special steps are needed to create the LaTeX file or the SAS program. Additionally, StatRep provides both a complete, customizable system for automatic handling of multiple outputs and page breaking and an easy-to-use, flexible method for output selection.

### The LaTeX Documentation System

The StatRep system is founded on the LaTeX documentation system and relies on it to handle the document display. The LaTeX documentation system produces high-quality typesetting and was designed for documents with mathematics. LaTeX is based on Donald E. Knuth's TeX typesetting language, which was first developed in 1985. LaTeX is free and open-source software. The LaTeX system differs from conventional word processors: it provides markup syntax that emphasizes the structure of the document, leaving the main work (writing the content) to the author while the system itself handles the typesetting.

For example, the article you are reading is written using LaTeX markup similar to that shown in the following code:

```
\documentclass{article}
\title{Using SAS and \LaTeX\ to Create Documents
       with Reproducible Results}
\author{Tim Arnold and Warren F. Kuhfeld}

\begin{document}
    \maketitle
    \section{Abstract}
        Reproducible research is an increasingly important ...

    \section{Overview}
        At the end of a research project, one of the most ...
\end{document}
```

The preceding LaTeX markup indicates the document structure: This document is an *article* with a *title*, *author*s, and *section*s.

A LaTeX source file is a plain text file. The pdfLaTeX typesetting engine creates a finished paper by compiling the text file to the PDF format. To learn more about the LaTeX document system and install it yourself, visit the TeX Users Group website. See the “References” section for more information.

### The StatRep LaTeX Package

When you use the StatRep LaTeX package, you follow a four-step process to create an executable document that guarantees that your research results are reproducible:

1. Create your LaTeX source file so it contains your text, data, and SAS code.
2. Compile your document with pdfLaTeX to generate the SAS program.
3. Execute the SAS program to capture your output. For each code block in your document, SAS creates a SAS Output Delivery System (ODS) document that contains the resulting output. For more information about ODS documents, see the *SAS Output Delivery System User’s Guide*. For each output request in your document, SAS replays the specified output objects to external files. All of your requested output is generated and captured when you execute the generated SAS program.
4. Compile your LaTeX document again. In this step, the requested outputs are embedded in the resulting final PDF document.

When you need to make a change in your data or SAS code, you make the change in one place (the LaTeX source file) and repeat steps 2 through 4. Your changes are automatically displayed in your code and in your results. You perform the steps only as needed—when you change your data or code.

You can share your LaTeX source with colleagues and be sure that your results are reproducible. Any SAS user can reproduce your analysis with your LaTeX document and the supplemental files that are described in this paper.

## Install the StatRep Package

To use the StatRep package, you need SAS 9.2 or later, pdfLaTeX 1.30 or later, and the StatRep package itself. You can install the StatRep package by downloading **statrep.zip** from `support.sas.com/StatRepPackage`. The zip file contains the following files:

| | |
|---|---|
| **statrepmanual.pdf** | The *StatRep User’s Guide* provides information about all options used in the StatRep package and the supplied SAS macros file. It also contains several examples that show how you can use the system to capture SAS Log output, the output from SAS procedures that do not support the ODS system, and output from interactive procedures. |
| **statrep_macros.sas** | The SAS macros used by the StatRep package. |
| **statrep.dtx** | The LaTeX package itself. |
| **quickstart.tex** | A template and tutorial sample LaTeX file. |

## How StatRep Works

The StatRep LaTeX package provides two environments and two tags that work together to display your SAS code and results and to generate the SAS program that produces those results. The two environments (**`Datastep`** and **`Sascode`**) display SAS code. The two tags (\**`Listing`** and \**`Graphic`**) display SAS output.

### The Datastep Environment

The purpose of the **`Datastep`** environment is to read in data. It produces no output. Table 1 summarizes the **`Datastep`** environment options.

**Table 1** Commonly Used **`Datastep`** Environment Options

| Option | Action |
|---|---|
| | By default, all lines are displayed and written to the program. |
| **`program`** | All lines are only written to the program. This option is used to produce a data set that does not need to be displayed. |
| **`display`** | All lines are only displayed. This option is used to display code fragments that will not run as is or to display code that is not needed in the generated SAS program. |
| **`first=`***n* | All lines are written to the program, but only the first *n* lines are displayed. This option is used when you have many data lines that do not need to be displayed, but which must be available to the program. |
| **`last=`***m* | When used in conjunction with the **`first=`***n* option, all lines are written to the program, but only the first *n* lines and the last *m* lines are displayed. |

### The Sascode Environment

The purpose of the **`Sascode`** environment is to generate output. Table 2 summarizes the **`Sascode`** environment options.

**Table 2** Commonly Used **`Sascode`** Environment Options

| Option | Action |
|---|---|
| | By default, all lines are displayed and written to the program. |
| **`program`** | All lines are only written to the program. This option is used to execute code that does not need to be displayed. |
| **`display`** | All lines are only displayed. This option is used to display code fragments that will not run as is or to display code that is not needed in the generated SAS program. |
| **`store=`** | Specifies the name of the ODS document to contain the SAS output. All output generated in the environment is stored in the specified ODS document. |

The **`Sascode`** environment also supports a finer degree of control with line-based commands to identify lines that should be only displayed or only passed to the generated program.

Table 3 summarizes the line commands you can use in the **`Sascode`** environment.

**Table 3** `Sascode` Line Commands

| Option | Action |
|---|---|
| `%* program` *n* `;` | The next *n* lines are only written to the program and not displayed. |
| `%* display` *n* `;` | The next *n* lines are only displayed and not written to the program. |
| `%*;` *program line* | The current line is only written to the program and not displayed. |

### The Tags

The `\Listing` and `\Graphic` tags specify the outputs to be displayed. The purpose of the `\Listing` tag is to display tabular output and notes. The purpose of the `\Graphic` tag is to display graphical output.

The `\Listing` and `\Graphic` tags also support several options, which are described in detail in the *StatRep User's Guide*. Table 4 shows the three options most commonly used.

**Table 4** Commonly Used `\Listing` and `\Graphic` Options

| Option | Specifies |
|---|---|
| `store=` | Name of the ODS document that contains the specified ODS objects |
| `objects=` | Space-separated list of ODS objects to capture |
| `caption=` | Caption to be used for the output in the final PDF |

## Using the StatRep package

Suppose you are writing a paper: You want to explain your methods, display your program code, and display and describe your results. This section describes in more detail the four-step process you follow, which was introduced in the section "The StatRep LaTeX Package" on page 2.

### Step One: Create Your Document

Write your document using any type of text editor. You can work with a LaTeX-aware editor such as the TeXworks editor or you can use any plain text editor, such as NotePad or emacs.

Figures 1 through 4 show excerpts from a source file that contains the `Datastep` and `Sascode` environments and the `\Listing` and `\Graphic` tags with which you can write your single-source LaTeX document to display your data, code, and results.

Figure 1 displays an excerpt from a LaTeX file that defines a `Datastep` environment.

**Figure 1** The **Datastep** Environment

. . . descriptive text that introduces data. . .

```
\begin{Datastep}[first=5, last=3]
data Wine;
   input WineType $ VisitLength @@;
   datalines;
white  80 white  98 white 115 white  89 white 103
white  91 white 119 white  31 white 109 white  95
white  71 white 105 white  66 white 141 white  79
white 113 white  69 white 120 white  87 red    93
red    87 red   106 red    76 red   121 red   143
red    81 red    97 red    74 red   107 red   112
red    67 red   107 red    72 red   116 red    99
red   104 red    91 red   132 red    78 red   107
red   101 red    92
;
\end{Datastep}
```

The SAS code in the **Datastep** environment creates a new data set **Wine**. The **first=** and **last=** options specify that only a portion of the data set is displayed.

Figure 2 displays an excerpt that defines a **Sascode** environment.

**Figure 2** The **Sascode** Environment

. . . descriptive text that introduces analysis. . .

```
\begin{Sascode}[store=wineA]
ods graphics on;
proc ttest data=Wine;
   class WineType;
   var VisitLength;
run;
ods graphics off;
\end{Sascode}
```

The SAS code in the **Sascode** environment performs an analysis that uses the TTEST procedure (PROC TTEST). You use the **store=** option so that you can refer to output that is created in the **Sascode** environment later in your document. In this example, all output that is generated by the analysis is stored in the ODS document **wineA**.

Figure 3 displays an excerpt that uses the \**Listing** tag to request output from the ODS document **wineA**.

**Figure 3** The \**Listing** Tag

. . . descriptive text that introduces statistics output. . .

```
\Listing[store=wineA,
   objects=PT Statistics ConfLimits,
   caption={Statistics for Red Vs. White Wine}]{tsta}
```

The \**Listing** tag selects three output objects (**PT**, **Statistics**, and **ConfLimits**) from the **wineA** ODS document.

Finally, Figure 4 displays an excerpt that that uses the \**Graphic** tag to request output from the ODS document **wineA**.

**Figure 4** The \Graphic Tag

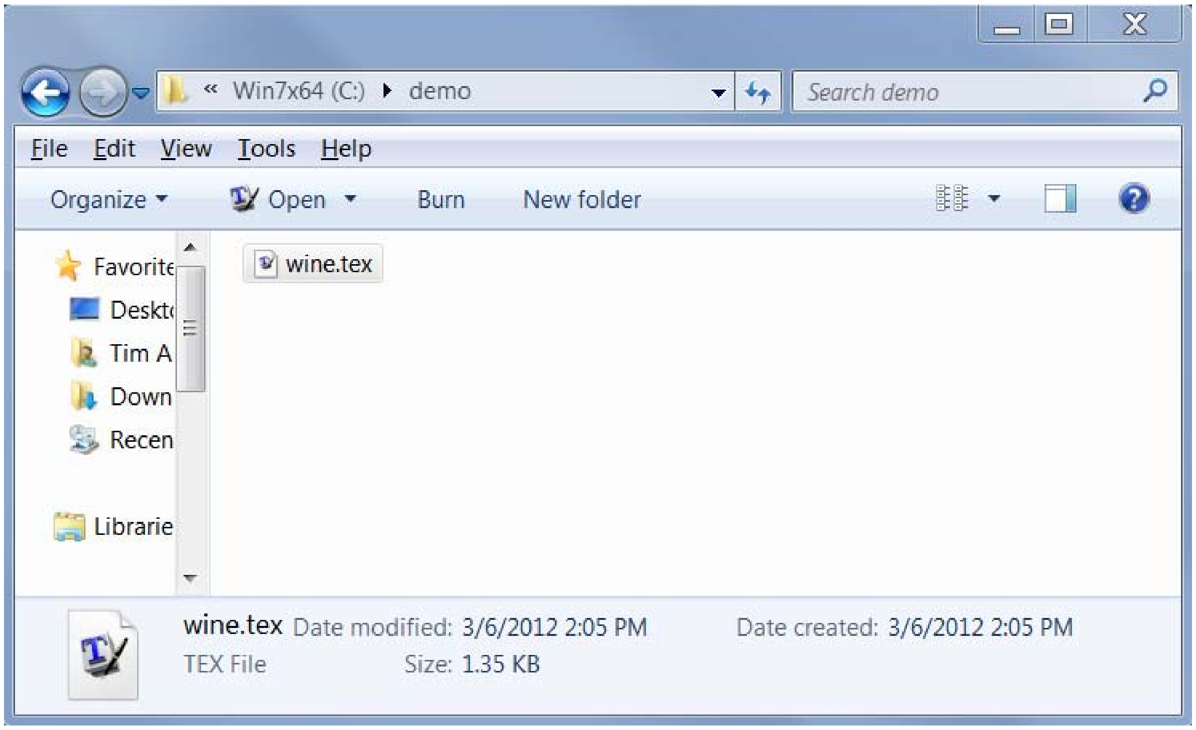
. . . descriptive text introducing graphic. . .

```
\Graphic[store=wineA,
   objects=SummaryPanel,
   caption={Comparative Histograms, Densitites, and Box Plots}]{tstb}
```

The \Graphic tag selects the SummaryPanel graph from the wineA ODS document.

When you save your document, it is stored in your working directory. Figure 5 shows the initial view of your working directory, which contains a single file, named *wine.tex*.

**Figure 5** Initial View of Working Directory

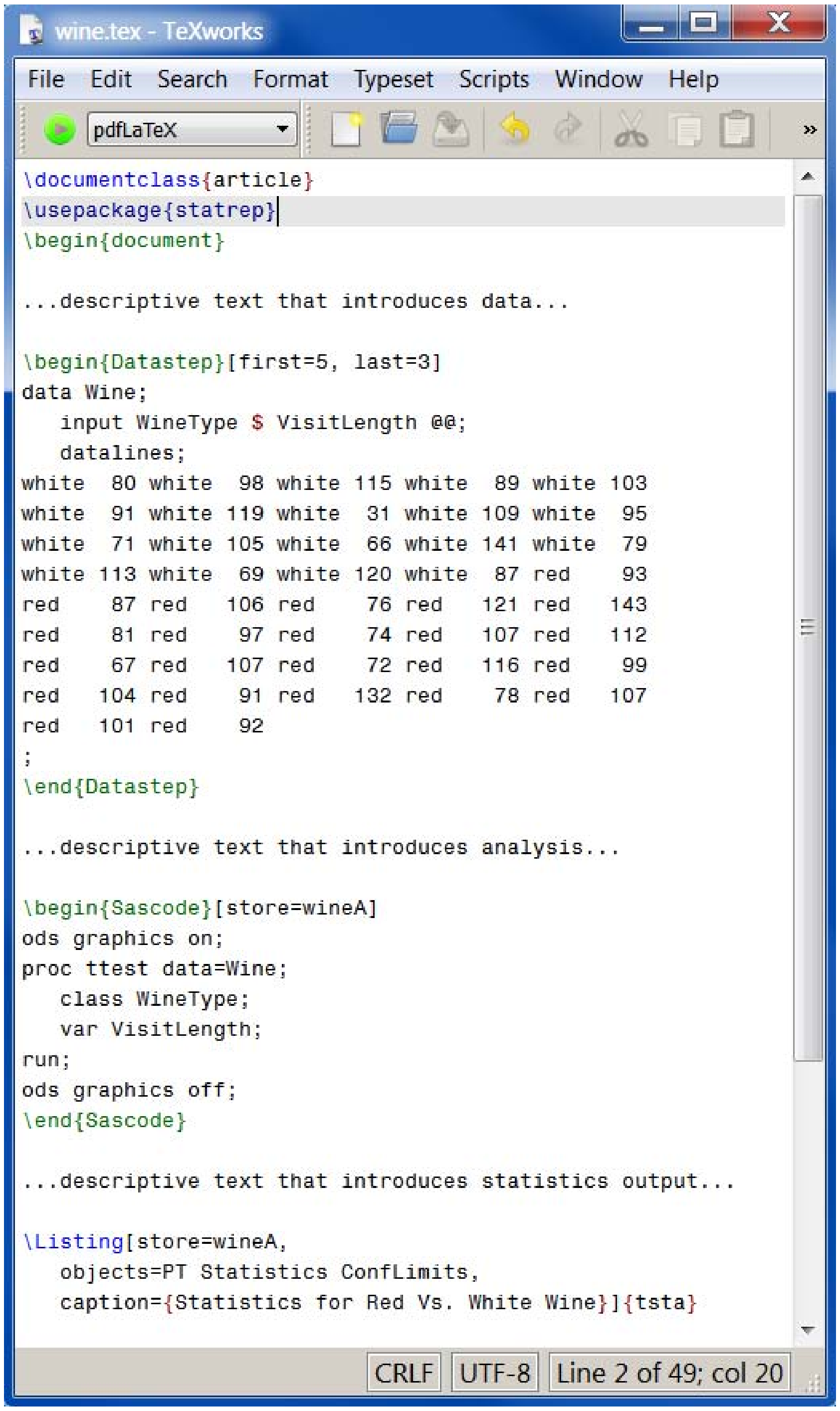


wine.tex Date modified: 3/6/2012 2:05 PM Date created: 3/6/2012 2:05 PM
TEX File Size: 1.35 KB

### Step Two: Compile Your Document

Figure 6 displays how your document appears in the LaTeX-aware editor, TeXworks. You can compile your document by clicking on the green arrow on the menu bar.

**Figure 6** Compiling the First Time with TEXworks


wine.tex - TeXworks
File Edit Search Format Typeset Scripts Window Help
pdfLaTeX
CRLF UTF-8 Line 2 of 49; col 20

```
\documentclass{article}
\usepackage{statrep}
\begin{document}

...descriptive text that introduces data...

\begin{Datastep}[first=5, last=3]
data Wine;
   input WineType $ VisitLength @@;
   datalines;
white  80 white  98 white 115 white  89 white 103
white  91 white 119 white  31 white 109 white  95
white  71 white 105 white  66 white 141 white  79
white 113 white  69 white 120 white  87 red    93
red    87 red   106 red    76 red   121 red   143
red    81 red    97 red    74 red   107 red   112
red    67 red   107 red    72 red   116 red    99
red   104 red    91 red   132 red    78 red   107
red   101 red    92
;
\end{Datastep}

...descriptive text that introduces analysis...

\begin{Sascode}[store=wineA]
ods graphics on;
proc ttest data=Wine;
   class WineType;
   var VisitLength;
run;
ods graphics off;
\end{Sascode}

...descriptive text that introduces statistics output...

\Listing[store=wineA,
   objects=PT Statistics ConfLimits,
   caption={Statistics for Red Vs. White Wine}]{tsta}
```

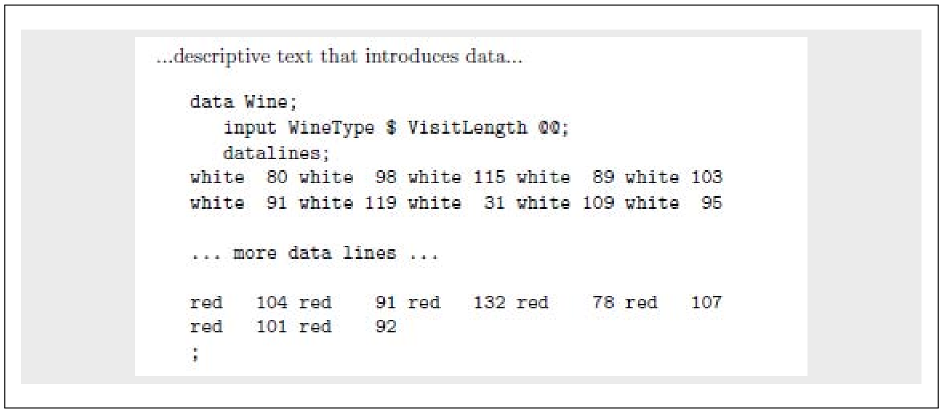
Alternatively, you can compile your LATEX file from the command prompt window or from a terminal window. You give the **pdflatex** command and the name of file to be compiled:

```
pdflatex wine.tex
```

When you compile your document, the StatRep system produces a PDF file and generates a SAS program.

The PDF document that is created in the first compile step displays each **Datastep** and **Sascode** block, as displayed in Figure 7 and Figure 8.

**Figure 7** Resulting PDF after Compiling: The **`Datastep`** Environment

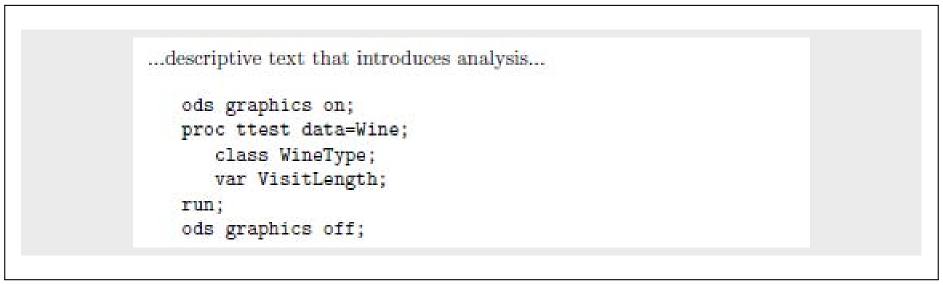


...descriptive text that introduces data...

```
   data Wine;
      input WineType $ VisitLength @@;
      datalines;
   white  80 white  98 white 115 white  89 white 103
   white  91 white 119 white  31 white 109 white  95

   ... more data lines ...

   red   104 red    91 red   132 red    78 red   107
   red   101 red    92
   ;
```

In Figure 7, only a portion of the **`Datastep`** environment is displayed, as specified by the **`first=`** and **`last=`** options in Figure 1.

**Figure 8** Resulting PDF after Compiling: The **`Sascode`** Environment

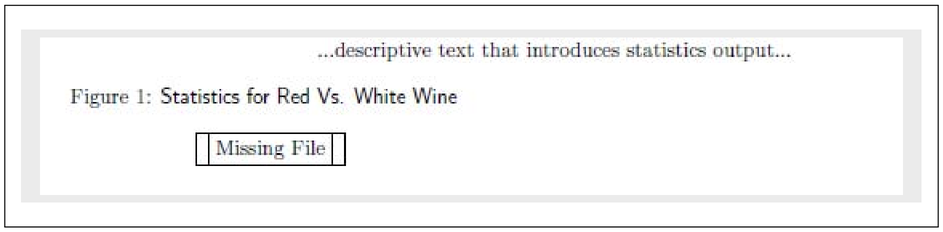


...descriptive text that introduces analysis...

```
   ods graphics on;
   proc ttest data=Wine;
      class WineType;
      var VisitLength;
   run;
   ods graphics off;
```

Figure 9 shows that the requested output is missing after the first compile step. (This is normal at this step.)

**Figure 9** Resulting PDF after Compiling: The \**`Listing`** Tag



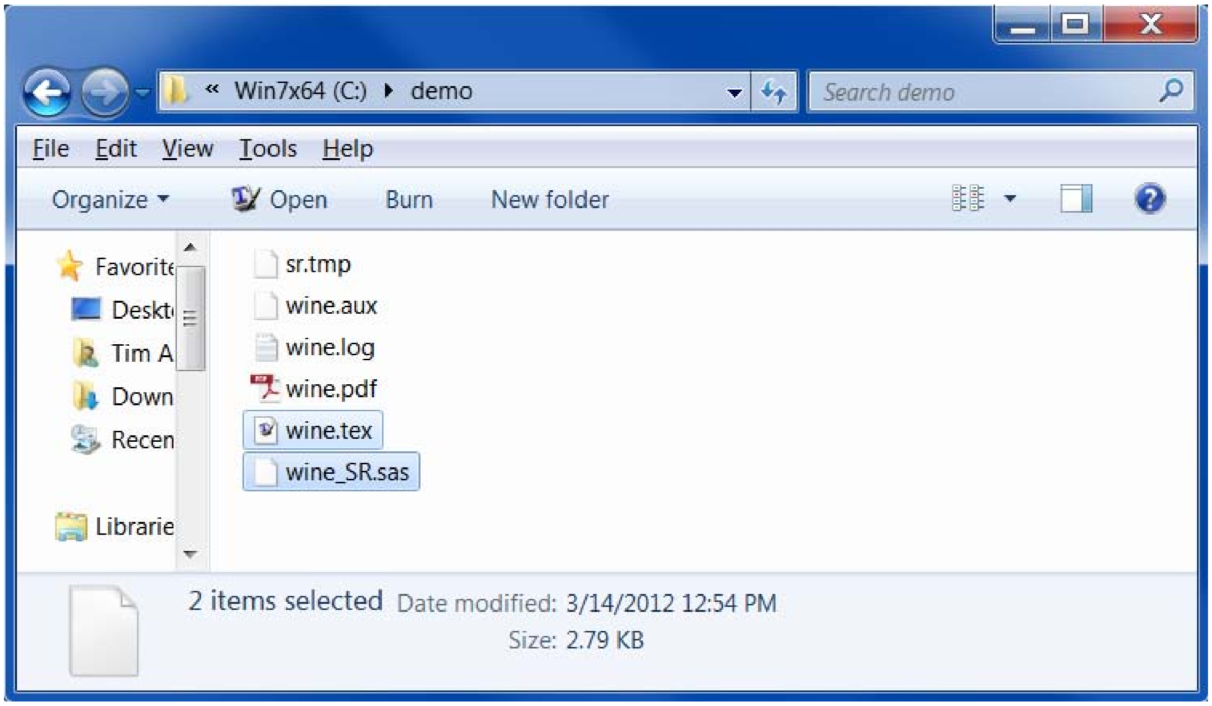
...descriptive text that introduces statistics output...

Figure 1: Statistics for Red Vs. White Wine

Missing File

Figure 10 displays the contents of the working directory after you compile the LaTeX document. The directory now contains the file **`wine_SR.sas`**, which is the generated SAS program.

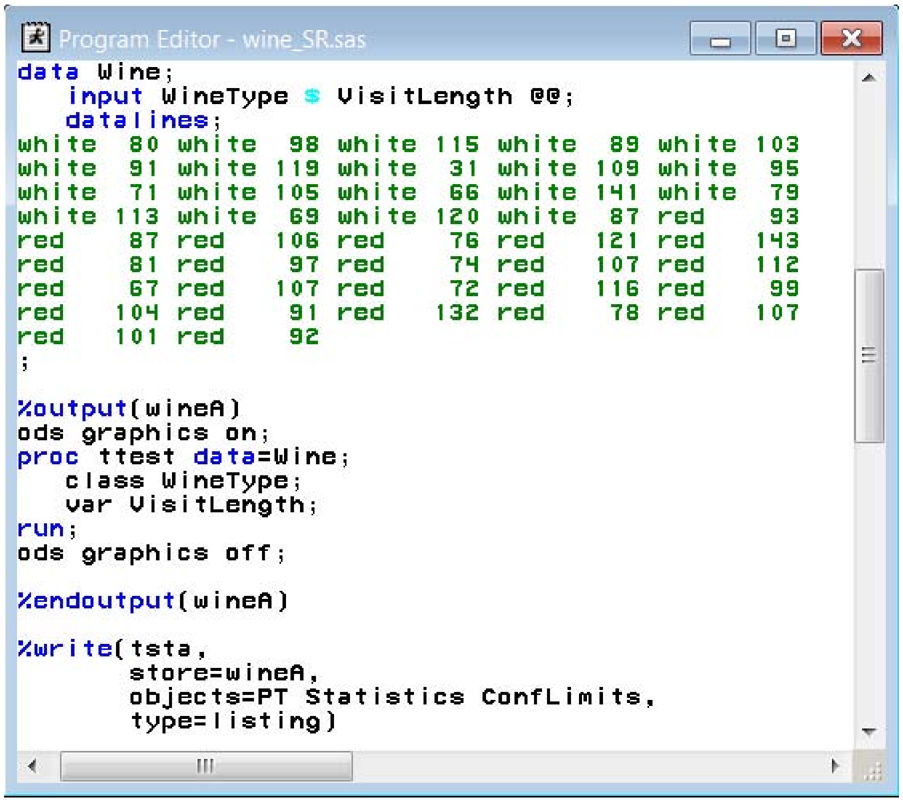
**Figure 10** Working Directory after Compiling



**Step Three: Capture Your Output**

To capture the outputs, you run the generated SAS program `wine_SR.sas`. The SAS Program Editor displays the generated program, as shown in Figure 11.

**Figure 11** Generated SAS Program in the SAS Program Editor



```
data Wine;
   input WineType $ VisitLength @@;
   datalines;
white  80 white  98 white 115 white  89 white 103
white  91 white 119 white  31 white 109 white  95
white  71 white 105 white  66 white 141 white  79
white 113 white  69 white 120 white  87 red    93
red    87 red   106 red    76 red   121 red   143
red    81 red    97 red    74 red   107 red   112
red    67 red   107 red    72 red   116 red    99
red   104 red    91 red   132 red    78 red   107
red   101 red    92
;

%output(wineA)
ods graphics on;
proc ttest data=Wine;
   class WineType;
   var VisitLength;
run;
ods graphics off;

%endoutput(wineA)

%write(tsta,
       store=wineA,
       objects=PT Statistics ConfLimits,
       type=listing)
```

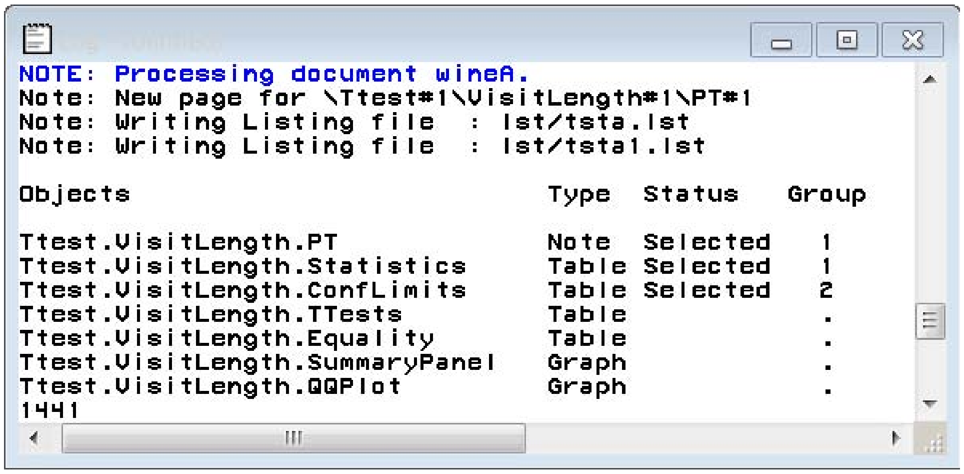
Figure 11 displays the code lines that were generated from the **`Datastep`** environment and the **`Sascode`** environment. Additionally, automatically generated macro calls (**`%output`** and **`%endoutput`**) store the generated output and replay the requested outputs to external files (**`%write`**).

Compile the program by selecting **Run**→**Submit** from the main SAS menu.

When you execute the program, the SAS Log displays an information table for each \**`Listing`** and \**`Graphic`** tag specified in your document. You can use the tables to determine the fully qualified name for each output object and to confirm your output selections. Figures 12 and 13 display the information tables for the \**`Listing`** tag and the \**`Graphic`** tag that are shown in Figure 3 and Figure 4, respectively. As the tables indicate, all of the output that is generated from the **`Sascode`** environment is available in the ODS document **`wineA`**. However, only the objects specified in the **`objects=`** option in the \**`Listing`** and \**`Graphic`** tag are selected.

Figure 12 displays the information table that corresponds to the \**`Listing`** tag.

**Figure 12** SAS Log after Execution, Listing Request

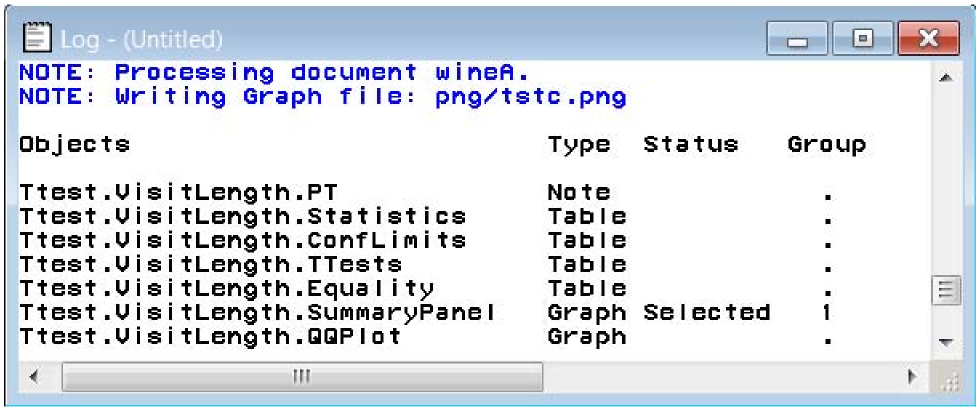


```
NOTE: Processing document wineA.
Note: New page for \Ttest#1\VisitLength#1\PT#1
Note: Writing Listing file  : lst/tsta.lst
Note: Writing Listing file  : lst/tsta1.lst

Objects                             Type  Status    Group

Ttest.VisitLength.PT                Note  Selected    1
Ttest.VisitLength.Statistics        Table Selected    1
Ttest.VisitLength.ConfLimits        Table Selected    2
Ttest.VisitLength.TTests            Table             .
Ttest.VisitLength.Equality          Table             .
Ttest.VisitLength.SummaryPanel      Graph             .
Ttest.VisitLength.QQPlot            Graph             .
1441
```

Figure 12 shows that the note **`PT`** and the tables **`Statistics`** and **`ConfLimits`** are selected from the **`wineA`** ODS document and written to external files. By default, the \**`Listing`** tag writes output files to a subdirectory named ‘lst’.

Figure 13 displays the information table that corresponds to the \**`Graphic`** tag.

**Figure 13** SAS Log after Execution, Graphic Request

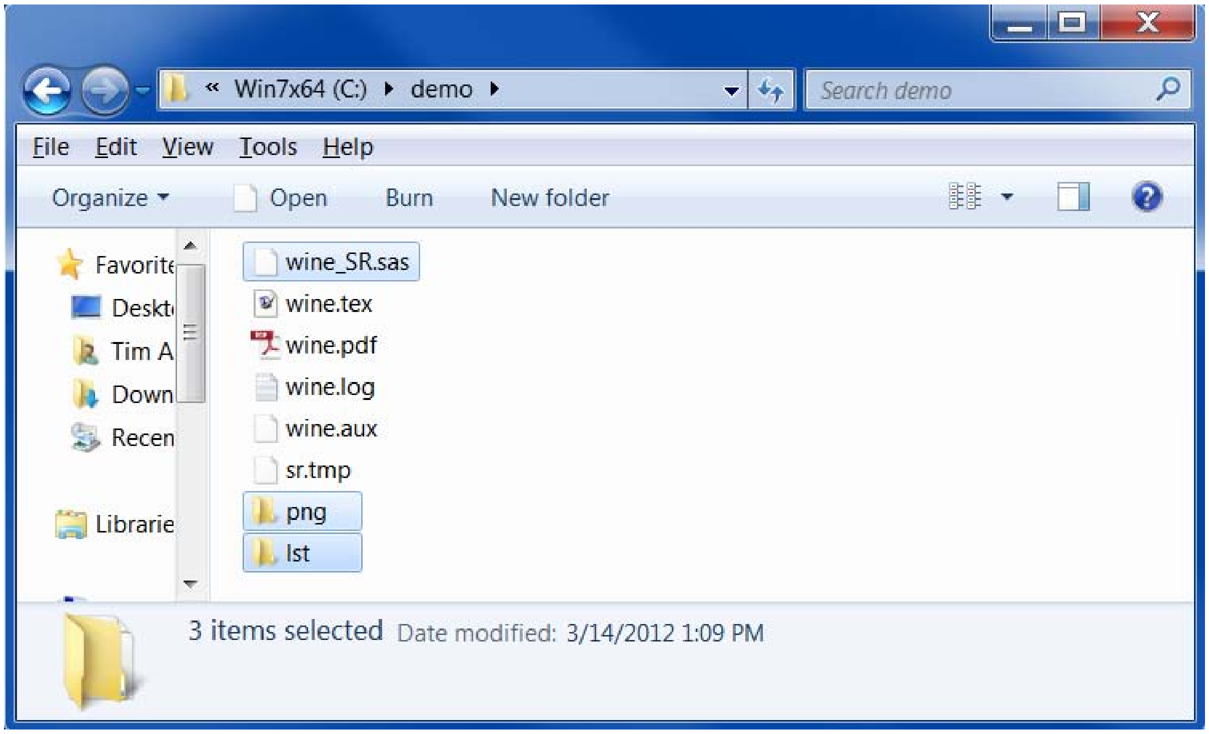


```
NOTE: Processing document wineA.
NOTE: Writing Graph file: png/tstc.png

Objects                             Type  Status    Group

Ttest.VisitLength.PT                Note              .
Ttest.VisitLength.Statistics        Table             .
Ttest.VisitLength.ConfLimits        Table             .
Ttest.VisitLength.TTests            Table             .
Ttest.VisitLength.Equality          Table             .
Ttest.VisitLength.SummaryPanel      Graph Selected    1
Ttest.VisitLength.QQPlot            Graph             .
```

Figure 13 shows that the graph **`SummaryPanel`** is selected from the **`wineA`** ODS document and written to an external file. By default, the \**`Graphic`** tag writes graph output files to a subdirectory named ‘png’.

Figure 14 displays the contents of the working directory after executing the SAS program. The two output subdirectories ‘lst’ and ‘png’ are now included.

**Figure 14** Working Directory after Executing SAS Program

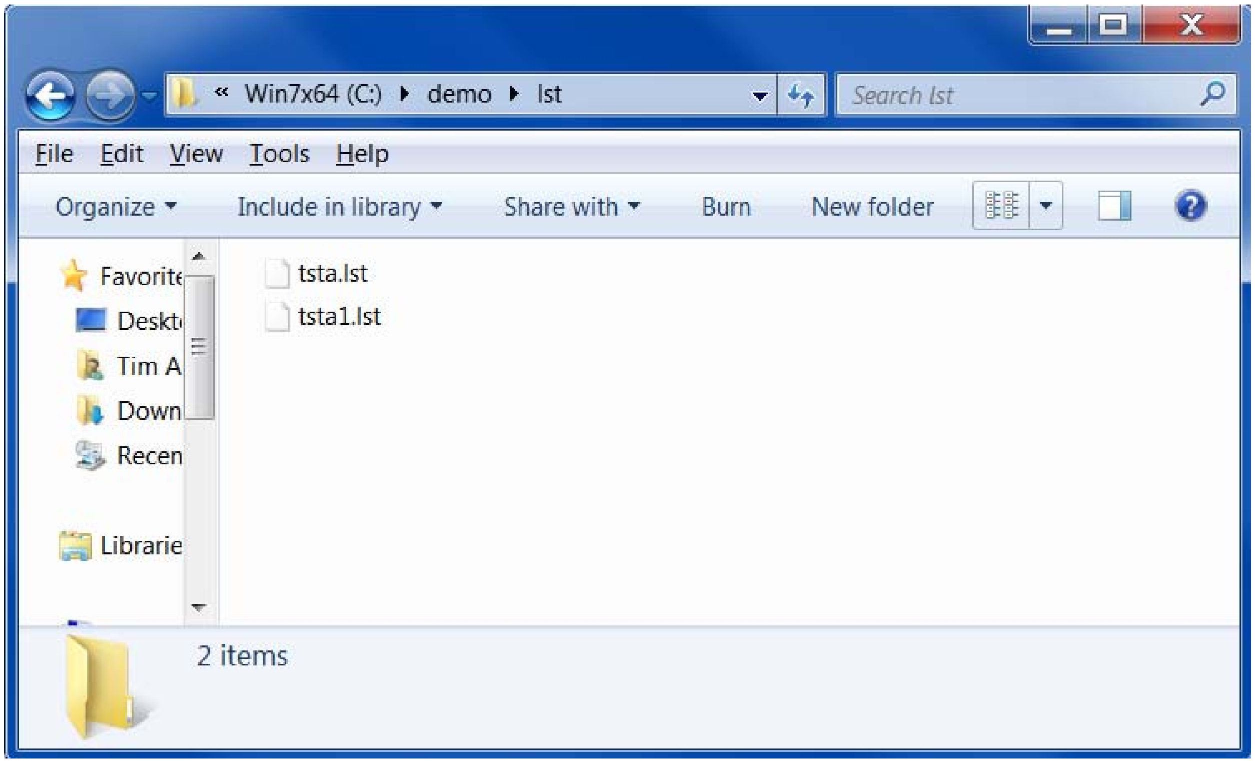


wine_SR.sas
wine.tex
wine.pdf
wine.log
wine.aux
sr.tmp
png
lst

3 items selected Date modified: 3/14/2012 1:09 PM

Figure 15 displays the contents of the ‘lst’ directory, which contains the output files with tables and notes.

**Figure 15** The ‘lst’ Directory after Executing SAS Program

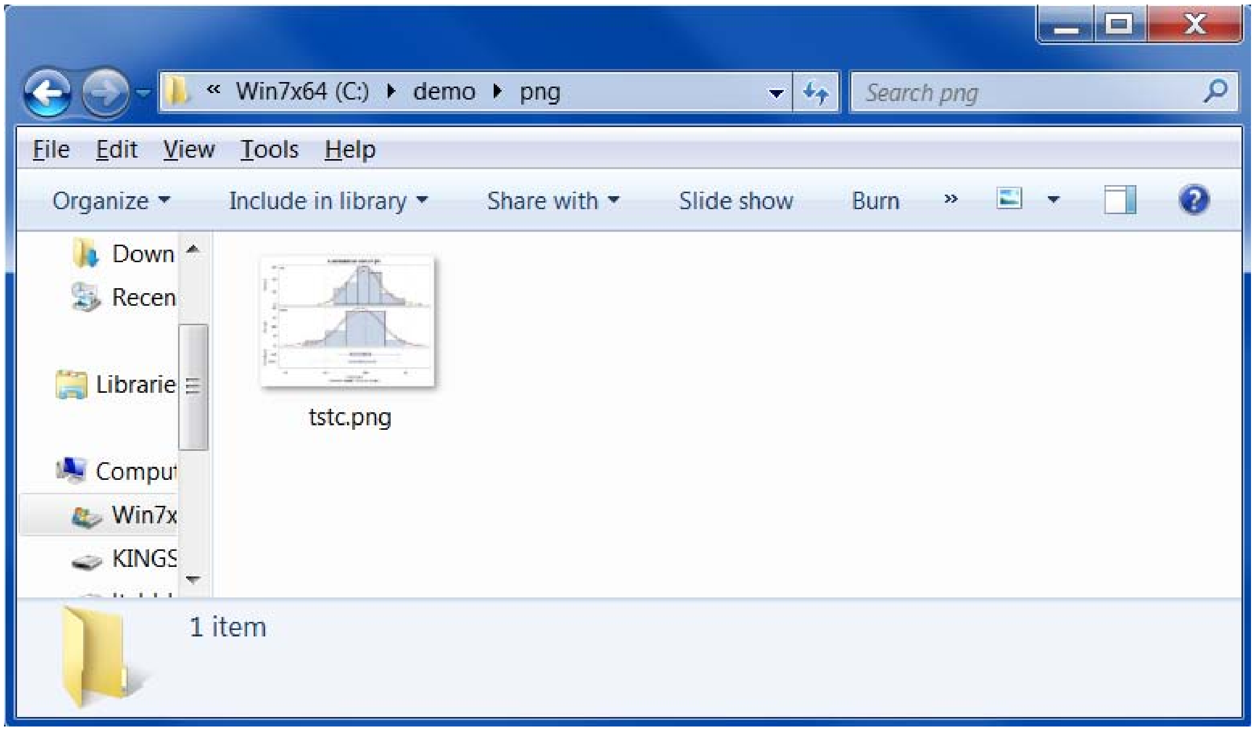

« Win7x64 (C:) ▸ demo ▸ lst
Search lst
File Edit View Tools Help
Organize Include in library Share with Burn New folder
Favorite
Deskt
Tim A
Down
Recen
Librarie

tsta.lst
tsta1.lst

2 items

Figure 16 displays the contents of the ‘png’ directory, which contains the output files with graphs.

**Figure 16** The ‘png’ Directory after Executing SAS Program


« Win7x64 (C:) ▸ demo ▸ png
Search png
File Edit View Tools Help
Organize Include in library Share with Slide show Burn
Down
Recen
Librarie
Comput
Win7x
KINGS
tstc.png
1 item

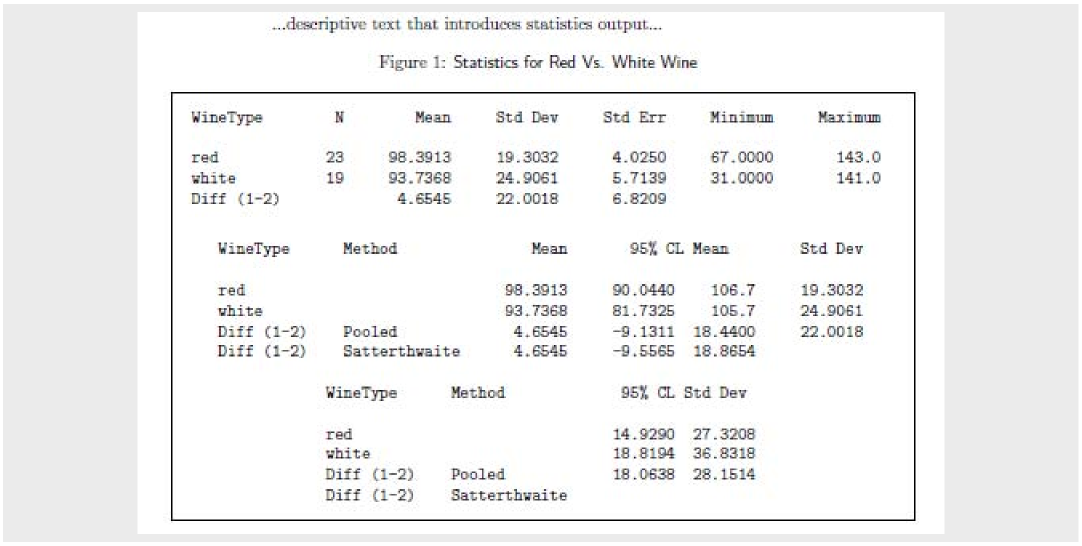
## Step Four: Recompile Your Document

The last step is to recompile your document with pdfLaTeX. As in the first compilation, you can use a LaTeX-aware editor such as TeXworks, or you can use the **`pdflatex`** command in a terminal window.

In this recompilation step, the outputs captured by the SAS program are included in the final PDF document.

Figure 17 displays the results of the \**`Listing`** tag. The content was missing after the first compilation (Figure 9), but now displays the selected outputs (**`PT`**, **`Statistics`**, and **`ConfLimits`**).

**Figure 17** Resulting PDF after Executing SAS Program: Listing



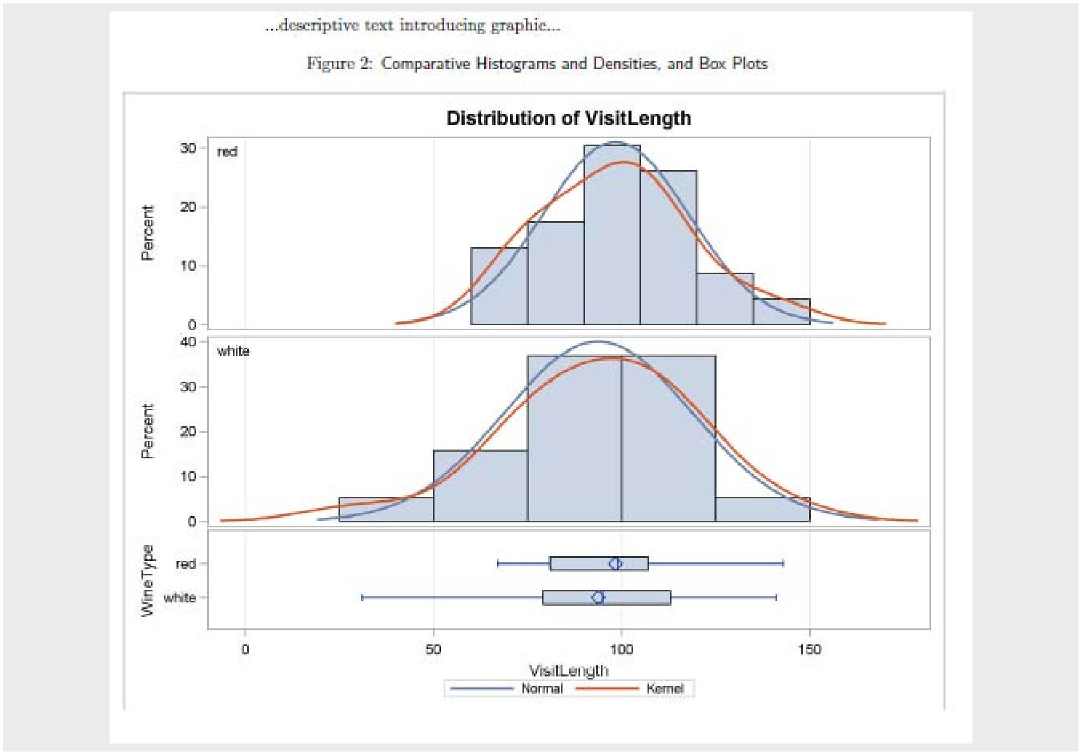
...descriptive text that introduces statistics output...

Figure 1: Statistics for Red Vs. White Wine

```
WineType        N       Mean    Std Dev    Std Err    Minimum    Maximum

red            23    98.3913    19.3032     4.0250    67.0000      143.0
white          19    93.7368    24.9061     5.7139    31.0000      141.0
Diff (1-2)            4.6545    22.0018     6.8209

   WineType      Method              Mean      95% CL Mean        Std Dev

   red                            98.3913    90.0440    106.7     19.3032
   white                          93.7368    81.7325    105.7     24.9061
   Diff (1-2)    Pooled            4.6545    -9.1311  18.4400     22.0018
   Diff (1-2)    Satterthwaite     4.6545    -9.5565  18.8654

               WineType      Method           95% CL Std Dev

               red                           14.9290  27.3208
               white                         18.8194  36.8318
               Diff (1-2)    Pooled          18.0638  28.1514
               Diff (1-2)    Satterthwaite
```

Figure 18 displays the results of the \**`Graphic`** tag, which displays the selected graph (**`SummaryPanel`**).

**Figure 18** Resulting PDF after Executing SAS Program: Graphic



...descriptive text introducing graphic...

Figure 2: Comparative Histograms and Densities, and Box Plots

When you generate the SAS program by compiling your document with pdfLaTeX, the StatRep package does the following:

- The lines in the **`Datastep`** environment are passed unchanged to the program.
- The lines in the **`Sascode`** environment are parsed for line commands and passed to the program.
- Each \**`Listing`** tag selects the specified notes and tables.
- Each \**`Graphic`** tag selects the specified graphs.

When you execute the generated SAS program, the output specified in the \**`Listing`** and \**`Graphic`** tags is automatically captured.

When you compile your LaTeX document again, the \**`Listing`** and \**`Graphic`** tags insert the requested SAS results and page breaks are handled automatically.

## Automatic or Manual Method

To maximize flexibility, the StatRep package provides two methods of writing code in your LaTeX document.

When you create your LaTeX document, you can use either the automatic method described in the preceding example (in which the SAS macro calls are generated automatically) or a manual method (in which you write the **`%output`**, **`%endoutput`**, and **`%write`** macros yourself).

In the automatic method, each **`Sascode`** code block generates an **`%output`** macro call at the beginning of the block and an **`%endoutput`** macro call at the end of the block. Each \**`Listing`** and \**`Graphic`** tag generates the **`%write`** macro to replay the selected output objects to external files.

In the manual method, you decide where and when to make the macro calls. It is only in this respect that the method is manual: the StatRep package still generates your SAS program and displays your code and results. The manual method is used in cases when you want to do one or more of the following:

- capture specialized or complicated output

- capture print output with SAS 9.2
- capture output from the SAS Log
- work interactively when writing (you can interactively develop or debug a certain section of your document by copying code from your LaTeX document and pasting it into a SAS session)

See the *StatRep User's Guide* for several examples that use the manual method to capture specialized output.

Figures 19 and 20 show the differences between the manual and automatic methods. Figure 20 displays the **Sascode** environment and the **Listing** tag from the preceding example. The figure contains blank lines to align with Figure 19, to help you visually compare the methods.

**Figure 19** Manual Method

```
 1 \begin{Sascode}
 2 %*;%output(wineA)
 3 ods graphics on;
 4 proc ttest data=Wine;
 5    class WineType;
 6    var VisitLength;
 7 run;
 8 ods graphics off;
 9
10 %*;%write(tsta,
11 %*;    objects=PT Statistics ConfLimits)
12
13 \end{Sascode}
14
15 \Listing[
16  caption={Statistics...}]{tsta}
17
```

**Figure 20** Automatic Method

```
\begin{Sascode}[store=wineA]

ods graphics on;
proc ttest data=Wine;
   class WineType;
   var VisitLength;
run;
ods graphics off;




\end{Sascode}

\Listing[store=wineA,
  objects=PT Statistics ConfLimits,
 caption={Statistics... }]{tsta}
```

The **Sascode** lines in Figure 19 that differ from those in Figure 20 each begin with the null SAS macro comment (`%*;`), which specifies that the line is written to the program and not displayed. (See Table 3, "**Sascode** Line Commands.") Consequently, both **Sascode** environments produce the same display.

Lines 2 and 10 in Figure 19 specify that the output generated by the intervening lines is stored in an ODS document named **wineA**. The **%output** macro opens the document, and the **%write** macro implicitly closes it.

Lines 10 and 11 call the **%write** macro, which writes the requested outputs (**PT**, **Statistics**, and **ConfLimits**) to the external listing file **tsta**.

Lines 15 and 16 display the **Listing** tag, which specifies the caption for the output and the name of the listing file (**tsta**) to include in the final PDF.

The code lines in Figure 19 and Figure 20 create the same output and produce the same display. The difference is that with the manual method you explicitly write your own macro calls, but with the automatic method the macro calls are written for you. You can use either method, and you can mix the methods in a single document. The manual method is provided for cases in which the automatic method is too inflexible. By using the line commands in a **Sascode** environment, you are free to write your program as you want, while retaining control of the code that is displayed in your final PDF document.

## The `objects=` Option

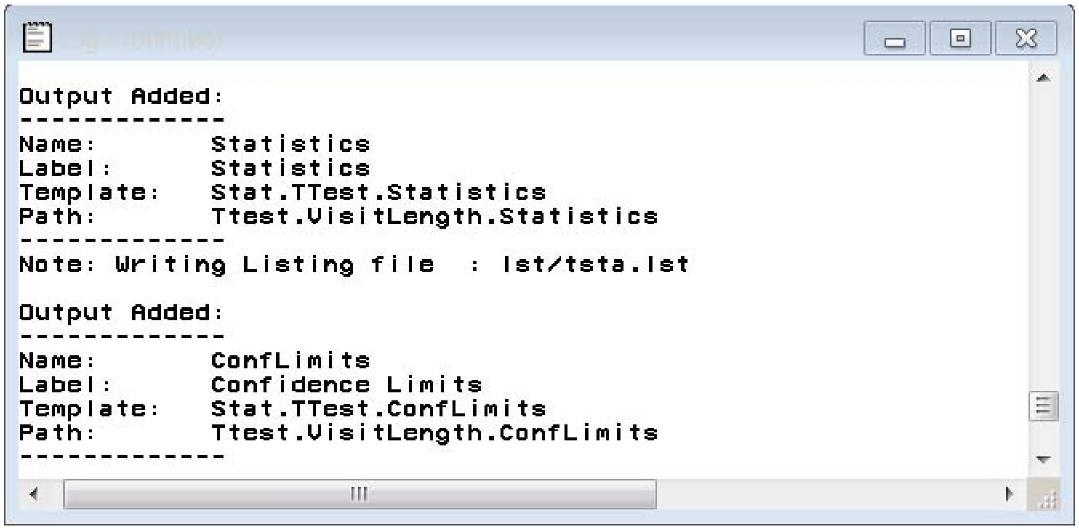
Use the **objects=** option when you want to display only part of the output that is generated from a **Sascode** environment. If you omit the **objects=** option, the **Listing** tag requests all tabular and note output and the **Graphic** tag requests all graph output.

To specify the **objects=** option, you provide the names of the ODS output objects in a space-separated list. You can determine these names from the SAS Log information tables (see Figures 12 and 13), or you can determine the names by issuing the following statement at the beginning of your SAS program:

```
      ODS TRACE ON;
```

The SAS Log shows information about each ODS object that is produced in your program. Figure 21 shows an example.

**Figure 21** Information from `ODS TRACE ON;`



```
Output Added:
-------------
Name:       Statistics
Label:      Statistics
Template:   Stat.TTest.Statistics
Path:       Ttest.VisitLength.Statistics
-------------
Note: Writing Listing file  : lst/tsta.lst

Output Added:
-------------
Name:       ConfLimits
Label:      Confidence Limits
Template:   Stat.TTest.ConfLimits
Path:       Ttest.VisitLength.ConfLimits
-------------
```

To turn off the information display, issue the following statement in your SAS program:

```
ODS TRACE OFF;
```

## Sharing Your Document

When you create your document with the StatRep package, you can share it with others easily. Others can reproduce your analysis with a copy of your document as long as they have SAS 9.2 or later, a LaTeX distribution, and the StatRep package.

You can collaborate with colleagues by using your document as a starting place from which to explore new ideas; you can provide readers with your document (even journal editors) so they can reproduce your analysis on their own. Most importantly, you can be assured that when your readers execute your displayed code, they will generate your displayed results.

## Conclusion

StatRep, a single-source document system, embeds SAS code in LaTeX source files. The system automatically generates a SAS program file that includes SAS macros that use the ODS document to capture the output as external files. When you use StatRep, you create an executable document that guarantees that your research results are reproducible. The StatRep system is available as an open-source package that includes a LaTeX package and a suite of SAS macros.

## References

Baggerly, K. A. and Coombes, K. R. 2009. *Deriving Chemosensitivity from Cell Lines: Forensic Bioinformatics and Reproducible Research in High-Throughput Biology*. Annals of Applied Statistics 3(4):1309–1334.

Knuth, Donald E. 1984. *The TeXBook*. Reading, MA: Addison-Wesley.

Lamport, Leslie. 1994. *LaTeX: A Document Preparation System* 2nd ed. Reading, MA: Addison-Wesley.

Leisch, Friedrich. 2002. Sweave: Dynamic Generation of Statistical Reports Using Literate Data Analysis in W. Härdile and B. Rönz, editors, *Compstat 2002–Proceedings in Computational Statistics*, 575–580. Heidelberg, Germany: Physika Verlag. ISBN 3-7908-1517-9.

Lenth, Russell V. and Højsgaard, Søren. 2007. “SASweave: Literate Programming Using SAS,” *Journal of Statistical Software*, 19(8) 1–20.

TeXLive LaTeX distribution 2011: A Comprehensive, Multi-Platform TeX Document Production System. Available at `www.tug.org/texlive/quickinstall.html`

## Acknowledgments

The authors are grateful to Maura Stokes and Bob Derr of the Advanced Analytics Division at SAS Institute Inc. for their valuable assistance in the preparation of this manuscript. They are also grateful to Anne Baxter for her editorial contributions.

## Contact Information

Your comments and questions are valued and encouraged. Contact the authors:

| | |
|---|---|
| Tim Arnold | Warren F. Kuhfeld |
| SAS Institute Inc. | SAS Institute Inc. |
| SAS Campus Drive | SAS Campus Drive |
| Cary, NC 27513 | Cary, NC 27513 |
| tim.arnold@sas.com | warren.kuhfeld@sas.com |