2018

Macros to Conduct Tests of Multimodality in SAS

Zacharia Neville and Naomi Brownstein

Preprint available from arXiv: https://arxiv.org/abs/1802.00151

Macros to Conduct Tests of Multimodality in SAS

Zachariah Neville[a] and Naomi Brownstein[b]

[a]Department of Statistics, Florida State University, Tallahassee, FL, USA; [b]Department of Behavioral Sciences & Social Medicine, Florida State University, Tallahassee, FL, USA

ARTICLE HISTORY
Compiled February 2, 2018

ABSTRACT
The Dip Test of Unimodality and Silverman's Critical Bandwidth Test are two popular tests to determine if an unknown density contains more than one mode. While the tests can be easily run in R, they are not included in SAS software. We provide implementations of the Dip Test and Silverman Test as macros in the SAS software, capitalising on the capability of SAS to execute R code internally. Descriptions of the macro parameters, installation steps, and sample macro calls are provided, along with an appendix for troubleshooting. We illustrate the use of the macros on data simulated from one or more Gaussian distributions as well as on the famous *iris* dataset.
Word Count: 7561

KEYWORDS
dip test; Silverman's critical bandwidth test; implementation; SAS; software; unimodality

1. Introduction

Determining the number of modes in an unknown density has applications to many fields, including biology [1], political science [2], and psychology [3]. The question of whether a data set contains more than one mode is asked frequently in clustering [4–7]. The presence of multiple modes may indicate that the data was generated from a population with multiple distinct subgroups.

Two popular methods for testing for multimodality include the Dip test by Hartigan and Hartigan [8] and the Critical Bandwidth test by Silverman [9]. Implementations of these tests exist in R, but SAS does not provide procedures for multimodality tests.

In this paper, we provide implementations of the Dip Test and Silverman Test that utilise the capability for SAS software to execute R statements in `PROC IML` [10]. The two SAS macros, %dip and %silverman, offer the same functionality as the R implementations, plus additional options, such as storing the output in a SAS data set or setting the random number generator seed for the Silverman test. Additionally, the SAS macros perform error checking and setup in SAS before executing R statements to calculate the test statistic and p value, and displaying the results in the Results Viewer.

CONTACT Naomi Brownstein. Email: naomi.brownstein@med.fsu.edu. Address: Department of Behavioral Sciences & Social Medicine, College of Medicine, 1115 West Call Street, Tallahassee, FL 32306-4300, USA

First, we present background on the Dip and Silverman tests and discuss current implementations of the tests in the R language. We then explain the details of the %dip and %silverman macros in SAS, including steps for configuring the SAS environment, the implementation details of the macros, the parameters available to customise the tests, and the format of the output. Next, we run the macros on some data sets and discuss the results. We close with a conclusion discussing limitations of our work and areas for future exploration. Finally, we provide an appendix to help with troubleshooting.

2. Background

When examining data from an unknown distribution, users are often interested in whether that underlying distribution is unimodal or multimodal. Many methods are available to answer this question, including the Dip test [8], Critical Bandwidth test [9], and the Excess Mass test [11]. Minnotte provides an overview of these and other tests [12]. Our efforts focus on implementing the Dip and Silverman tests in SAS. We begin by providing some background on these two tests.

The Dip test computes the maximum difference between the empirical distribution function and the unimodal distribution function that minimises the maximum difference [8]. This maximum difference is called the *dip*. To test for statistical significance, the dip statistic D is compared to critical values obtained by simulating with uniform distributions. Hartigan and Hartigan show that the distribution of the dip is asymptotically larger for the uniform than any other distribution with exponentially decreasing tails [8]. The choice of the uniform distribution as a reference yields a conservative test. In fact, in moderate to large samples, the Silverman test has greater power than the the Dip test [13]. Calibration of the Dip test was performed by Cheng and Hall [13], but this calibrated version of the test is not available in R, and hence is not available in the %dip SAS macro. The Dip test is designed for univariate data. An extension of the Dip test for multivariate data was discussed in [8], but the extension is not implemented in the R package or in our SAS macro.

Unlike the Dip test, which uses the empirical distribution function of the data, the Silverman critical bandwidth test uses kernel density estimates to obtain a test statistic [9]. The normal density function is used as the kernel for both theoretical and computational reasons [9]. The test statistic is the critical window width (h_{crit}) defined as the smallest window such that the kernel density has `k` or fewer modes. Larger values of h_{crit} indicate that more smoothing of the data is necessary to obtain a density with `k` or fewer modes, thereby suggesting rejection of the null hypothesis. To assess the statistical significance using a p value, bootstrap samples are generated, and the kernel density estimate and window width h are calculated for each sample. The unadjusted test as originally given in Silverman [9] is vulnerable to identifying spurious modes in the tails of distributions; this tendency is discussed further in Hall and York [14]. In other cases, the unadjusted Silverman test is conservative and thus can suffer from low power [14]. For the `k = 1` (unimodal null hypothesis and a multimodal alternative) case, our SAS implementation and the `silvermantest` [15] package in R use the calibration performed by Hall and York that is more asymptotically accurate. The Silverman test is designed only for univariate data.

Additionally, Silverman's test may be used to determine the number of modes in the distribution of the population from which the data was sampled. If the null hypothesis of unimodality is rejected, the user may wish to find out if there are only two modes,

or if more modes exist. On the other hand, when the conclusion of the (unadjusted) Silverman test is a failure to reject the unimodal null hypothesis, one should not retry the Silverman test with a larger value of `k`. This is because a failure to reject for `k`=1 means there is insufficient evidence to conclude that the data came from a multimodal distribution. As a result, one should not further test the null hypothesis that the data came from a distribution with two or fewer modes, because one would expect the same result as before: a failure to reject. In general, the number of modes can be estimated by conducting Silverman tests sequentially, beginning with `k`=1 and stopping at the first `k` such that the null hypothesis is not rejected at the *a priori* specified significance level.

Implementations of both the Silverman test and Dip test are available in R. The Dip test can be performed using the `diptest` R package, which is currently available for download [16]. As of the time of writing of the present paper, the Silverman test is not available for download as an R package. We are aware that the `silvermantest` package was previously available as an R package through CRAN, but it was only available to download as a package from the original authors' website at the time of writing our macros and documentation [15][1]. The uncalibrated test as originally described by Silverman [9] is the default selection in both the R package and the SAS macro. For the `k` = 1 (unimodal vs multimodal) case only, the calibrated version of the test provided by Hall and York [14] is available as an alternative option. The Dip test function in R does not include an option to produce the calibration recommended by Hall and Cheng.

SAS software does not currently provide implementations of the Dip Test or Silverman Test, nor of any other tests for multimodality. Because neither Dip nor Silverman is currently available in base SAS software, SAS users wishing to perform these tests must transfer their data and run their analysis in a separate R environment before returning to work in SAS. This limitation is problematic, because the user may not be familiar with the R language, it is not convenient to repeatedly export data from SAS and import into R, and it is easier and more reproducible to use only a single program while performing data analyses.

While the %dip and %silverman SAS macros require R to be installed on the machine, they allow a SAS user to perform these multimodality tests without leaving the SAS environment. They also work directly with SAS data sets, and the user does not need to transfer data between SAS software and R. The next section provides details on the implementation of our macros.

3. Macro Implementation

We implement both %dip and %silverman as macros in SAS. The `macroDefinitions.sas` file contains the macro definitions and must be run before the macros can be used. Once the `macroDefinitions.sas` file has been successfully executed and the macros initialised, the macros can be used in the SAS session. These SAS macros utilise existing implementations in R, along with the ability of SAS software to execute R statements, in order to add the Dip and Silverman tests to SAS without having to code them from scratch.

[1]To create the %silverman SAS macro, we downloaded the `silvermantest_1.0.tar.gz` file and utilised code from the R files inside the package.

3.1. General System Requirements

Because both macros call R from the SAS system, R statistical software must be installed on either the SAS server (if running in an enterprise environment or using SAS OnDemand for Academics, for example) or on the local machine (if running SAS software on an individual computer), and the `RLANG` option must be enabled. The `RLANG` option can only be set at SAS startup; one convenient way is to modify the `SASV9.cfg` file and insert the `RLANG` option [17, p. 241]. Additional information about locating and modifying the configuration file can be found in SAS documentation [18, p. 12]. Further details on how to enable the `RLANG` option are provided in Section 3.3.

The interface to R is supported only on computers running a Windows or Linux operating system. R cannot be called from SAS University Edition [17, p. 240]. At the time of writing, our initial testing with SAS OnDemand for Academics (SAS ODA) was also unsuccessful; we were unable to modify the `SASV9.cfg` file and could not execute any R statements on SAS ODA. The SAS Viya software platform allows the execution of code in many languages, including R, and may be capable of running these macros. However, we have not tested this functionality.

The %dip SAS macro uses the `diptest` package in R, and the %silverman SAS macro uses the `splines` package in R if the user has specified that they wish to use the calibrated version of the Silverman Test. The SAS macros will install the required R packages if they are not already installed on the system. For each macro, either the machine must be connected to the Internet and capable of installing new R packages, or the packages must already be installed on the machine. If the macro needs to install a new package, a pop-up list of CRAN mirrors may appear. After selecting one, the package will automatically download and install. For best results, the user should use the most current release of SAS. Older versions of SAS may have compatibility problems with the `RLANG` option or the specific version of R installed on the machine.

3.2. Required Arguments for Both Macros

The %dip and %silverman macros have certain attributes in common. Each has two required arguments: a univariate SAS data set containing only numeric values, and a file path or fileref to the appropriate macro file containing the R statements executed by the macro. All other arguments for each macro are optional.

The data set provided to the %dip and %silverman is specified in the `dipData` and `silvData` argument, respectively. For both macros, the input must be a one-dimensional SAS data set; the implementations of the Dip and Silverman tests were not meant for multivariate data. If multidimensional data is used in either macro, then that macro generates an error and quits. The data set must be entirely numeric; if non-numeric data is found in the data set, an error is generated in the Log and Results Viewer, and the test is not run. Data sets with certain SAS formats - such as date and time formats - may generate errors when used in the macros. For more details, including steps for removing formats, please see the appendix.

The `include` argument in the %dip and %silverman macros must be a complete file path to the `dip.sas` or `silverman.sas` file provided with this paper, respectively. The `.sas` extension and quotation marks are required. Using a relative file path, such as `include =` "`dip.sas`", does not work. The `include` argument is also required, because the `dip.sas` and `silverman.sas` files contain all of the `PROC IML` code (which then contains R code) needed to perform the tests. Missing or improperly specified files result in error messages. Alternatively, a fileref can be associated with the `dip.sas`

or `silverman.sas` file, and that fileref can be used for the `include` argument. The `FILENAME` statement can be used to assign a fileref; more details are available in SAS documentation [18, p. 149]. Easy portability is one advantage of using a fileref instead of hard-coding the file path as a string, because the file path needs to be only changed once (where the fileref is assigned) rather than for every single macro call. The `samples.sas` file provided with this paper uses `FILENAME` statements.

3.3. Installation Steps

A summary of steps required to prepare the SAS environment to run the macros is provided in Table 1. When running the macros for the first time, the instructions in both sections of Table 1 must be completed. After that, only the second section needs to be completed in order to use the macros. Additional details for each step are provided below.

(1) **Ensure that R is installed on the machine.**
R software must be installed on the same machine that runs the SAS software. If accessing a SAS workspace server through client software such as SAS Enterprise Guide, then R software must be installed on the SAS server [17, p. 240]. The interface to R is supported on computers running Windows or Linux operating systems.

(2) **Open the `SASV9.cfg` file and ensure that the `RLANG` option has been inserted.**
More details on locating and modifying the `SASV9.cfg` configuration file can be found in the SAS Companion [18, p. 12]. One possible location may be: `C:\Program Files\SASHome\SASFoundation\9.4\nls\en\sasv9.cfg`. Adding `-RLANG` to the configuration file will suffice; place it near the top of the file. An excerpt from a sample `SASV9.cfg` file is shown in Figure 1, with line numbers on the left side showing that `-RLANG` is placed at the top of the config file. The other configuration options in Figure 1 are only shown to provide context. The `RLANG` option needs to be specified at startup. After adding the `RLANG` option to `SASV9.cfg`, the SAS software should be restarted so that the updated configuration is used.
[Figure 1 near here.]

(3) **If using the %dip macro or the adjusted option of the %silverman macro: Verify that the machine can install new R packages from the Internet or that the appropriate R packages are installed on the machine.**
The Dip test requires the `diptest` R package to be installed. When using the adjusted Silverman test, the `splines` R package must be installed. If these packages are not already installed on the computer, the SAS macros will install them automatically. Therefore, the machine must already have the packages installed or the machine must be capable of installing these packages.

(4) **Download the supplemental code files and store `dip.sas` and `silverman.sas` in a safe location for future use.**
Download the attached supplemental files. The `dip.sas` and `silverman.sas` files are necessary to run the %dip and %silverman SAS macros, respectively. These files should be stored in an easily accessible folder for future use. Because the %dip and %silverman macros need to know the file paths to the `dip.sas` and `silverman.sas` files, respectively, storing the files in a temporary location,

such as the Downloads folder, may not be ideal.

(5) **Execute the `macroDefinitions.sas` file to add the macros to the operating environment.**
The `macroDefinitions.sas` file contains the macro definitions. After the file has been executed, the macros will be available for use in the current SAS session. Users who plan to run these macros regularly may choose to store them for use between sessions by using the autocall macro facility or the stored compiled macro facility. More details are available in the SAS Macro Language: Reference documentation [19, p. 113].

(6) **Upon successful completion of the prior steps, the macros can be used as described in the paper.**
The `samples.sas` file contains replication code for the examples shown in Section 4.

[Table 1 near here.]

3.4. Dip Macro

The following is the full macro call with all of the parameters and default values listed:

```
%dip(dipData =, simulatepvalue = 0, reps = 2000, out = ,
completecase = 0, include = )
```

The %dip macro takes several user arguments, with two of them required: a data set (`dipData`) and a file path to the `dip.sas` file (`include`). It performs some basic error checking and setup in SAS and then transfers the arguments and data set to R, where it performs the Dip Test of Unimodality.

A list of the arguments to the %dip macro and their descriptions is provided in 3.4.1 below. For arguments that have a default value, the default is the same in the SAS macro as in the `diptest` R package. Note that, because SAS software does not have Boolean data types, we use 0 in place of FALSE and 1 in place of TRUE.

The `completecase` argument allows for the macro to perform a complete case analysis. While the Dip Test implementation (in R, and thus, in our SAS macro) ignores missing values in the data set by default, the `completecase` argument is added to be consistent with the %silverman SAS macro and Silverman Test, which require missing data to be removed before running the test. If missing data is present and `completecase` is either not specified or set equal to zero, then the macro will stop and return an error. If `completecase` is set to 1, then a complete case analysis will be performed and missing data ignored while performing the Dip test.

The results from running the Dip test in R using the `diptest` package and from running the Dip test using the %dip macro are the same. This allows for reproducibility of results for users of both R and SAS. The %dip macro will output the same data as the Dip test in R: the name of the test, the Dip statistic D, the p value, and the alternative hypothesis are all output to the Results Viewer. Error and warning messages are displayed both in the Results Viewer and the Log. Messages output to the SAS Log follow the same color scheme as in base SAS: red for errors, green for warnings, and blue for successful execution.

3.4.1. Arguments for the %dip macro

Below is a complete list of arguments for the %dip macro, including whether the argument is required and the default value if there is one. For those SAS macro arguments

which also exist in the `diptest` R package [16], the name of the corresponding R argument is given.

`dipData` (named `x` in R), Required, Default: *none*
`dipData` specifies the SAS data set to be used in the analysis. All data in `dipData` must be numeric.

`simulatePvalue` (named `simulate.p.value` in R), Optional, Default: 0 (FALSE)
A Boolean (0 for FALSE, 1 for TRUE) value indicating whether to simulate p values by Monte Carlo simulation.

`reps` (named `B` in R), Optional, Default: 2000
The number of repetitions to be used in Monte Carlo simulation. If `simulatePvalue` is 0, then this parameter will be ignored.

`out`, Optional, Default: *none*
Specifies a SAS data set in which to store output from the macro, including the dip statistic D, the alternative hypothesis in words, and the p value. There should not be quotes around the data set name. If left blank, then output will be displayed in the SAS Results Viewer window and not stored in any data set.

`completecase`, Optional, Default: 0 (FALSE)
A Boolean (0 for FALSE, 1 for TRUE) value indicating whether to perform a complete case analysis.

`include`, Required, Default: *none*
The complete file path to `dip.sas`, given as a string and surrounded by quotes, or a SAS fileref associated with the `dip.sas` file.

3.5. Silverman Macro

The following is the full macro call with all of the parameters and default values listed:

```
%silverman(silvData, k = 1, M = 999, adjust = 0, digits = 6, setSeed
= , showSeed = 0, outSeed = , out = , completecase = 0, include = )
```

The %silverman macro takes two required arguments: a SAS data set containing the data to be tested, and the complete file path to the `silverman.sas` file. Several optional arguments are available to change how the test is performed and control how output is displayed and stored. The macro performs basic error checking in SAS before transferring the data set and arguments to R, where the Silverman Critical Bandwidth test is performed. If applicable, error and warning messages are displayed both in the Results Viewer and the Log. Results of the test are shown in the Results Viewer. Additional parameters for customising the behavior are available, such as providing a seed for the random number generator (RNG) in R or providing a SAS data set to store results from the test.

A list of arguments and their descriptions for the %silverman macro can be found in 3.5.1 below. For arguments that have a default value, the default is the same in the macro as in the `silvermantest` package obtained from [15].

Optional arguments to the Silverman macro allow for customisation. The `completecase` argument allows for the macro to perform a complete case analysis. The original code from the `silvermantest` R package does not run the test when missing data is present: an error is returned. The `completecase` argument for the SAS macro was added to handle missing data. If `completecase` is not set or the value is set to 0, and missing data is present, then the macro stops and returns an error. If `completecase` is set to 1, a complete case analysis is performed and missing data is ignored while performing the Silverman test.

The `setSeed` argument allows for the user to set the seed for the RNG in R, potentially allowing for reproducible results between different users and environments. Currently, the results between the Silverman test in R and the %silverman macro in SAS do not perfectly match even if the same seed is set in both environments. However, results within a given environment (either R or SAS) can be consistently reproduced by setting the seed with `setSeed` argument in the %silverman SAS macro or calling `set.seed()` in R. The `showSeed` argument outputs the state of the RNG to the Results Viewer, and may be useful to compare between machines during troubleshooting to verify that the RNG state is the same on each machine. The `outSeed` argument outputs the state of the RNG to a specified SAS data set and may prove useful during troubleshooting.

The %silverman SAS macro outputs the name of the test, the null hypothesis, and the p value to the Results Viewer. Additional informational messages are output when optional arguments are specified. For example, a message is output if the R seed is set or if the calibrated version of the Silverman test is used.

3.5.1. Arguments for the %silverman macro

A complete list of arguments for the %silverman macro is given below. Whether the argument is required and the default value (if applicable) are also provided. For those SAS macro arguments which also exist in the `silvermantest` R package [15], the name of the corresponding R argument is given.

`silvData` (named `x` in R), Required, Default: *none*
The SAS data set to be used in the Silverman test. Must be one-dimensional and numeric.

`k` (named `k` in R), Optional, Default: 1
The number of modes to be tested. Null hypothesis H_0: number of modes $\leq k$. Must be an integer value.

`M` (named `M` in R), Optional, Default: 999
The number of bootstrap replications. An integer value.

`adjust` (named `adjust` in R), Optional, Default: 0 (FALSE)
A Boolean value indicating whether to adjust p values using the work by Hall and York [14]. Use `adjust` = 0 for FALSE (do not adjust) and 1 for TRUE (adjust). The adjustment is performed only when `k` = 1.

`digits` (named `digits` in R), Optional, Default: 6

Number of digits for rounding the p value. Only applicable when `adjust` = 1.

`setSeed`, Optional, Default: *none*
A number to be passed as an argument to the `set.seed()` function in R. This sets the seed of the random number generator (RNG) in R.

`showSeed`, Optional, Default: 0 (FALSE)
A Boolean value (0 for FALSE, 1 for TRUE) indicating whether to show the result of `.Random.seed` from R (returns a vector with the current random number generator state). Used primarily for troubleshooting.

`outSeed`, Optional, Default: *none*
The name of a SAS data set to store the result of `.Random.seed` from R. Used primarily for troubleshooting.

`out`, Optional, Default: *none*
Specifies a SAS data set that will contain output from the macro, including the null hypothesis, the number of modes tested, and the p value. If left blank, then output will be displayed in the SAS Results Viewer window and not stored in any data set.

`completecase`, Optional, Default: 0 (FALSE)
A Boolean value (0 for FALSE, 1 for TRUE) indicating whether to perform a complete case analysis.

`include`, Required, Default: *none*
The complete file path to `silverman.sas`, given as a string and surrounded by quotes, or a SAS fileref associated with the `silverman.sas` file.

4. Examples

We demonstrate the use of the %dip and %silverman macros on a variety of data sets. Section 4.1 focuses on data simulated from one or more Gaussian distributions. Section 4.2 highlights the performance of the macros on a famous dataset used in computer science containing measurements from iris flowers described in [20]. Within both of these subsections, we describe the data sets, provide the code and output to conduct the %dip and %silverman macros on the data sets, and summarise the results.

4.1. Simulated Gaussian Data

The first example includes a series of data sets with 300 observations generated from a single standard normal distribution or small mixture of univariate normal distributions with unit variance. Four data sets are discussed: one with a single normal distribution (*oneNorm*), two with two normal distributions (*twoNorms1* and *twoNorms2*), and one with three normal distributions (*threeNorms*). Data sets consisting of multiple normal distributions contain equal proportions of data from each component distribution (150 observations from each distribution for the mixture of two normal distributions, and

100 observations from each distribution for the mixture of three normals). The components of the mixtures in *twoNorms1* are centered at 0 and 2. Similarly, the components of *twoNorms2* are centered around 0 and 4. The final example, *threeNorms*, consists of a mixture of three normal distributions with means of 0, 3.5, and 7.

The SAS code to generate these data sets is provided in the `samples.sas` file; the random number generator is set for each data set to facilitate reproducibility. Before the user can perform any of the tests, the user should open and execute the `macroDefinitions.sas` file. Execution creates both macros and makes them available for use, as detailed in Section 3.3 and Table 1. Then, to replicate the examples in this section, the user may open the `samples.sas` file. The first portion of the file is dedicated to creating the data sets used in these examples, and the remainder calls the macros.

4.1.1. Dip Macro with Simulated Gaussian Data

The first goal of the analysis is to use the Dip test to test each of these four SAS data sets for multimodality. The macro calls shown below achieve this goal:

```
%dip(oneNorm, include = "C:\Documents\SAS\dip.sas")
%dip(twoNorms1, include = "C:\Documents\SAS\dip.sas")
%dip(twoNorms2, include = "C:\Documents\SAS\dip.sas")
%dip(threeNorms, include = "C:\Documents\SAS\dip.sas")
```

The file path given in the `include` argument needs to be updated to reflect the location of the `dip.sas` file on the user's specific machine or server. In this example, the optional arguments `simulatepvalue`, `reps`, and `out` are not specified. To further customise the behavior of the Dip test and output from the test, the user can modify these parameters as desired. Descriptions of these parameters are given in 3.4.1.

4.1.2. Silverman Macro with Simulated Gaussian Data

The second goal of the analysis is to use Silverman's test to determine, for each of these four SAS data sets, whether the data was generated from a multimodal distribution. The first round of tests uses the default value of $\texttt{k} = 1$, which tests whether the data came from a unimodal or multimodal distribution. Because the Hall and York calibration [14] is available for the $\texttt{k} = 1$ case, we also present results for the adjusted Silverman test. The macro calls to test the null hypothesis of unimodality are shown below. The first four use the unadjusted version of the Silverman test, while the next four use the adjustment. Including both sets of calls illustrates the effect of the adjustment on the results.

The `setSeed` argument is provided for all of the calls to the %silverman macro in these examples to facilitate reproducibility of the results.

```
%silverman(oneNorm, setSeed = 1234, include =
"C:\Documents\SAS\silverman.sas")
%silverman(twoNorms1, setSeed = 1234, include =
"C:\Documents\SAS\silverman.sas")
%silverman(twoNorms2, setSeed = 1234, include =
"C:\Documents\SAS\silverman.sas")
%silverman(threeNorms, setSeed = 1234, include =
"C:\Documents\SAS\silverman.sas")
%silverman(oneNorm, setSeed = 1234, adjust = 1, include =
```

```
"C:\Documents\SAS\silverman.sas")
  %silverman(twoNorms1, setSeed = 1234, adjust = 1, include =
"C:\Documents\SAS\silverman.sas")
  %silverman(twoNorms2, setSeed = 1234, adjust = 1, include =
"C:\Documents\SAS\silverman.sas")
  %silverman(threeNorms, setSeed = 1234, adjust = 1, include =
"C:\Documents\SAS\silverman.sas")
```

The third and final goal is to determine the number of modes in the distribution for each of these four SAS data sets. The application of Silverman's test with larger values of `k` is shown for data sets for which the null hypothesis of unimodality was rejected. The adjustment by Hall and York [14] is only available for the `k` = 1 case; thus, the remaining tests in this example are restricted to the unadjusted version. The macro calls are included below:

```
  %silverman(twoNorms2, k = 2, setSeed = 1234, include =
"C:\Documents\SAS\silverman.sas")
  %silverman(threeNorms, k = 2, setSeed = 1234, include =
"C:\Documents\SAS\silverman.sas")
  %silverman(threeNorms, k = 3, setSeed = 1234, include =
"C:\Documents\SAS\silverman.sas")
```

4.1.3. Results for Simulated Gaussian Data

[Figure 2 near here.]

The output from the first %dip macro call is displayed in Figure 2. The results from the %dip and %silverman macro calls on the simulated data are summarised in Table 2. The Dip test correctly fails to reject the null hypothesis for the data set *oneNorm*, concluding there is insufficient evidence that the data was generated from a multimodal distribution. For *twoNorms2*, the Dip Test easily rejects the null hypothesis of unimodality, with a p value of 2.032×10^{-6}. It is very unlikely that data generated from a unimodal distribution would display such a large dip. On the other hand, for *twoNorms1*, the p value associated with this test is 0.3146, indicating a failure to reject the null hypothesis of unimodality. In this case, the means of these two normal distributions are separated by only two standard deviations. Thus, as is evident in a histogram of the data, the modes approach each other and are somewhat more difficult to distinguish visually and statistically.

[Figure 3 near here.]

[Table 2 near here.]

The output from the first call to %silverman is displayed in Figure 3, and the p values from each of these tests are provided in Table 2. Like the Dip, Silverman's test correctly fails to reject the unimodal null hypothesis for the *oneNorm* data set and rejects the null for the *twoNorms2* data set. In contrast to the Dip, while the unadjusted Silverman test fails to reject for the *twoNorms1* data set, the adjusted Silverman test is very close to the 0.05 level, indicating that adjusted Silverman test can more easily distinguish the nearby but distinct modes. In the final data set, *threeNorms*, the difference between the adjusted and unadjusted Silverman tests remains apparent. The unadjusted test is borderline, with a p value of 0.0591, while the adjusted test rejects handily at the 0.05 level, with a p value of 0.0118. The unadjusted Silverman test yields higher p values than the adjusted version, supporting the known finding, discussed in Section 2, that the unadjusted Silverman test is conservative [14].

Results for Silverman's test with `k` > 1 are included in the bottom part of Table 2. This example shows how to use the %silverman macro not only to classify the modality of the distribution, but also to estimate the number of modes of the original distribution. The first row displays the test of the null hypothesis that the original distribution (from which *twoNorms2* was sampled) has two or fewer modes against the alternative of three or more modes. The p value of 0.1882 corresponds to correctly failing to reject the null and concluding that the distribution has two or fewer modes. Recall that we obtained a p value of 0 when we did the Silverman test with *twoNorms2* and `k` $= 1$ (see Table 2) which made us suspect the distribution was multimodal. Combining these two pieces of information, we would argue the underlying distribution is likely bimodal, which we know to be correct. Similarly, for the *threeNorms* dataset, we reject the null hypothesis for `k` $= 2$. Yet, with a p value of 0.6607, we fail to reject the null hypothesis for `k` $= 3$. Therefore, we correctly suspect that the original distribution for *threeNorms* is trimodal.

4.2. Application to Iris Data

The next example demonstrates the use of the SAS macros on the *iris* data set [20], which is found in the `sashelp` library in the base SAS software and commonly used to demonstrate clustering tasks. The *iris* data set includes 150 flowers, each with 4 measurements: sepal length, sepal width, petal length, and petal width. Because the multimodality tests in both SAS and R are only designed for univariate data, we choose to examine the Petal Width. It is necessary to construct a new univariate data set containing only the Petal Width data. The `samples.sas` file, included with this paper, provides code to create this data set, denoted *irisPW*.

4.2.1. Dip Macro with Iris Data

The following call to the %dip macro executes the Dip test for unimodality on *irisPW*:

```
%dip(irisPW, include = "C:\Documents\SAS\dip.sas")
```

4.2.2. Silverman Macro with Iris Data

Calls to the %silverman macro for *irisPW* use the code shown below. The first two calls test whether *irisPW* was generated from a unimodal distribution. Both the unadjusted and adjusted versions of the Silverman Test are provided to illustrate how the results may differ. Additionally, the test is repeated for `k` > 1 to estimate the number of modes in the dataset.

```
%silverman(irisPW, setSeed = 1234, include =
"C:\Documents\SAS\silverman.sas")
%silverman(irisPW, adjust = 1, setSeed = 1234, include =
"C:\Documents\SAS\silverman.sas")
%silverman(irisPW, k = 2, setSeed = 1234, include =
"C:\Documents\SAS\silverman.sas")
```

4.2.3. Results from Iris Data

Results from the macro calls on *irisPW* are shown in Table 3. The p value associated with the Dip test is below 2.2×10^{-16}, which is extremely low. Table 3 also includes very low p values for both the unadjusted and adjusted versions of the Silverman Test for

`k` = 1. The null hypothesis that the Petal Widths came from a unimodal distribution is rejected for all three tests. The conclusion that the Petal Width distribution contains multiple modes is consistent with the fact that *iris* is known to consist of multiple clusters, corresponding to three different species of iris flowers [20].

[Table 3 near here.]

Additional calls to the Silverman Test facilitate estimation of the number of modes in *irisPW*. The use of `k` = 2 corresponds to a test of the null hypothesis that *irisPW* came from a distribution with two or fewer modes, against the alternative that it came from a distribution with three or more modes. The second row in Table 3 displays the results for `k` = 2. The only available option, the unadjusted Silverman test, yields a p value of 0.4775. We fail to reject the null hypothesis that the data has two or fewer modes. Consequently, we suspect that the Petal Width distribution is bimodal, not trimodal. Petal Width does vary by species. Indeed, it is smallest for *setosa* flowers compared to *versicolor* and *virginica*. However, the distance between the modes corresponding to *setosa* and *versicolor* species was much larger than the distance between the modes corresponding to the *versicolor* and *virginica* species [20]. Thus, Silverman's test found insufficient evidence to reject the null hypothesis that *irisPW* was generated from a distribution with two or fewer modes, likely because the distributions for the *versicolor* and *virginica* species were close together and difficult to distinguish.

5. Concluding Remarks

The present paper describes two macros, %dip and %silverman, to perform the Dip Test and Silverman Test, each of which tests the null hypothesis that a data set was generated from a unimodal distribution. We present the structure of the macros, include detailed instructions for how to run the macros and modify parameters to customise results, and display several examples of the macros in action. We also show how the %silverman macro can help estimate the number of modes in the distribution. Examples, including Gaussian mixtures and the *iris* Petal Width measurements, illustrate the capability of and differences between the two macros. Along with the example file (`samples.sas`), both macros are available and simple for users to download, install, and execute entirely within the SAS environment. Helpful tips for troubleshooting are provided in the appendix.

While our work adds new functionality for users of SAS software, it does have some limitations. One limitation of our work is its dependence on the `diptest` package and need to call R statements, thus requiring R to be installed on the machine performing the test. The adjusted version of the Dip test is also not included as an option in the SAS macro, because it is not available in the `diptest` package. In addition, the `RLANG` option cannot be readily changed in SAS Studio as part of SAS Online for Academics, or in the SAS University Edition software. These limitations make it impossible to run our macros in these popular environments.

Another potential limitation of our macro is its restriction to univariate data. Both the Dip and Silverman tests in R can technically accept multivariate data (in the form of an R matrix, for example). However, in their implementation, both tests perform data sorting, which merges all of the data into a single long vector. Often, the columns of multivariate data represent distinct variables measured on different scales. In such cases, collapsing the columns results in a single long vector with data whose combination lacks practical meaning. Any test performed on the combined vector yields misleading, even meaningless results. Because of this dangerous tendency, we

consider the limitation of our SAS macros to only one-dimensional SAS data sets an improvement over the R packages, which conduct tests on the inappropriately merged data without warning the user that columns were combined.

In fact, multivariate data can still be used with our macros if the user first reduces the data to a single dimension and then runs the multimodality tests on the reduced data set. For example, principal component analysis is a method to reduce data to a smaller dimension while retaining the original essence of the full data set [21]. Alternatively, clustering applications frequently summarise data using the set of distances between each pair of points in the data set. Examples of combining dimension reduction techniques with multimodality tests include the Dip test on the set of pairwise distances [5] and Silverman's test with the first principal component [6]. Additional combinations are being studied in an area of computer science called clusterability, which is an active research area about a critical but rarely used pre-validation step in the clustering process [22].

Implementation of these methods – and any new methods that utilise the Dip test or Silverman's test – in popular programming environments, such as SAS, is an avenue of future research that depends on the macros in the present paper.

We hope that users will find these macros useful for their SAS programming and research needs. Ultimately, however, we hope that the Dip and Silverman tests will be incorporated into base SAS software in the future to facilitate continuous maintenance and compatibility with newer releases of SAS, as well as access for users on all SAS interfaces.

References

[1] Bambach RK, Knoll AH, Wang SC. Origination, extinction, and mass depletions of marine diversity. Paleobiology. 2004;30(4):522–542.

[2] Cox GW. Strategic voting equilibria under the single nontransferable vote. American Political Science Review. 1994;88(3):608–621.

[3] Kousta ST, Vigliocco G, Vinson DP, et al. The representation of abstract words: why emotion matters. Journal of Experimental Psychology: General. 2011;140(1):14.

[4] Liu Y, Hayes DN, Nobel A, et al. Statistical significance of clustering for high-dimension, low-sample size data. Journal of the American Statistical Association. 2008;103(483):1281–1293. Available from: http://dx.doi.org/10.1198/016214508000000454.

[5] Kalogeratos A, Likas A. Dip-means: an incremental clustering method for estimating the number of clusters. In: Pereira F, Burges CJC, Bottou L, et al., editors. Advances in neural information processing systems 25. Curran Associates, Inc.; 2012. p. 2393–2401.

[6] Ahmed MO, Walther G. Investigating the multimodality of multivariate data with principal curves. Computational Statistics and Data Analysis. 2012;56(12):4462 – 4469. Available from: http://www.sciencedirect.com/science/article/pii/S0167947312001028.

[7] Shahbaba M, Beheshti S. Efficient unimodality test in clustering by signature testing. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); May; 2014. p. 8282–8286.

[8] Hartigan JA, Hartigan P. The dip test of unimodality. The Annals of Statistics. 1985; :70–84.

[9] Silverman BW. Using kernel density estimates to investigate multimodality. Journal of the Royal Statistical Society Series B (Methodological). 1981;:97–99.

[10] SAS Institute Inc. Sas/stat software, version 9.4. Cary, NC; 2012. Available from: http://www.sas.com/.

[11] Müller DW, Sawitzki G. Excess mass estimates and tests for multimodality. Journal of the American Statistical Association. 1991;86(415):738–746.
[12] Minnotte MC. A test of mode existence with applications to multimodality [dissertation]. Rice University; 1993.
[13] Cheng MY, Hall P. Calibrating the excess mass and dip tests of modality. Journal of the Royal Statistical Society: Series B (Statistical Methodology). 1998;60(3):579–589.
[14] Hall P, York M. On the calibration of silverman's test for multimodality. Statistica Sinica. 2001;:515–536.
[15] Schwaiger F, Holzmann H. Package which implements the silvermantest; 2013. Available from: https://www.mathematik.uni-marburg.de/~stochastik/R_packages/.
[16] Maechler M. diptest: Hartigan's dip test statistic for unimodality - corrected; 2016. R package version 0.75-7; Available from: https://CRAN.R-project.org/package=diptest.
[17] SAS Institute Inc. Sas/iml® 14.3 user's guide. Cary, NC; 2017. Available from: http://support.sas.com/documentation/onlinedoc/iml/143/imlug.pdf.
[18] SAS Institute Inc. Sas® 9.2 companion for windows, second edition. Cary, NC; 2010. Available from: http://support.sas.com/documentation/cdl/en/hostwin/63285/PDF/default/hostwin.pdf.
[19] SAS Institute Inc. Sas® 9.3 macro language: Reference. Cary, NC; 2011. Available from: http://support.sas.com/documentation/cdl/en/mcrolref/62978/PDF/default/mcrolref.pdf.
[20] Fisher RA. The use of multiple measurements in taxonomic problems. Annals of eugenics. 1936;7(2):179–188.
[21] Jolliffe I. Principal component analysis. Springer; 2002. Springer Series in Statistics; Available from: https://books.google.com/books?id=_olByCrhjwIC.
[22] Adolfsson A, Ackerman M, Brownstein N. To cluster, or not to cluster: How to answer the question; 2017. Available from: http://www.mayaackerman.info/pub/clusterability2017.pdf.
[23] SAS Institute Inc. Sas® 9.2 language reference: Dictionary, fourth edition. Cary, NC; 2011. Available from: http://support.sas.com/documentation/cdl/en/lrdict/64316/PDF/default/lrdict.pdf.

6. Appendix: Troubleshooting

The following information may be helpful to users when they are beginning to prepare and use the macro. Because we encountered these obstacles in our own testing, we provide the problems and solutions for others.

Q: I can't edit my SAS config file. What should I do?

A: Check the permissions on your computer. One option is to edit the config file in a plain text editor, such as Notepad, and run the editor as an administrator. (Right click the editor before opening to allow the option to run as administrator.) You should then be able to save the changes to your config file.

Alternatively, use the Save As feature of your text editor to save a new copy of the config file, and then navigate to the location of the original and replace it with the new file.

Make sure to use an ASCII text editor such as Notepad or a SAS text editor to avoid corrupting the config file. Using a specialised editor such as Microsoft Word or WordPad, for example, may corrupt the file [18, p. 14].

Q: I have one or more of the following errors:
ERROR: SAS could not initialize the R language interface.

ERROR: The installed version of R cannot be used. The entry point "ConsoleIob" could not be located.

What can I do?

A: One cause of these errors is having an out of date version of SAS installed on your computer. (For example, we encountered these errors when running SAS version 9.4 TS1M0, while the most current release at the time of writing was 9.4 TS1M5.) Please check your Log for your SAS release, and, if needed, upgrade your copy of SAS to the newest release.

Q: I ran the `dip.sas` and `silverman.sas` files and encountered errors. What happened?

A: You do not need to run the `dip.sas` and `silverman.sas` files. Doing so will generate errors due to undefined parameters. Running the `macrodefinitions.sas` file and properly calling the %dip and %silverman macros (see the `samples.sas` file for examples) is sufficient.

Q: I receive the following error when running the %dip (or %silverman) macros:

ERROR: File path for include parameter is not valid. Please use the full file path in quotes and include the extension ".sas". For example,
include = "C:\Documents\SAS\dip.sas"

But I *did* specify the file path. What can I do?

A: Make sure you have provided the full file path to `dip.sas` (or `silverman.sas`, depending on which macro is being used), including the ".sas" extension. A relative file path such as "dip.sas" will not work. The file path should also be surrounded by double quotes " ".

If you are using a fileref, such as in the `samples.sas` supplemental file, make sure to follow these same guidelines.

Q: I receive the following error when running the %dip (or %silverman) macros:

ERROR: Your data was input in SAS as numeric, but R did not recognise it as numeric. This may be due to the use of SAS formats, such as dates, times, or custom formats. If you decide that removing a format is appropriate, then documentation with directions is available at the following link: http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/viewer.htm#a000178212

What can I do?

A: This error is displayed in the Results Viewer when the provided data set is a SAS data set with numeric values, but R did not recognise the data as numeric. One cause of this error is using a SAS data set with a format applied to one or more of its variables. Certain data formats, such as date and time formats, can cause errors when running the %dip or %silverman macros. In this situation, removing the format may resolve the error. Code to remove a format using the `FORMAT` statement is shown below:

```
data mydata_without_formats;
    set mydataset;
    format x;
run;
```

where `mydataset` is the name of your original SAS data set, `x` is the name of the variable that has the format associated with it, and `data_without_formats` is the name of the newly created unformatted SAS data set. Additional information on SAS formats is available at the link provided in the error message, and in the SAS Language Reference [23, p. 1579].

7. Tables

Steps to Set Up the `dip` and `Silverman` Macros in SAS
First Time
1. Ensure that R is installed on the machine.
2. Open the `SASV9.cfg` file and ensure that the `RLANG` option has been inserted.
3. If using the %dip macro or the adjusted option of the %silverman macro: Verify that the machine can install new R packages from the Internet or that the appropriate R packages are installed on the machine.
4. Download the supplemental code files and store `dip.sas` and `silverman.sas` in a safe location for future use.
Every Time
5. Execute the `macroDefinitions.sas` file to add the macros to the operating environment.
6. From here, the macros can be used as described in the paper.

Table 1. Instructions for setting up SAS environment to run the macros.

Data Set Name	**Description**	**k**	**Dip**	**Silverman**	**Silverman (adjusted)**
oneNorm	One Normal	1	0.7542	0.5105	0.4351
twoNorms1	Two Normals (Close)	1	0.3146	0.1361	0.0546
twoNorms2	Two Normals (Far)	1	2.032×10^{-6}	$0^{\dagger}$	0^{*}
threeNorms	Three Normals	1	0.0094	0.0591	0.0118
twoNorms2	Two Normals (Far)	2	N/A	0.1882	N/A
threeNorms	Three Normals	2	N/A	0.0010	N/A
threeNorms	Three Normals	3	N/A	0.6607	N/A

Table 2. Summary of output (p values) from running the %dip and %silverman SAS macros with various simulations from Normal distributions.

† For the unadjusted test, the p value is a bootstrap p value and is not rounded.

* For the adjusted test only, the %silverman macro rounds any p value below 0.005 down to 0.

Data Set Name	**Description**	**k**	**Dip**	**Silverman**	**Silverman (adjusted)**
irisPW	Petal Width	1	$< 2.2 \times 10^{-16}$	$0.0030^{\dagger}$	0^{*}
irisPW	Petal Width	2	N/A	0.4775	N/A

Table 3. Summary of results from running the %silverman and %dip SAS macros with the *iris* Petal Width feature.

† For the unadjusted test, the p value is a bootstrap p value and is not rounded.

* For the adjusted test only, the %silverman macro rounds any p value below 0.005 down to 0.

8. Figures

```
/* set default locations                                              */
-TRAINLOC ""

/* set the default fileref for the PARMCARDS= option                  */
-SET FT15F001  'FT15F001.DAT'

/* set the RLANG option to enable execution of R code */
-RLANG
```

Figure 1. The location of the `RLANG` option in the `SASV9.cfg` file

Dip Test: oneNorm

```
        Hartigans' dip test for unimodality / multimodality

data:  as.matrix(dipdata)
D = 0.018204, p-value = 0.7542
alternative hypothesis: non-unimodal, i.e., at least bimodal
```

Figure 2. Results from the %dip SAS macro test of unimodality on the *oneNorm* data set.

Silverman Test: oneNorm

```
Silverman Test

Seed set in R to: 1234

Using adjusted p-values. See Hall and York

HO: number of modes <=  1
p-value: 0.435056
```

Figure 3. Results from the %silverman macro on *oneNorm*, using the adjusted version from Hall and York [14].

9. Figure Captions

Figure 1: The location of the `RLANG` option in the `SASV9.cfg` file

Figure 2: Results from the %dip SAS macro test of unimodality on the *oneNorm* data set.

Figure 3: Results from the %silverman macro on *oneNorm*, using the adjusted version from Hall and York [14].