**Package Validation for “procs”**

**Background:** The “procs” package simulates some SAS® procedures in R. The package was written to make it easier for SAS® programmers to work in R. The package was also written to align statistical output between SAS® and R, as the native R functions do not necessarily match SAS®. This “pre-validation” will reduce the number of discrepancies in statistical output, and speed the process of analysis.

**Purpose:** The purpose of this document is to validate the R “procs” package against equivalent SAS® output.

**Methodology:** For this validation, first a test was defined that could be accomplished in both SAS® and R. Then a programmer wrote a small script in each language that produced similar results. The results were compared visually to ensure a match. Any discrepancies would be investigated and resolved either by modifying the comparison scripts, or modifying the “procs” package code until it matched SAS®. Comparison code and results were documented.

**SAS Version:** SAS Studio 3.81

**R Version:** 4.3.2

**Procs Version:** 1.0.6

**Author:** David J. Bosak, Chief Software Architect, Archytas Clinical Solutions

**Contributor:** Yifei Chen

**Last Modified Date:** 3/02/2024

## Change History

|  |  |  |
| --- | --- | --- |
| **Date** | **Item** | **Procs Version** |
| 2023/06/23 | Initial version. | 0.0.9007 |
| 2023/06/24 | Created first few tests for each function. | 0.0.9007 |
| 2023/07/05 | Discovered discrepancy in Procs-means-009 Upper 95% confidence limit. Had to change code to add alpha option and one-sided confidence limit calculation. | 0.0.9007 |
| 2023/07/08 | Resolved discrepancy on confidence limits. Added test means-010 to confirm two-sided CL. | 0.0.9008 |
| 2023/07/10 | Added Procs-means-011 tests for means t-test and p-value. | 0.0.9008 |
| 2023/07/11 | Added Procs-means-012 test for Uncorrected Sum of Squares. | 0.0.9008 |
| 2023/07/20 | Added Procs-means-013 test for skewness and kurtosis. | 0.0.9008 |
| 2023/07/25 | Added ability to use factors for sorting proc\_freq output. | 0.0.9008 |
| 2023/07/26 | Fixed Fisher’s exact test for Cell 1.1. This was off when table was sorted properly. | 0.0.9008 |
| 2023/07/29 | Incremented to version 1.0.0 for CRAN submission. | 1.0.0 |
| 2023/08/08 | Incremented to version 1.0.2 for CRAN fixes | 1.0.2 |
| 2023/08/10 | Fixed zero-fill row on proc\_freq() when user supplied a factor. | 1.0.2 |
| 2023/08/11 | Fixed n-way class summaries on proc\_means() when multiple class vars are present. | 1.0.2 |
| 2023/08/13 | Fixed some labels on proc\_means() report output. | 1.0.2 |
| 2023/08/18 | Add means-015 for more complicated statistics with by variable. | 1.0.2 |
| 2023/09/04 | Upgrade version and prepare for CRAN submission. | 1.0.3 |
| 2023/12/17 | Add validation for proc\_ttest(). | 1.0.4 |
| 2023/12/20 | Reran all comparisons and made a few fixes. | 1.0.4 |
| 2024/01/20 | Bug fixes. No changes to validation. | 1.0.5 |
| 2024/03/02 | Added validation for proc\_reg(). | 1.0.6 |

## Freq Data

|  |  |
| --- | --- |
| **SAS** | **R** |
| data Color;  input Region Eyes $ Hair $ Count @@;  label Eyes ='Eye Color'  Hair ='Hair Color'  Region='Geographic Region';  datalines;  1 blue fair 23 1 blue red 7 1 blue medium 24  1 blue dark 11 1 green fair 19 1 green red 7  1 green medium 18 1 green dark 14 1 brown fair 34  1 brown red 5 1 brown medium 41 1 brown dark 40  1 brown black 3 2 blue fair 46 2 blue red 21  2 blue medium 44 2 blue dark 40 2 blue black 6  2 green fair 50 2 green red 31 2 green medium 37  2 green dark 23 2 brown fair 56 2 brown red 42  2 brown medium 53 2 brown dark 54 2 brown black 13  ; | dat <- read.table(header = TRUE, text = '  Region Eyes Hair Count  1 blue fair 23  1 blue dark 11  1 green medium 18  1 brown red 5  1 brown black 3  2 blue medium 44  2 green fair 50  2 green dark 23  2 brown medium 53  1 blue red 7  1 green fair 19  1 green dark 14  1 brown medium 41  2 blue fair 46  2 blue dark 40  2 green red 31  2 brown fair 56  2 brown dark 54  1 blue medium 24  1 green red 7  1 brown fair 34  1 brown dark 40  2 blue red 21  2 blue black 6  2 green medium 37  2 brown red 42  2 brown black 13  ') |
| \*\* Above input data will be used for all frequency comparisons unless data is provided in the script itself. | |

## proc\_freq-001

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc freq data=Color;  tables Eyes Hair Eyes\*Hair / out=FreqCount outexpect sparse;  weight Count;  title 'Eye and Hair Color of European Children';  run; |  |
| **R Code and Output** |  |
| library(procs)  proc\_freq(dat, tables = v(Eyes, Hair, Eyes \* Hair), weight = Count,  title = "Eye and Hair Color of European Children") |  |
| **Comparison** | **Results** |
| Compare basic one and two way frequencies with weight option. | Pass |

## proc\_freq-002

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc freq data=Color nlevels;  tables Eyes Hair Eyes\*Hair / out=FreqCount outexpect sparse;  title 'Eye and Hair Color of European Children';  run; |  |
| **R Code and Output** |  |
| proc\_freq(dat, tables = v(Eyes, Hair, Eyes \* Hair),  title = "Eye and Hair Color of European Children",  options = v(nlevels)) |  |
| **Comparison** | **Result** |
| Compare basic frequencies with no weight and nlevels option. | Pass |

## proc\_freq-003

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| data SummerSchool;  input Gender $ Internship $ Enrollment $ Count @@;  datalines;  boys yes yes 35 boys no no 27  boys no yes 14 girls yes no 10  girls yes yes 32 girls no no 23  girls no yes 53 girls . no 25  boys yes . 29 girls . no 29  ;  proc freq data=SummerSchool order=data nlevels;  tables Internship / out = Fork missing;  options missing=. mi;  run; |  |
| **R Code and Output** |  |
| prtm <- read.table(header = TRUE, text = '  sex internship enrollment count  1 boys yes yes 35  2 boys no yes 14  3 girls yes yes 32  4 girls no yes 53  5 boys yes NA 29  6 boys no no 27  7 girls yes no 10  8 girls no no 23  9 girls NA yes 25  10 girls NA no 29')  res <- proc\_freq(prtm, tables = v(internship),  options = v(nlevels, missing)) |  |
| **Comparison** | **Result** |
| Compare basic frequencies with nlevels and missing options. | Pass |

## proc\_freq-004

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| data SummerSchool;  input Gender $ Internship $ Enrollment $ Count @@;  datalines;  boys yes yes 35 boys yes no 29  boys no yes 14 boys no no 27  girls yes yes 32 girls yes no 10  girls no yes 53 girls no no 23  ;  proc freq data=SummerSchool order=data;  tables Internship \* Enrollment / out = Fork chisq;  by Gender;  weight Count;  run; | |  |  | | --- | --- | |  |  | |
| **R Code and Output** |  |
| prt <- read.table(header = TRUE, text = '  sex internship enrollment count  1 boys yes yes 35  2 boys no yes 14  3 girls yes yes 32  4 girls no yes 53  5 boys yes no 29  6 boys no no 27  7 girls yes no 10  8 girls no no 23')  prt$enrollment <- factor(prt$enrollment, c("yes", "no"))  prt$internship <- factor(prt$internship, c("yes", "no"))  proc\_freq(prt, tables = "internship \* enrollment",  options = ChiSq,  by = "sex",  weight = "count") |  |
| **Comparison** | **Result** |
| Compare two way with chisquare statistic. | Pass. |

## proc\_freq-005

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| data SummerSchool;  input Gender $ Internship $ Enrollment $ Count @@;  datalines;  boys yes yes 35 boys yes no 29  boys no yes 14 boys no no 27  girls yes yes 32 girls yes no 10  girls no yes 53 girls no no 23  ;  proc freq data=SummerSchool order=data;  tables Internship \* Enrollment / out = Fork chisq;  by Gender;  weight Count;  run; | |  |  | | --- | --- | |  |  | |
| **R Code and Output** |  |
| prt <- read.table(header = TRUE, text = '  sex internship enrollment count  1 boys yes yes 35  2 boys no yes 14  3 girls yes yes 32  4 girls no yes 53  5 boys yes no 29  6 boys no no 27  7 girls yes no 10  8 girls no no 23')  prt$enrollment <- factor(prt$enrollment, c("yes", "no"))  prt$internship <- factor(prt$internship, c("yes", "no"))  proc\_freq(prt, tables = "internship \* enrollment",  options = Fisher,  by = "sex",  weight = "count") |  |
| **Comparison** | **Result** |
| Compare two way with Fisher’s Exact statistic. | Pass. |

## Means Data

|  |  |
| --- | --- |
| **SAS** | **R** |
| data cake;  input LastName $ 1-12 Age 13-14 PresentScore 16-17  TasteScore 19-20 Flavor $ 23-32 Layers 34 ;  datalines;  Orlando 27 93 80 Vanilla 1  Ramey 32 84 72 Rum 2  Goldston 46 68 75 Vanilla 1  Roe 38 79 73 Vanilla 2  Larsen 23 77 84 Chocolate 3  Davis 51 86 91 Spice 3  Strickland 19 82 79 Chocolate 1  Nguyen 57 77 84 Vanilla 3  Hildenbrand 33 81 83 Chocolate 1  Byron 62 72 87 Vanilla 2  Sanders 26 56 79 Chocolate 1  Jaeger 43 66 74 1  Davis 28 69 75 Chocolate 2  Conrad 69 85 94 Vanilla 1  Walters 55 67 72 Chocolate 2  Rossburger 28 78 81 Spice 2  Matthew 42 81 92 Chocolate 2  Becker 36 62 83 Spice 2  Anderson 27 87 85 Chocolate 1  Merritt 62 73 84 Chocolate 1  ; | datm <- read.table(header = TRUE, text = '  LastName Age PresentScore TasteScore Flavor Layers  Orlando 27 93 80 Vanilla 1  Ramey 32 84 72 Rum 2  Goldston 46 68 75 Vanilla 1  Roe 38 79 73 Vanilla 2  Larsen 23 77 84 Chocolate 3  Davis 51 86 91 Spice 3  Strickland 19 82 79 Chocolate 1  Nguyen 57 77 84 Vanilla 3  Hildenbrand 33 81 83 Chocolate 1  Byron 62 72 87 Vanilla 2  Sanders 26 56 79 Chocolate 1  Jaeger 43 66 74 NA 1  Davis 28 69 75 Chocolate 2  Conrad 69 85 94 Vanilla 1  Walters 55 67 72 Chocolate 2  Rossburger 28 78 81 Spice 2  Matthew 42 81 92 Chocolate 2  Becker 36 62 83 Spice 2  Anderson 27 87 85 Chocolate 1  Merritt 62 73 84 Chocolate 1  ') |
| \*\* Above input data will be used for all means comparisons unless data is provided in the script itself. | |

## proc\_means-001

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc means data=cake n mean median stddev min max maxdec=4;  var Age PresentScore TasteScore layers;  title 'Summary of Presentation and Taste Scores';  output out=outdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datm, var = v(Age, PresentScore, TasteScore, Layers),  stats = c("n", "mean", "median", "std", "min", "max"),  options = v(maxdec = 4),  titles = "Summary of Presentation and Taste Scores") |  |
| **Comparison** | **Result** |
| Simple Proc means on multiple variables. | Pass |

## proc\_means-002

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc means data=cake nmiss median mode lclm uclm stderr maxdec=4;  var Age PresentScore TasteScore layers;  title 'Summary of Presentation and Taste Scores';  output out=outdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datm, v(Age, PresentScore, TasteScore, Layers),  stats = v(nmiss, median, mode, clm, stderr),  options = v(maxdec = 4),  titles = "Summary of Presentation and Taste Scores") |  |
| **Comparison** | **Result** |
| Comparison of more statistics options. | Pass |

## proc\_means-003

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc sort data=cake out=cake2;  by Layers;  run;  proc means data=cake2 n mean median stddev min max maxdec=4;  var Age PresentScore TasteScore;  title 'Summary of Presentation and Taste Scores';  by Layers;  output out=outdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datm, var = v(Age, PresentScore, TasteScore),  stats = c("n", "mean", "median", "std", "min", "max"),  by = "Layers",  options = v(maxdec = 4),  titles = "Summary of Presentation and Taste Scores") |  |
| **Comparison** | **Result** |
| Comparison of by variable. | Pass |

## proc\_means-004

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc sort data=cake out=cake2;  by Layers;  run;  proc means data=cake2 n mean median stddev min max maxdec=4;  var Age PresentScore TasteScore;  title 'Summary of Presentation and Taste Scores';  class Layers;  output out=outdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datm, var = v(Age, PresentScore, TasteScore),  stats = c("n", "mean", "median", "std", "min", "max"),  class = "Layers",  options = v(maxdec = 4),  titles = "Summary of Presentation and Taste Scores") |  |
| **Comparison** | **Result** |
| Comparison of class variable. | Pass |

## proc\_means-005

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| data cake2;  set cake;  if (\_n\_ = 5) then do;  PresentScore = .;  end;  run;  proc means data=cake2 n nmiss mean median lclm uclm stddev maxdec=4;  var PresentScore TasteScore;  title 'Summary of Presentation and Taste Scores';  output out=outdata;  run; |  |
| **R Code and Output** |  |
| datm2 <- datm  datm2[5, "PresentScore"] <- NA  proc\_means(datm2, var = c("PresentScore", "TasteScore"),  stats = c("n", "nmiss", "mean", "median", "mode", "clm", "std"),  titles = "My first title for Means",  options = v(notype, nonobs, maxdec=4)) |  |
| **Comparison** | **Result** |
| Comparison of class variable. | Pass |

## proc\_means-006

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc means data=cake css cv lclm mode n maxdec=4;  var Age PresentScore TasteScore layers;  title 'Summary of Presentation and Taste Scores';  output out=outdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datm, var = v(Age, PresentScore, TasteScore, Layers),  stats = c("css", "cv", "lclm", "mode", "nobs"),  options = v(maxdec = 4),  titles = "Summary of Presentation and Taste Scores") |  |
| **Comparison** | **Result** |
| Comparison of more statistics options. | Pass, except SAS doesn’t have NOBS keyword. |

## proc\_means-007

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc means data=cake p1 p5 p10 p20 p25 p30 p40 p50 maxdec=4;  var Age PresentScore TasteScore layers;  title 'Summary of Presentation and Taste Scores';  output out=outdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datm, var = v(Age, PresentScore, TasteScore, Layers),  stats = c("p1", "p5", "p10", "p20", "p25", "p30", "p40", "p50"),  options = v(maxdec = 4),  titles = "Summary of Presentation and Taste Scores") |  |
| **Comparison** | **Result** |
| Comparison of more statistics options. | Pass |

## proc\_means-008

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc means data=cake p60 p70 p75 p80 p90 p95 p99 maxdec=4;  var Age PresentScore TasteScore layers;  title 'Summary of Presentation and Taste Scores';  output out=outdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datm, var = v(Age, PresentScore, TasteScore, Layers),  stats = c("p60", "p70", "p75", "p80", "p90", "p95", "p99"),  options = v(maxdec = 4),  titles = "Summary of Presentation and Taste Scores") |  |
| **Comparison** | **Result** |
| Comparison of more statistics options. | Pass |

## proc\_means-009

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc means data=cake q1 q3 qrange range sum uclm var maxdec=4;  var Age PresentScore TasteScore layers;  title 'Summary of Presentation and Taste Scores';  output out=outdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datm, var = v(Age, PresentScore, TasteScore, Layers),  stats = c("q1", "q3", "qrange", "range", "sum", "uclm", "vari"),  options = v(maxdec = 4),  titles = "Summary of Presentation and Taste Scores") |  |
| **Comparison** | **Result** |
| Comparison of more statistics options. | Pass |

## proc\_means-010

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc means data=cake mean clm alpha = 0.1 maxdec=4;  var PresentScore TasteScore;  title 'Summary of Presentation and Taste Scores';  output out=outdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datm, var = v(PresentScore, TasteScore),  stats = v(mean, clm),  options = v(alpha = 0.1, maxdec=4),  titles = "Summary of Presentation and Taste Scores") |  |
| **Comparison** | **Result** |
| Comparison of 90% confidence limit with alpha option. | Pass |

## proc\_means-011

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| DATA WEIGHT;  INPUT WBEFORE WAFTER;  \* Calculate WLOSS in the DATA step \*;  WLOSS=WAFTER-WBEFORE;  DATALINES;  200 190  175 154  188 176  198 193  197 198  310 240  245 204  202 178  ;  run;    PROC MEANS N MEAN T PRT;  VAR WLOSS;  TITLE 'Paired t-test example using PROC MEANS';  RUN; |  |
| **R Code and Output** |  |
| datp <- read.table(header = TRUE, text = '  WBEFORE WAFTER  200 190  175 154  188 176  198 193  197 198  310 240  245 204  202 178')  datp$WLOSS <- datp$WAFTER - datp$WBEFORE  res <- proc\_means(datp, var = WLOSS,  stats = v(n, mean, t, prt),  titles = c("Paired t-test example")) |  |
| **Comparison** | **Result** |
| Comparison of hypothesis tests. | Pass |

## proc\_means-012

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc means data=cake n mean uss;  var Age PresentScore TasteScore Layers;  title 'Summary of Presentation and Taste Scores';  output out=outdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datm, var = v(Age, PresentScore, TasteScore, Layers),  stats = v(n, mean, uss),  titles = c("Test")) |  |
| **Comparison** | **Result** |
| Test USS statistics. | Pass |

## proc\_means-013

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc means data=cake n mean skewness kurtosis maxdec = 8;  var Age PresentScore TasteScore Layers;  title 'Summary of Presentation and Taste Scores';  output out=outdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datm, var = v(Age, PresentScore, TasteScore, Layers),  stats = v(n, mean, skew, kurt),  titles = c("Test"),  options = c(maxdec = 8)) |  |
| **Comparison** | **Result** |
| Test Skewness and Kurtosis | Pass |

## proc\_means-014

|  |  |
| --- | --- |
| **SAS Data** | **R Data** |
| data cake2;  input LastName $ 1-12 Age 13-14 PresentScore 16-17  TasteScore 19-20 Flavor $ 23-32 Layers 34 Region 36;  datalines;  Orlando 27 93 80 Vanilla 1 1  Ramey 32 84 72 Rum 2 1  Goldston 46 68 75 Vanilla 1 1  Roe 38 79 73 Vanilla 2 1  Larsen 23 77 84 Chocolate 3 1  Davis 51 86 91 Spice 3 1  Strickland 19 82 79 Chocolate 1 1  Nguyen 57 77 84 Vanilla 3 1  Hildenbrand 33 81 83 Chocolate 1 1  Byron 62 72 87 Vanilla 2 1  Sanders 26 56 79 Chocolate 1 2  Jaeger 43 66 74 1 2  Davis 28 69 75 Chocolate 2 2  Conrad 69 85 94 Vanilla 1 2  Walters 55 67 72 Chocolate 2 2  Rossburger 28 78 81 Spice 2 2  Matthew 42 81 92 Chocolate 2 2  Becker 36 62 83 Spice 2 2  Anderson 27 87 85 Chocolate 1 2  Merritt 62 73 84 Chocolate 1 2  ; | datmr <- read.table(header = TRUE, text = '  LastName Age PresentScore TasteScore Flavor Layers Region  Orlando 27 93 80 Vanilla 1 1  Ramey 32 84 72 Rum 2 1  Goldston 46 68 75 Vanilla 1 1  Roe 38 79 73 Vanilla 2 1  Larsen 23 77 84 Chocolate 3 1  Davis 51 86 91 Spice 3 1  Strickland 19 82 79 Chocolate 1 1  Nguyen 57 77 84 Vanilla 3 1  Hildenbrand 33 81 83 Chocolate 1 1  Byron 62 72 87 Vanilla 2 1  Sanders 26 56 79 Chocolate 1 2  Jaeger 43 66 74 NA 1 2  Davis 28 69 75 Chocolate 2 2  Conrad 69 85 94 Vanilla 1 2  Walters 55 67 72 Chocolate 2 2  Rossburger 28 78 81 Spice 2 2  Matthew 42 81 92 Chocolate 2 2  Becker 36 62 83 Spice 2 2  Anderson 27 87 85 Chocolate 1 2  Merritt 62 73 84 Chocolate 1 2  ') |
| **SAS Code and Output** |  |
| proc means data=cake2 n min max mean stddev maxdec=4;  var Age PresentScore TasteScore;  title 'Summary of Presentation and Taste Scores';  class Region Layers;  output out=mdata;  run; |  |
| **R Code and Output** |  |
| res <- proc\_means(datmr,  var = c("Age", "PresentScore", "TasteScore"),  stats = c("n", "min", "max","mean", "std"),  output = c("out", "report"),  class = c("Region", "Layers"),  titles = "My first title for Means",  options = c("long", maxdec = 4)) |  |
| **Comparison** | **Result** |
| Test multiple class variables | Pass |

## proc\_means-015

|  |  |
| --- | --- |
| **SAS Data** | **R Data** |
| data cake;  input LastName $ 1-12 Age 13-14 PresentScore 16-17  TasteScore 19-20 Flavor $ 23-32 Layers 34 Region 36;  datalines;  Orlando 27 93 80 Vanilla 3 1  Ramey 32 84 72 Rum 2 1  Goldston 46 68 75 Vanilla 1 1  Roe 38 79 73 Vanilla 2 1  Larsen 23 77 84 Chocolate 3 1  Davis 51 86 91 Spice 3 1  Strickland 19 82 79 Chocolate 1 1  Nguyen 57 77 84 Vanilla 3 1  Hildenbrand 33 81 83 Chocolate 1 1  Byron 62 72 87 Vanilla 2 1  Sanders 26 56 79 Chocolate 1 1  Jaeger 43 66 74 1 2  Davis 28 69 75 Chocolate 2 2  Conrad 69 85 94 Vanilla 1 2  Walters 55 67 72 Chocolate 2 2  Rossburger 28 78 81 Spice 2 2  Matthew 42 81 92 Chocolate 2 2  Becker 36 62 83 Spice 2 2  Anderson 27 87 85 Chocolate 1 2  Merritt 62 73 84 Chocolate 1 2  ; | datsp <- datm  datsp$Layers[1] <- 3 |
| **SAS Code and Output** |  |
| proc sort data=cake out=cake2;  by Layers;  run;  proc means data=cake2 kurt skew cv clm maxdec = 8;  var Age PresentScore TasteScore;  title 'Summary of Presentation and Taste Scores';  by Layers;  output out=mdata;  run; |  |
| **R Code and Output** |  |
| proc\_means(datsp, var = c("Age", "PresentScore", "TasteScore"),  stats = c("kurtosis", "skew", "cv", "clm"),  by = c("Layers"),  options = c(maxdec = 8)  ) |  |
| **Comparison** | **Result** |
| Test more complicated statistics with by variables and low sample sizes. | Pass |

## Transpose Data

|  |  |
| --- | --- |
| **SAS** | **R** |
| data score;  input Student $9. +1 StudentID $ Section $ Test1 Test2 Final;  datalines;  Capalleti 0545 1 94 91 87  Dubose 1252 2 51 65 91  Engles 1167 1 95 97 97  Grant 1230 2 63 75 80  Krupski 2527 2 80 76 71  Lundsford 4860 1 92 40 86  McBane 0674 1 75 78 72  ; | score <- read.table(header = TRUE,  colClasses = c(Student = "character", StudentID = "character",  Section = "character"), text = '  Student StudentID Section Test1 Test2 Final  Capalleti "0545" "1" 94 91 87  Dubose "1252" "2" 51 65 91  Engles "1167" "1" 95 97 97  Grant "1230" "2" 63 75 80  Krupski "2527" "2" 80 76 71  Lundsford "4860" "1" 92 40 86  McBane "0674" "1" 75 78 72  ', stringsAsFactors = FALSE) |

## proc\_transpose-001

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc transpose data=score out=score\_transposed name = Exam;  id Student;  run;  proc print data=score\_transposed noobs;  title 'Student Test Scores in Variables';  run; |  |
| **R Code and Output** |  |
| res <- proc\_transpose(score, var = c("Test1", "Test2", "Final"),  id = Student, name = "Exam")  proc\_print(res, titles = "Student Test Scores in Variables") |  |
| **Comparison** | **Results** |
| Perform basic transpose with ID variable and name. | Pass |

## proc\_transpose-002

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc transpose data=score out=score\_transposed  prefix = Student suffix= score name = Exam;  run;  proc print data=score\_transposed noobs;  title 'Student Test Scores in Variables';    run; |  |
| **R Code and Output** |  |
| res <- proc\_transpose(score, var = c("Test1", "Test2", "Final"),  name = "Exam", prefix = "Student", suffix = "score")  proc\_print(res, titles = "Student Test Scores in Variables") |  |
| **Comparison** | **Results** |
| Perform basic transpose with prefix and suffix. | Pass |

## proc\_transpose-003

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc sort data = score out = score\_sorted;  by Section;  run;  proc transpose data=score\_sorted out=score\_transposed name = Exam;  id Student;  by Section;  run;  proc print data=score\_transposed noobs;  title 'Student Test Scores in Variables';    run; |  |
| **R Code and Output** |  |
| res <- proc\_transpose(score, var = c("Test1", "Test2", "Final"),  name = "Exam", id = "Student", by = "Section")  proc\_print(res, titles = "Student Test Scores in Variables") |  |
| **Comparison** | **Results** |
| Perform basic transpose with by group. | Pass |

## proc\_transpose-004

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc means data = score nonobs; output out = stats; run;  data score2 (keep = Group \_STAT\_ Test1 Test2 Final) ; set stats (drop =\_TYPE\_ \_FREQ\_); Group = "Group1"; run;  proc print data=score2 noobs;  title 'Student Test Scores in Variables';   run; |  |
| **R Code and Output** |  |
| stats <- proc\_means(score, options = v(notype, nonobs))  res1 <- data.frame(Group = "Group1", stats)  res2 <- proc\_transpose(res1, copy = "Group",  name = "STAT", id = "VAR")  proc\_print(res2, titles = "Student Test Scores in Variables") |  |
| **Comparison** | **Results** |
| Perform basic transpose with copy parameter. | Pass. Differences that SAS proc means puts stats in rows anyway, as doesn’t need to be transformed. Also, data step does not reorder columns. Procs package tries to improve on both of these issues. |

## T-Test Data

|  |  |
| --- | --- |
| **SAS** | **R** |
| data cls; set sashelp.class; if \_n\_ < 10 then do; region = 'A'; end; else do; region = 'B'; end;  run; | cls <- read.table(header = TRUE, text = '  Name Sex Age Height Weight region  Alfred M 14 69.0 112.5 A  Alice F 13 56.5 84.0 A  Barbara F 13 65.3 98.0 A  Carol F 14 62.8 102.5 A  Henry M 14 63.5 102.5 A  James M 12 57.3 83.0 A  Jane F 12 59.8 84.5 A  Janet F 15 62.5 112.5 A  Jeffrey M 13 62.5 84.0 A  John M 12 59.0 99.5 B  Joyce F 11 51.3 50.5 B  Judy F 14 64.3 90.0 B  Louise F 12 56.3 77.0 B  Mary F 15 66.5 112.0 B  Philip M 16 72.0 150.0 B  Robert M 12 64.8 128.0 B  Ronald M 15 67.0 133.0 B  Thomas M 11 57.5 85.0 B  William M 15 66.5 112.0 B') |

## proc\_ttest-001

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc ttest data=cls  h0=65  alpha=0.05;  var Height;  run; |  |
| **R Code and Output** |  |
| proc\_ttest(cls,  var = c("Height"),  options = c("h0" = 65, "alpha" = 0.05)) |  |
| **Comparison** | **Results** |
| Perform basic T-Test on single variable. | Pass. |

## proc\_ttest-002

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc ttest data=cls alpha=0.05;  paired weight \* height; run; |  |
| **R Code and Output** |  |
| proc\_ttest(cls,  paired = "Weight \* Height",  options = c("alpha" = 0.05)) |  |
| **Comparison** | **Results** |
| Perform basic T-Test on paired variables. | Pass. |

## proc\_ttest-003

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc ttest data=cls alpha=0.05;  class sex;  var height; run; |  |
| **R Code and Output** |  |
| proc\_ttest(cls,  var = "Height",  class = "Sex",  options = c("alpha" = 0.05)) |  |
| **Comparison** | **Results** |
| Perform basic T-Test on variable with class. | Pass, except std and std CL not available for Pooled CL. |

## proc\_ttest-004

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc ttest data=cls  h0=65  alpha=0.1;  var Height;  by region;  run; |  |
| **R Code and Output** |  |
| proc\_ttest(cls,  var = c("Height"),  by = "region",  options = c("h0" = 65, "alpha" = 0.05)) |  |
| **Comparison** | **Results** |
| Perform T-Test on single variable with by. | Pass. |

## proc\_ttest-005

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc ttest data=cls alpha=0.1;  paired weight \* height;  by region;  run; |  |
| **R Code and Output** |  |
| proc\_ttest(cls,  paired = "Weight \* Height",  by = "region",  options = c("alpha" = 0.1)) |  |
| **Comparison** | **Results** |
| Perform T-Test on paired variables with by. | Pass. |

## proc\_ttest-006

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc ttest data=cls alpha=0.1;  class sex;  var height;  by region;  run; |  |
| **R Code and Output** |  |
| proc\_ttest(cls,  var = "Height",  class = "Sex",  by = "region",  options = c("alpha" = 0.1)) |  |
| **Comparison** | **Results** |
| Perform T-Test on unpaired variable with by. | Pass. Initially F Value and CL values were not coming out. |

## proc\_ttest-007

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc ttest data=cls alpha=0.1;  class sex;  var Weight;  by region;  run; |  |
| **R Code and Output** |  |
| proc\_ttest(cls,  var = "Weight",  class = "Sex",  by = "region",  options = c("alpha" = 0.1)) |  |
| **Comparison** | **Results** |
| Perform T-Test on unpaired variable with by. | Pass. More testing to ensure it is coming out with different variables. |

## proc\_ttest-008

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc ttest data=cls alpha=0.1;  class sex;  var Age;  by region;  run; |  |
| **R Code and Output** |  |
| proc\_ttest(cls,  var = "Age",  class = "Sex",  by = "region",  options = c("alpha" = 0.1)) |  |
| **Comparison** | **Results** |
| Perform T-Test on unpaired variable with by. | Pass. Another test with a different variable. |

## proc\_ttest-009A

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc ttest data=cls alpha=0.1;  class sex;  var Height Weight Age;  by region;  run; |  |
| **R Code and Output** |  |
| proc\_ttest(cls,  var = c("Height", "Weight", "Age"),  class = "Sex",  by = "region",  options = c("alpha" = 0.1)) |  |
| **Comparison** | **Results** |
| Perform T-Test on multiple unpaired variable with by. | Pass. |

## proc\_ttest-009B

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc ttest data=cls alpha=0.1;  class sex;  var Height Weight Age;  by region;  run; |  |
| **R Code and Output** |  |
| proc\_ttest(cls,  var = c("Height", "Weight", "Age")  class = "Sex",  by = "region",  options = c("alpha" = 0.1)) |  |
| **Comparison** | **Results** |
| Perform T-Test on multiple unpaired variable with by. | Pass. |

## Reg Data

|  |  |
| --- | --- |
| **SAS** | **R** |
| datalines;  44 89.47 44.609 11.37 40 75.07 45.313 10.07  44 85.84 54.297 8.65 42 68.15 59.571 8.17  38 89.02 49.874 9.22 47 77.45 44.811 11.63  40 75.98 45.681 11.95 43 81.19 49.091 10.85  44 81.42 39.442 13.08 38 81.87 60.055 8.63  44 73.03 50.541 10.13 45 87.66 37.388 14.03  45 66.45 44.754 11.12 47 79.15 47.273 10.60  54 83.12 51.855 10.33 49 81.42 49.156 8.95  51 69.63 40.836 10.95 51 77.91 46.672 10.00  48 91.63 46.774 10.25 49 73.37 50.388 10.08  57 73.37 39.407 12.63 54 79.38 46.080 11.17  52 76.32 45.441 9.63 50 70.87 54.625 8.92  51 67.25 45.118 11.08 54 91.63 39.203 12.88  51 73.71 45.790 10.47 57 59.08 50.545 9.93  49 76.32 48.673 9.40 48 61.24 47.920 11.50  52 82.78 47.467 10.50  ; | fitness <- read.table(header = TRUE, text = '  Age Weight Oxygen RunTime  44 89.47 44.609 11.37  40 75.07 45.313 10.07  44 85.84 54.297 8.65  42 68.15 59.571 8.17  38 89.02 49.874 9.22  47 77.45 44.811 11.63  40 75.98 45.681 11.95  43 81.19 49.091 10.85  44 81.42 39.442 13.08  38 81.87 60.055 8.63  44 73.03 50.541 10.13  45 87.66 37.388 14.03  45 66.45 44.754 11.12  47 79.15 47.273 10.60  54 83.12 51.855 10.33  49 81.42 49.156 8.95  51 69.63 40.836 10.95  51 77.91 46.672 10.00  48 91.63 46.774 10.25  49 73.37 50.388 10.08  57 73.37 39.407 12.63  54 79.38 46.080 11.17  52 76.32 45.441 9.63  50 70.87 54.625 8.92  51 67.25 45.118 11.08  54 91.63 39.203 12.88  51 73.71 45.790 10.47  57 59.08 50.545 9.93  49 76.32 48.673 9.40  48 61.24 47.920 11.50  52 82.78 47.467 10.50') |

## proc\_reg-001

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc reg data = fitness;  model Oxygen = RunTime;  run; |  |
| **R Code and Output** |  |
| proc\_reg(fitness,  model = Oxygen ~ RunTime) |  |
| **Comparison** | **Results** |
| Perform basic Reg on single variable. | Pass. |

## proc\_reg-002

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc reg data = fitness;  model Oxygen = RunTime / clb;  run; |  |
| **R Code and Output** |  |
| proc\_reg(fitness,  model = Oxygen ~ RunTime,  stats = clb) |  |
| **Comparison** | **Results** |
| Perform basic Reg on single variable with clb option. | Pass. |

## proc\_reg-003

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc reg data = fitness;  model Oxygen = RunTime / hcc;  run; |  |
| **R Code and Output** |  |
| proc\_reg(fitness,  model = Oxygen ~ RunTime) |  |
| **Comparison** | **Results** |
| Perform basic Reg on single variable with hcc option. | Pass. |

## proc\_reg-004

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc reg data = fitness;  model Oxygen = RunTime / hcc hccmethod=3;  run; |  |
| **R Code and Output** |  |
| proc\_reg(fitness,  model = Oxygen ~ RunTime) |  |
| **Comparison** | **Results** |
| Perform basic Reg on single variable with hcc type 3. | Pass. |

## proc\_reg-005

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc reg data = fitness;  model Oxygen = RunTime / p;  run; |  |
| **R Code and Output** |  |
| proc\_reg(fitness,  model = Oxygen ~ RunTime,  stats = p) |  |
| **Comparison** | **Results** |
| Perform basic Reg on single variable and p option | Pass. Decimal difference on dep variable OK. |

## proc\_reg-006

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| data fitness2;  set fitness;  if (Age < 50) then do;  AgeCat = "<50";  end;  else do;  AgeCat = ">=50";  end;  run;  proc sort data = fitness2 out = fitness2;  by AgeCat;  run;  proc reg data = fitness2;  model Oxygen = RunTime;  by AgeCat;  run; |  |
| **R Code and Output** |  |
| fitness2 <- fitness  fitness2$AgeCat <- ifelse(fitness$Age < 50,  "<50", ">=50")  proc\_reg(fitness2,  model = Oxygen ~ RunTime,  by = AgeCat) |  |
| **Comparison** | **Results** |
| Perform basic Reg on single variable with by groups. | Pass. |

## proc\_reg-007

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc reg data = fitness;  model Oxygen = RunTime Age;  run; |  |
| **R Code and Output** |  |
| proc\_reg(fitness,  model = Oxygen ~ RunTime + Age) |  |
| **Comparison** | **Results** |
| Perform basic Reg on single variable with two ind variables | Pass. |

## proc\_reg-008

|  |  |
| --- | --- |
| **SAS Code and Output** |  |
| proc reg data = fitness;  model Oxygen = RunTime Age Weight;  run; |  |
| **R Code and Output** |  |
| proc\_reg(fitness,  model = Oxygen ~ RunTime + Age + Weight ) |  |
| **Comparison** | **Results** |
| Perform basic Reg on single variable with three ind variables. | Pass. |