Common Lisp Analytical Statistics Package

CLASP*

David Fisher
Experimental Knowledge Systems Laboratory
Department of Computer Science
University of Massachusetts
Amherst, Massachusetts 01003

October 7, 1991

*This work was suppported by University Research Initiative grant, ONR N00014-86-K-0764. Original statistical analysis code by Paul Cohen and Scott Anderson. RTM relational table manager developed by Paul Silvey.

Abstract

This document provides the user manual for CLASP, an analytical tool developed to support research evaluation. CLASP is implemented in Common Lisp and runs on Texas Instruments Explorers® and MicroExplorers®. Through statistics we attempt to separate those events which are simply due to chance from those which are a result of, or are related to, other events in our world. Through careful analysis it is possible to understand a great deal about the state of the world. CLASP provides the tools necessary for these tasks. Along with describing the operation of CLASP this manual also details its programming interface, as an aid for building extensions.

Contents

List of Figures

Chapter 1

Introduction

When events occur in the world, people turn to statistics to analyze and describe them. This is especially true of the behavioral sciences, psychology and sociology. Through statistics we attempt to separate those events which are simply due to chance from those which are a result of, or are related to, other events in our world. Through careful analysis it is possible to understand a great deal about the state of the world. Within PHOENIX we have a new aspect of behavioral science, the study of the behavior of intelligent agents within a computer simulation, and so we turn to statistical analysis to describe the state of their world and the effects upon it. We would expect that, like our world, the PHOENIX world would lend itself to the same forms of analysis as those with which we study our world. We are also presented with the opportunity for the agents to use these techniques of analysis within the scope of the PHOENIX world, as methods for planning and evaluation. In order to address these two main goals it was necessary to develop a suite of statistical analysis tools which would be executable on a Texas Instruments Explorer®, implemented in Common Lisp. Building on the functions developed by Paul Cohen and Scott Anderson, the *Common Lisp Analytical Statistics Package*, CLASP, should grow into an integrated part of the PHOENIX desk-top.

Before embarking on this task and actually writing any code there were questions which needed to be answered. The first question that came to mind was 'What do we expect to learn from the study of the behavior of PHOENIX agents, and how can statistical analysis be used in support of this?' Much of what we wish to study revolves around the ability of the agents to implement a successful plan for the fighting of a fire, with data concerning the physical properties of the fire, the time periods necessary to put out the fire (both projected and actual), and other physical measurements, such as wind speed. Because of the nature of the simulation a number of factors are controllable, allowing very specific situations to be examined with a single factor being varied for any given trial. This provides an excellent environment for the analysis of effects of specific treatments, such as fighting a fire first or not, on the success of an individual plan. More complex effects, such as the combination of many factors, can also be examined in a similar fashion. Although the type of behavior we are studying is new, in the sense that it is programmed in a machine, it still lends itself to the analysis of effects in the same manner as the behavior of people or animals.

The second question, which is much more difficult, is 'How can the PHOENIX agents make use of statistical analysis techniques, and can their use be implemented in such a way as to support the activities of planning and plan evaluation?' The ability to evaluate the quality of

a plan, and the ability of the agent implementing that plan to modify it in the face of adverse or unexpected conditions, has an impact on the success of the overall plan being used. In the same way as researchers use modelling techniques for evaluation, the agents should model their own world in support of their planning activity.

In developing CLASP, a number of choices have been put off, or left to happenstance, especially concerning the interface, pending user evaluation. This is a different methodology from that to which I am accustomed, working on commercial applications, where almost all of the specifications for a program are 'carved in stone,' and there is little opportunity for experimentation. Implementing this package in Lisp has also been an interesting challenge, with the power and grace of the language just beginning to reveal themselves now that most of the basic coding has been completed. Overall work on this package has offered an opportunity for exploration, which seems appropriate for the environment. Every member of the lab has contributed to the development of this package, through advice, answering questions, and suggesting possible alternatives for solving difficult problems. Special thanks must be given to Paul Silvey for the use of his relational table manager, RTM, as the database engine for this program, and for his willing support during the integration of the two packages.

Chapter 2

Getting Started

2.1 Required Files

CLASP requires the following files, which are available in "*Christa: dfisher.clasp;*" and "*Christa: dfisher.clasp.v1;*".

- "*allocation-fns*"
- "*chi-2*"
- "*clasp-help.text*"
- "*clasp-system*"
- "*defs*"
- "*file-fns*"
- "*graph-fns*"
- "*help-fns*"
- "*loglinear*"
- "*reporting-fns*"
- "*selection-fns*"
- "*stat-fns*"
- "*transformation-fns*"
- "*win-fns*"

In addition RTM requires the following files, which are available in "*Christa: silvey.rtm;*" and "*Christa: silvey.rtm.v1.1;*". These are located and loaded by the `make-system` function, so the user does not have to load them manually.

- "*index-fns*"
- "*kernel-fns*"
- "*query-fns*"
- "*rtm-system*"
- "*startup-fns*"
- "*table-fns*"
- "*user-fns*"

2.2 Starting Up

To use CLASP the following forms need to be executed in the Lisp Listener, or in the users "*login-init*" file. The appropriate pathname should be used for the "*clasp-system*" file.

```
> (unless (find-package "CLASP")
        (make-package "CLASP"))
> (si:set-system-source-file :clasp "clasp-system")
> (si:set-system-source-file :rtm "Christa:  silvey.rtm;")
> (si:make-system :clasp :noconfirm)
> (w:modify-system-access-spec :clasp :assign-defaults)
```

On execution of `clasp-init` or use of the system–key stroke <SYSTEM>-Z an instance of the *Clasp Interaction* window will be created if necessary, and then selected. If the global variables for the default values of the CLASP and RTM data directories have not been set a menu is presented for entering these values. The primary command for using the program is the `get-user-action` command which presents a menu of possible actions, which can also be brought up by clicking left with the mouse. It is possible to drop down below the menu interface and call on the individual functions directly, but it is necessary to obtain a listing of the internal variable symbol names to facilitate use of the functions.

The tradeoff here is one of ease of use versus power. Combining the functions in small programs targeted at specific pieces of data allow more specialized information to be gathered, however it is more difficult to use the program without the menus. Whereas with the menus the choices are somewhat limited, they are very easy to use and supply a reasonable amount of information and functionality. To use the functions without the menu interface consult the argument lists for the specific functions you want to use. The numeric functions, i.e. `mean`, `variance` etc., expect the argument to be in the form of a sequence, in this case an unquoted internal variable symbol i.e. `var-n`, where n is the specific number of the variable you are interested in. The reporting functions, such as `statistical-summary`, expect a quoted symbol, i.e. `'var-n`. The multi-variate functions, `one-way`, `two-way`, `linear-regression`, and `plot-y-on-x`, expect the variables to be given in the form of quoted variable symbols. Predominately the argument names for the functions are descriptive of the argument's type.

Chapter 3

Using CLASP

3.1 Data Files Format

The CLASP format for data files uses two components. The first section of the file consists of some number of quoted strings, the first of which is the header string, or identifier, for the file, and the rest the names for the variables. These names can be more than one word. Immediately following the strings should be some number of lists, with each list having one value for each variable named above, in the same order as the strings. The values of each of the individual elements of the list can be of any type.

3.2 User Interface

3.2.1 User Menus

Main User Action Menu

The interface functions of the package are variations on some type of 'get' or 'select' action. Each 'get' involves the presentation of a menu allowing the user to make a selection of some type. The primary function the user will want to invoke in the *Clasp Interaction* window is `get-user-action`, which can be run by clicking left with the mouse, which invokes the main menu for using the package. Each of the options of the main menu uses defaults which are expected to be appropriate in most situations, however all of the functions can be run at top level if different parameters are desired.

The options of the *main user action* menu are:

- Load a data-set from a file
  Calls `load-data-set` with no arguments. This defaults to a menu selection of files with the default data extension from the default data directory. Those values are set at initialization or with the function `set-global-variable-values`. `Load-data-set` can also be invoked by clicking right with the mouse.
- Load a data-set from multiple files
  Calls `load-data-from-files` with no arguments. All files are merged into a single data set. This function will return an error if the variable name strings (columns) in the files are not identical. This can be done by double clicking left with the mouse.

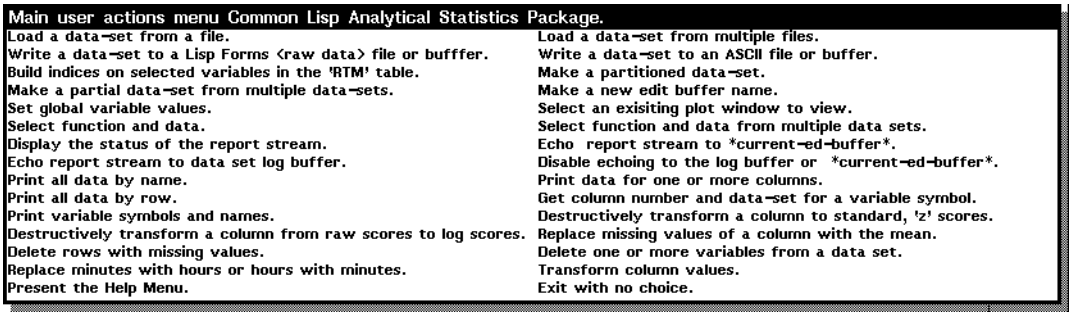

Figure 3.1: Main User Action Menu

- Write a data-set to a Lisp Forms file or buffer.
  Calls `write-lisp-forms-file-or-buffer` with no arguments. The default is the menu selection of a data-set which is written to either the buffer name which is the value of `*current-ed-buffer*` or the filename of the data set, based on a menu choice. The output format is the same as the raw data file input format.
- Write a data-set to an ASCII file or buffer.
  Calls `write-ascii-file-or-buffer` with no arguments. As above the default is for a menu choice of a data set written to either the filename of the data set or the buffer named by `*current-ed-buffer*`. The default delimiter for the file is the space character.
- Build indices on selected variables in the RTM table.
  Builds indices for the selected variables, which speeds up the response times for the functions `statistical-summary`, `one-way`, and `two-way`, which use the table to select values for the calculations. With a large data set this is a time consuming operation. With very large data sets building too many indices can cause memory to overflow.
- Make a partitioned data-set.
  Calls `make-partitioned-data-set` with no arguments. This generates an internal symbol for the data set name and presents a selection menu to choose which variable to partition on. After selecting a variable a second menu selects a property, such as 'less than', 'greater than' or 'equal' on which to base the partition. A third menu is presented for entry of the key value on which to apply the property test selected in the previous menu. This creates a new data-set with all variable values from the rows in which the key variable satisfied the test.
- Make a partial data-set from multiple data-sets.
  Calls `make-partial-multiple-data-set` with no arguments. As above this results in a menu for selecting variables, this time from more than one data-set, allowing the combination of observations from different sets. This creates a new data-set with only the selected variables.
- Make a new edit buffer name.
  Calls `make-new-buffer-symbol` which generates a new buffer name of the form 'DATA-BUFFER-n' where n is the value of `*buffer-counter*`.

- Set global variable values.
  Calls `set-global-variable-values` which presents the menu for entering the values of the default data paths and extension. The variables which are set are:
  - `*clasp-data-directory*`
  - `*rtm-data-directory*`
  - `*clasp-data-extension*`
- Select an existing plot window to view.
  Presents a menu of the existing plot windows for selection.
- Select function and data.
  Calls `get-data-set-fun-data` which presents the *data set selection* menu, the *statistical functions* menu, and the *variable selection* menu. Multiple variables can be selected at one time. This can be called by clicking middle with the mouse.
- Select function and data from multiple data-sets.
  Calls `get-multiple-data-set-fun-data` which behaves as above but presents a *data set selection* menu between each variable selection. One variable value at a time should be selected.
- Display the status of the report stream.
  Calls `check-stream-status` which displays the current streams to which reports are being echoed.
- Echo report stream to `*current-ed-buffer*`.
  Sets the value of `*reports-to-ed-buffer-p*` to true. All output to the *Clasp Interaction* window is echoed to the buffer named by `*current-ed-buffer*`.
- Echo report stream to data set log buffer.
  Sets the value of `*reports-to-log-p*` to true. All output to the *Clasp Interaction* window is echoed to the log buffer of the `data-set` from which they were generated.
- Disable echoing to the log buffer or `*current-ed-buffer*`.
  Presents a selection menu for disabling echoing of the report stream.
- Print data for one or more columns
  Calls `print-column-data` for each selected variable, selected from the *variable selection* menu.
- Print all data by name.
  Calls `print-all-data` which displays all of the data for the selected data set in columnar format, two columns at time.
- Print all data by row.
  Prompts for a choice of selecting a single data-set, if not it applies `print-row-data` to each row of each data-set. It is recommended to always select a single data-set.
- Get column number and data-set title for a variable symbol.
  Calls `column-number` with a menu choice of data-sets and variables. Predominantly an internal function. Use this function to get information needed when using CLASP below the menu interface.
- Print variable symbols and names.
  Prompts for choice of data-set or goes through all available data-sets if none is selected.

Calls `print-variable-symbol-name` to display the internal symbols and the associated name strings for each variable of the data-set(s).

- Destructively transform a column to standard, 'z' scores.
  Calls `ntransform-raw-score-to-z-score` with menu selection of data-set and variable. This converts raw scores to deviation scores preserving the distribution of the original data.
- Destructively transform a column to log scores.
  Calls `ntransform-raw-score-to-log-score` with menu selection of data-set and variable. This converts raw scores to the natural log of the scores.
- Destructively replace missing values with the mean value of a selected column.
  Calls `nreplace-with-mean` which provides menu selection of the variables to be transformed. For most cases this substitution is adequate for dealing with missing values. When the number of observations are small, or when paired data values are important the function `delete-rows-with-missing-values` should be used.
- Delete rows with missing values.
  Presents a menu for selecting variables and deleting all rows from the data set where there is a missing, i.e., `Nil` value. This is a destructive operation.
- Replace hours with minutes or minutes with hours.
  Presents a menu for selecting one of two transformation functions.
- Transform column values.
  Presents a menu for entering new values for a variable. Displays the old values for reference.
- Display the Help Menu.
  Presents the *Help Topics* menu which displays the selected topics in the *Help* window.
- Exit with no choice.
  Exits the menu with no selection.

Support Menus

The current data set is selected with the *Data Set Selection* menu, which presents the available data sets by filename. This is the first menu to be displayed when *Select function and data* is chosen from the main menu, which can also be called by clicking middle with the mouse. The data sets are selected by their filename, which is the same as the filename of the file which was loaded in the case of single data sets from single files, and the filename which was entered by the user in the case of partitioned data sets, or data sets composed of multiple files. The documentation line for this menu indicates which data set was most recently used.

The *Statistical Functions* menu presents the functions available for analysis of the data, including the graphic plotting of the data through histograms and regressions. The documentation line for this menu provides a brief description of the functions and help is available online from the *Help Topics* menu.

The default values for the parameters to each of these functions allow for the selection of the variables from the *variable selection* menu, which presents the available variables in a data set. The names of the variables on the menu are the strings from the original data file. The title bar of the variable selection menu indicates the type of variable which is expected

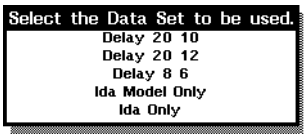

Figure 3.2: Data Set Selection Menu

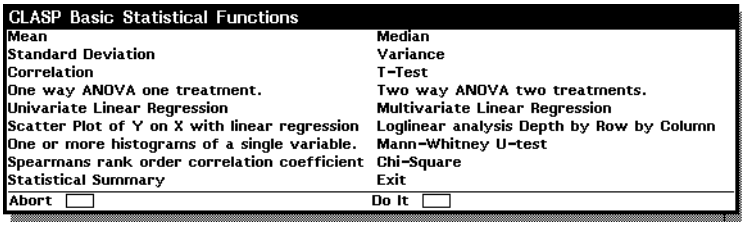

Figure 3.3: Statistical Functions Menu

to be selected, such as independent, dependent, or partition variable, as well as the data set to which the variables belong.

Global variable values are set using the *Set Variable Values* menu, which displays the values for `*clasp-data-directory*`, `*rtm-data-directory*` and `*clasp-data-extension*`. It is necessary to set the values of these variables in order for the *Directory* menu to work properly. These values can be preset in the user's "*login-init*" file. If the user presets these values they must be strings and should be prefaced with their package qualifier, `clasp`, or `rtm` for `*rtm-data-directory*`. Values entered in the menu do not require quotes, however the pathnames require the trailing semicolon, and the extension requires the leading period.

Data files are selected from the *Directory* menu, which displays the directory listing of the directory, `*clasp-data-directory*` for files with the extension `*clasp-data-extension*`. One or more files can be selected at one time.

The *Help Topics* menu presents a list of possible help topics which can be displayed in the *Help* window. The help is stored in the file "*clasp-help.text*" in the same directory as the source files. The online help system is under development with basic help available for

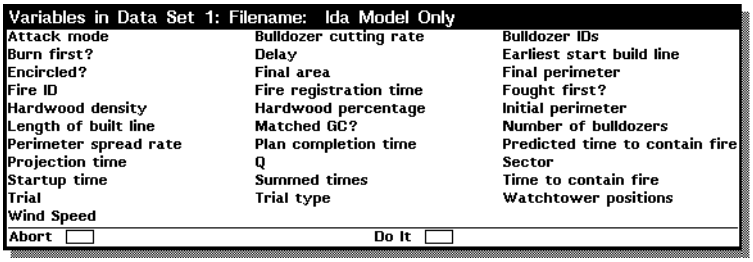

Figure 3.4: Variable Selection Menu

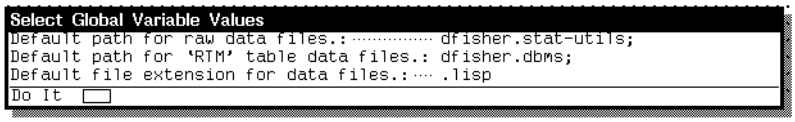

Figure 3.5: Set Global Variables Menu

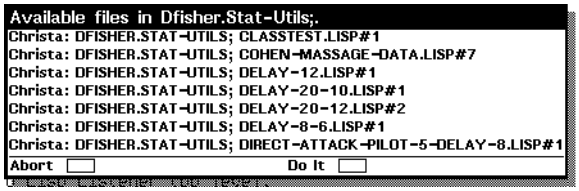

Figure 3.6: Data File Directory Menu

general topics. One or more topics can be selected from the help menu. If more than one topic is selected the user must deselect the window, by pressing <END> or clicking on a different window with the mouse, in order to see the next selected topic. To exit help simply press <END> or select another window with the mouse. While in the help window the *Help Topics* menu can be brought up by clicking right with the mouse.

These menus make up the basis for the operation of CLASP with most of the functionality of the system encapsulated in the menu selection actions. The limitations imposed by using menus can be overcome by using the individual functions directly in the *Clasp Interaction* window, as has been mentioned above. When using the functions without the menu interface the value of `*current-data-set*` should be updated manually, using the form: `(setf *current-data-set* data-set-n)`, where n is the specific number of the data set being used. This value is located in the title field of the `data-set` structure. Individual variables are referenced by their internal symbols, which are of the form: `var-n`. A listing of these can be made using the function `print-variable-symbol-name`.

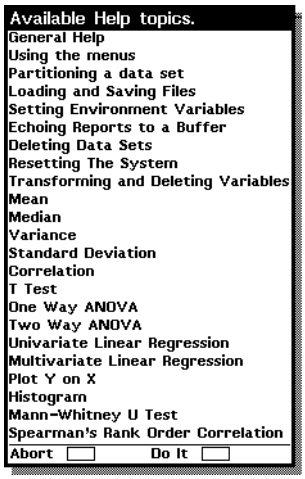

Figure 3.7: Help Menu

Chapter 4

Statistical Analysis

The analytical functions available in CLASP cover the range of basic descriptive and inferential statistics. The argument list and formulas for each are described in **Functions and Variables**. Most analyses will focus on the variation of a variable due to the influence of some *effect* or *treatment*, such as the property of which fire is fought first, or the direction of the wind. The functions supporting this type of analysis fall into two categories, parametric and non-parametric, each of which is most applicable for certain specific types of problems. Non-parametric statistics are used predominately when the distribution of the sample data is unknown, or when the assumption that the data is normally distributed does not hold. The values generated by the parametric and non-parametric tests will be identical when the distribution of the data is normal or near normal, and there is no strong argument either for or against the use of one rather than the other. In general the parametric statistics are prefered because they are more widely known. This section will provide an overview of the use of each of these statistics and the general model with which they are associated. There are many excellent statistics texts available (a good general text is *Statistical Reasoning in Psychology and Education* by Edward W. Minium, 1978, John Wiley and Sons) and one of these should be refered to for a more in-depth treatment for any of the following items.

4.1 ANOVA

Analysis of variance is used to identify the amount of variation in a sample due to the effect of some treatment. There are two forms of *ANOVA* implemented in the package, *one way* and *two way*. *One way ANOVA* is used to examine the effect of a single treatment on one or more samples, such as comparing the *time to contain fire* when *Burn first?* is `true` versus `nil`. *Two way ANOVA* is used when considering the effects of multiple treatments and their interactions. Each is calculated in a similar fashion. Values used in calculating the *ANOVA* table are based on the sum of squares of the deviation scores, SS, and the degrees of freedom, df, associated with that sum of squares. The general relationship is in the form

$$s^2 = \frac{SS}{df}$$

The types of variation involved for *one way ANOVA* are those of:

$$
\begin{aligned}
SS_W &= \sum(X_i - \overline{X}_i)^2 + \sum(X_j - \overline{X}_j)^2 + \cdots \\
SS_A &= \sum_i^k(\overline{X}_i - \overline{\overline{X}})^2 \\
SS_T &= \sum(X - \overline{\overline{X}})^2 \\
df_W &= \sum(n_i - 1) \\
df_A &= k - 1 \\
df_T &= \sum n_i - 1 \\
s_W^2 &= \frac{SS_W}{df_W} \\
s_A^2 &= \frac{SS_A}{df_A}
\end{aligned}
$$

Figure 4.1: One Way ANOVA Calculations

1. Variability of the values about the grand mean (the mean of all scores). Given by the deviation: $(X - \overline{\overline{X}})$, where $\overline{\overline{X}}$ is the grand mean. This is the known as the *total* variance, symbolized as s_T^2. This value is useful only in the calculating of the following items.

2. Variability of scores about their subgroup sample means. Given by the deviation: $(X - \overline{X})$,where $\overline{X}$ is the mean of the subgroup which contains X. This is known as the *within groups* variance, denoted by s_W^2.

3. Variability of subgroup sample means about the grand mean of all scores. Given by the deviation: $(\overline{X} - \overline{\overline{X}})$. This is the *among groups* variance, denoted by, s_A^2.

The *within groups* variance indicates the amount of variation inherent in the subgroup, all of the members of the group receive the same treatment so none of the variation can be due to the treatment. The *among groups* variance contains two components, the variance inherent in the population and the variance due to the treatment. By examining these components the effect of a treatment on a population can be determined.

Calculations for these values are described by the formulae in Figure 4.1. where X_i are the scores in subgroup i, $\overline{X}_i$ is the mean of subgroup i, n_i is the number of observations in the ith subgroup and k is the number of subgroups.
Using these values, the F ratio is calculated. The F ratio indicates the probability that the variation in the samples could be accounted for by chance. Calculation of the F ratio uses the following formula:

$$F = \frac{s_A^2}{s_W^2}$$

For *two way ANOVA* there is an additional interaction between different treatments. These are separated into *Row* and *Column* treatments, for example including the *wind speed* in the example above would partition the group across two different treatments, with each cell of the table being a combination of the two. This adds an additional interaction effect, where the property of being burnt first may combine with certain wind speeds to produce variation in the time to contain the fire. This *Row* by *Column* effect is calculated with the formulae in Figure 4.2. Where the four variance estimates correspond to:

$$
\begin{aligned}
SS_C &= \sum(\overline{X}_{C_i} - \overline{\overline{X}})^2 \\
SS_R &= \sum(\overline{X}_{R_i} - \overline{\overline{X}})^2 \\
SS_{WC} &= \sum(X - \overline{X}_{cell})^2 \\
SS_{R\times C} &= SS_T - SS_C - SS_R - SS_{WC} \\
df_{WC} &= \sum^{all\ cells}(n_{WC} - 1) \\
df_C &= (C-1) \\
df_R &= (R-1) \\
df_{R\times C} &= (C-1)(R-1) \\
df_T &= (R)(C)(n_{WC}) - 1 \\
s^2_C &= \frac{SS_C}{df_C} \\
s^2_R &= \frac{SS_R}{df_R} \\
s^2_{WC} &= \frac{SS_{WC}}{df_{WC}} \\
s^2_{R\times C} &= \frac{SS_{R\times C}}{df_{R\times C}}
\end{aligned}
$$

Figure 4.2: Two Way ANOVA Calculations

s^2_{WC} *within cells estimate*, derived from the individual cell variation. This measures the inherent variation in an subgroup free from the effect of any treatment.

s^2_C *column estimate*, derived from the differences from the column means. If there is an effect from the column treatments this value will tend to be larger than s^2_{WC}.

s^2_R *row estimate*, as above but for row treatments.

$s^2_{R\times C}$ *interaction estimate*, derived from the discrepancy between the means of several cells. This value will tend to be larger than s^2_{WC} if there is an interaction effect between the *Row* and *Column* treatments.

There are three F ratios calculated, column, row, and row by column in the form $F = \frac{s^2_C}{s^2_{WC}}$, $F = \frac{s^2_R}{s^2_{WC}}$, and $F = \frac{s^2_{R\times C}}{s^2_{WC}}$.

A significant value of F indicates that the hypothesis that the means of the subgroups are equal, that is that there is no variance due to the treatment, should be rejected. *Two way ANOVA* requires that each of the cells have the same number of observations in order to calculate a meaningful statistic, unequal group sizes make the statistic unstable.

4.2 Chi Square χ^2

Chi square provides a method for comparing the observed frequencies of a condition or event against the expected frequencies for the population. A significant value of *Chi square* indicates that the null hypothesis should be rejected and that an interaction effect may be present. This opens the door to further analysis to identify the relationship, or dependency, between the variables.

4.2.1 χ^2 2×2

χ^2 is implemented in CLASP for a 2×2 contingency table in the form

a	b
c	d

where each cell is the observed frequency for the combination of the two properties being analyzed. The null hypothesis for the test is that the variables being tested are independent in the population, that is that the expected frequencies for each of the conditions will be the same. The function optionally uses the *Yate's correction*, which adjusts for the discontinuity of the χ^2 distribution.

4.2.2 χ^2 $m \times n$*

The statistic is also implemented in the general form for an $m \times n$ contingency table. The *Yate's correction* is applied when $df = 1$. The raw input data is used to calculate the observed frequencies. The $m \times n$ contingency table is of the form

$$\begin{matrix} f_{o_{11}} & f_{o_{12}} & \cdots & f_{o_{1n}} \\ f_{o_{21}} & f_{o_{22}} & \cdots & f_{o_{2n}} \\ \vdots & & & \\ f_{o_{m1}} & f_{o_{m2}} & \cdots & f_{o_{mn}} \end{matrix}$$

Based on the above observed frequencies, the expected frequencies for the population are computed. The null hypothesis for the test is that the variables being tested are independent in the population. The expected frequency in cell (i, j) is defined as the product of the sum of the i'th row and the sum of the j'th column, divided by the sample size:

$$f_{e_{ij}} = \frac{\sum_{k=1}^{n} f_{o_{ik}} \sum_{k=1}^{m} f_{o_{kj}}}{\sum_{p=1}^{m} \sum_{q=1}^{n} f_{o_{pq}}}$$

χ^2 is then used top compare the observed frequencies of a condition or event against the expected frequencies of the population using the formula:

$$\chi^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{\left(f_{o_{ij}} - f_{e_{ij}}\right)^2}{f_{e_{ij}}}$$

Yates correction is applied when $df = (m-1)(n-1) = 1$ as follows:

$$\chi^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{\left(\left|f_{o_{ij}} - f_{e_{ij}}\right| - 0.5\right)^2}{f_{e_{ij}}}$$

*This statistic was implemented by Sameen Fatima of EKSL, who also wrote this section of the manual.

4.3 Correlation r_p

This statistic, *Pearson's correlation coefficient*, describes how well the value of the dependent, Y, variable can be predicted from the score of the independent, X, variable. This relationship does not imply causality, it simply indicates that a high value of Y is likely to occur when there is a high value of X. The two variables could be entirely unrelated. Consider the case of comparing the orbital radii of the planets of our solar system to the atomic mass of the elements of the periodic table. Each of the sets of observations are strictly increasing (if we start with Mercury and Hydrogen) and will exhibit a high positive correlation even though we know that the two items are totally unrelated. This statistic is considered by some to be overused and abused, but generally it is a good indicator of an effect or relationship between two variables.

4.4 Linear Regression

This function yields a great deal of information concerning the relationship between two or more variables. From the regression equation the value of the dependent variable, Y, can be predicted from the value of the independent variable, X, with the F ratio for the regression indicating the probability of the apparent relationship occurring due to chance. Combining a plot of the regression line with a scatter plot of the individual data points provides a picture of the relationship between the variables and also gives a good indication of the correlation between the variables. It is important to remember that there may be relationships between the variables which are non-linear, such as a logarithmic or quadratic relationship.

Multi-variate linear regression performs the same calculations with multiple independent variables, calculating the equation $Y = a + b_1X_1 + b_2X_2 + \cdots + b_nX_n + e$ where b_i is the *coefficient of regression* for the corresponding X, a is the intercept of the regression line, and e is the error, or residual. Each b has a corresponding t statistic indicating the probability of whether or not the relationship could occur by chance. A significant b indicates that a portion of the *variance* of the dependent variable is due to the influence of the associated independent variable. The amount of the influence is measured as a percentage of the total *variance* of the dependent variable, which is calculated as the square of the *correlation* between Y and X_i. The percentage of the *variance* due to all of the independent variables is given by R^2, with the F statistic for the regression calculated from this value. The report for this procedure includes the correlation coefficients between the independent variables.

4.5 Log-linear Analysis[†]

Log-linear models form the basis for the analysis of contingency tables. For a 3-way table, the log-linear model is

$$\ln \hat{f}_{ijk} = \mu + \alpha_i + \beta_j + \gamma_k + \alpha\beta_{ij} + \alpha\gamma_{ik} + \beta\gamma_{jk} + \alpha\beta\gamma_{ijk}$$

[†]This statistic was implemented by Dorothy Mammen of EKSL, who also wrote this section of the manual. This analysis of three-way contingency tables is based on the discussion of the analysis of three–way and multiway tables given in Sokal and Rohlf, Biometry, 2nd edition, 1981, Section 17.5, pp. 747–765. It is roughly summarized here, but the reader is referred to that source for a more thorough discussion, as well as a complete example analysis.

where $\ln \hat{f}_{ijk}$ is the expected frequency in row i, column j, depth k in a 3-way contingency table; μ is the mean of the logarithms of the expected frequencies; α_i, β_j, and γ_k are the effects of categories i, j, and k of factors **A**, **B** and Γ respectively; $\alpha\beta_{ij}$, $\alpha\gamma_{ik}$, and $\beta\gamma_{jk}$ are the two-way interaction terms; and $\alpha\beta\gamma_{ijk}$ is the three-way interaction effect. Multiple analyses are done here. Each analysis consists of comparing the fit of the log-linear model calculated with and without some term (or terms), in order to determine whether the term contributes significantly. The G-statistic for goodness of fit is used for testing the significance of the term being left out. *Note that the appropriateness and interpretation of some of the statistics reported in this analysis depend on others; information to this effect is contained in this documentation (and in Sokal and Rohlf) but does not appear with the output statistics themselves.*

The first analysis is a test for the significance of the three-way interaction term, $\alpha\beta\gamma_{ijk}$, that is, a test of the log-linear model above with that term omitted versus with that term included. An iterative proportional fitting algorithm must be used to compute the $\hat{f}$ for this model (see Sokal and Rohlf). If the three-factor interaction term is significant, this means that the degree of association between any pair of variables depends upon the different levels of a third variable. *If this is the case, it does not make sense to fit any of the simpler models below.* Rather, one could proceed with separate two-way tests of independence within each level of one of the factors.

If the three-factor interaction is not significant, the next step is to test for the significance of two-way interactions. The second set of analyses consists of testing each of the three possible models with one two-factor term absent. Each of these analyses tests for the independence of two factors from each other given the level of the third factor.

The third set of analyses consists testing each of the three possible models with two two-factor terms absent. Each of these analyses tests for the complete independence of one factor versus the other two factors.

The final analysis tests for the complete independence of all three factors. It is only meaningful if there were no two-way interactions in the second set of analyses above.

4.6 Mann-Whitney U-test

This is a non-parametric statistic for comparing the means of two independent samples. Interpretation of z is the same as for the t test, with the null hypothesis being that the two means are the same. When the samples are of unequal size the smaller sample is made the X variable and the larger the Y variable. A significant z indicates that the two distributions are not the same, with a negative value indicating the mean of X is less than the mean of Y and a positive value the reverse.

4.7 Median

Another measure of central tendency, which indicates the value at the center of the distribution. Half of the scores will be above and half below, however it is also affected by skewed distributions. The larger the difference between the *median* and the *mean* the more likely that the distribution is not normal.

4.8 Mean $\overline{X}$

This is the average of a group of numbers, which describes the center point of the distribution of the sample. This value is a good measure of the central tendency of a population, however it is affected by a skewed distribution, or when there are extreme outliers.

4.9 Spearman's Rank Order Correlation Coefficient r_s

This is a non-parametric statistic which indicates the correlation between two samples. This is used when the distributions of the samples may be inappropriate for comparison with *Pearson's correlation coefficient*, or when sample sizes are small. This statistic is based on the relative ranks of the observations, so the ordering of the data is important. This statistic should not be used on unordered data.

4.10 Standard Deviation s

This describes the range over which the values of the sample are distributed. It is the square root of the *variance*. With a normal distribution 95% of the values will fall within the range of $\overline{X} \pm 1.96s$.

4.11 T Test

The *t test* is used when comparing the means of two independent distributions. In general the null hypothesis is that the two means are equal with a significant value of t indicating that the two means are different and that the null hypothesis should be rejected. An insignificant t indicates that the null hypothesis should not be rejected. The *t test* has been implemented using both the *pooled variance* model, for testing independant samples, and the *matched pairs* model, for testing dependant samples.

4.12 Variance s^2

This describes the amount of variation which is present in the sample in terms of deviations from the *mean*. With this value a picture of the distribution of the population from which the sample was drawn can be made. The amount of *variance* in a sample due to a specific treatment or effect is what is examined with *ANOVA*.

Chapter 5

Functions and Variables

5.1 Statistical Functions

`chi-2-2` *&optional (a nil) (b nil) (c nil) (d nil) (yates t)* [*Function*]

Runs a χ^2 test for association on a simple 2 x 2 table. By convention, the cells are labelled

a	b
c	d

If *yates* is `nil`, the correction for continuity is not done. If the cell values are not provided they are prompted for in the interaction window.

`chi**2` *&optional (row-treatments nil) (column-treatments nil) (report-stream *clasp-report-stream*)* [*Function*]

Performs $m \times n$ χ^2 analysis. Applies the Yate's correction for continuity when $df = 1$.

`chi-2-pdf` *chi**2 dof* [*Function*]

Returns value of p indicating the probability the value of *chi**2*, with degrees of freedom *df*, could occur by chance.

`cross-product` *number-list-1 number-list-2* [*Function*]

Takes two sequences of numbers and returns a sequence of cross products. This is an element-by-element multiplication of the two sequences.
Formula:

$$(XY)_i = X_i Y_i$$

`difference-list` *number-list* [*Function*]

Takes a sequence of numbers and returns a sequence of differences from the mean.
Formula:

$$x_i = X_i - \overline{X}$$

`f-pdf` *f m n* [*Function*]

Returns the probability that the value of f could occur by chance.

`gamma` *a* [*Function*]

Returns $\Gamma(a)$ where $\Gamma(n+1) = n!$ and $\Gamma(1/2) = \sqrt{\pi}$. This function is defined only over multiples of 1/2.

`linear-regression` *&optional (x-var nil) (y-var nil) (print? t)* [*Function*]

Performs linear regression of Y on X, calculating the intercept and regression coefficient. Calculates the F statistic, intercept and the correlation coefficient for Y on X. If a variable is not specified, a menu is presented for selection.

`log-linear` *&optional (depth-treatments nil)* [*Function*]
(row-treatments nil)
(column-treatments nil)
*(report-stream *clasp-report-stream*)*

Runs a log-linear analysis for associations on a 3-way table. Table is a 3-level nested sequence, depth-row-column [A-B-C] from outside to inside. Analysis based on description in Sokal and Rohlf, Biometry, 2nd ed.,1981, pp 747-765

`mean` *number-list* [*Function*]

Takes a sequence of numbers, returns a single float arithmetic mean. Removes nil values from sequence.
Formula:

$$\overline{X} = \frac{\sum_{i=1}^{N} X_i}{N}$$

`median` *number-list* [*Function*]

Selects the median value for a variable.

`multi-linear-regression` *x-var-list y-var-symbol* [*Function*]
&key (plot? nil) (print? t)

Performs linear regression of Y on multiple X's, calculating the intercept and regression coefficient. Calculates the F statistic, intercept and the correlation coefficient for Y on X's.

`one-way` *&optional (independent-var nil)* [*Function*]
(dependent-var nil)
*(report-stream *clasp-report-stream*)*

Takes some number of sequences and returns the one way ANOVA table, Sum-of-Squares within, Sum-of-Squares between, Sum-of-Squares total, Degrees of freedom, Mean square error, and the F statistic. If no variables are given, a menu is presented for selection.

`r-score` *number-list-1 number-list-2* [*Function*]

Takes two sequences and returns the correlation coefficient.
Formula:

$$r_p = \frac{\sum_{i=1}^{N}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{N}(X_i - \overline{X})^2 \sum_{i=1}^{N}(Y_i - \overline{Y})^2}}$$

`scalar-matrix-multiply` *scalar matrix* [*Function*]

Multiplies a matrix **M** by a scalar value s in the form:

$$M_{[i,j]} = sM_{[i,j]}$$

`spearmans-rho` *x-var y-var* [*Function*]

Calculates the Spearman rank order correlation coefficient. Useful for comparing small samples where the distributions of the data may be inappropriate for analysis using the Pearson correlation coefficient. Samples must have the same number of observations in order to calculate a meaningful statistic.

`square` *number* [*Function*]

Returns the square of a number.

`standard-error` *number-list* [*Function*]

Takes a sequence of numbers and returns the standard error of the mean. This value approximates the standard deviation of the distribution of the sample means from the population mean.
Formula:

$$s_{\overline{X}} = \frac{s}{\sqrt{N}}$$

`statistical-summary` *var-symbol &key (data-set *current-data-set*) (file *clasp-report-stream*)* [*Function*]

Returns summary statistics for a sequence of numbers; Mean, Variance, Standard Deviation, Minimum, Maximum, Range. Output can be optionally sent to a file or edit buffer.

`std-deviation` *number-list* [*Function*]

Takes a sequence of numbers, returns a single float standard deviation.
Formula:

$$s = \sqrt{\frac{\sum_{i=1}^{N}(X_i - \overline{X})^2}{(N-1)}}$$

`sum-list` *number-list* [*Function*]

Takes a sequence of numbers and returns their sum.
Formula:

$$\sum_{i=1}^{N} X_i$$

`sum-of-squares` *number-list &optional (mean (mean number-list))* [*Function*]

Takes a sequence of numbers and returns the sum of squared deviations.
Formula:

$$\sum x^2 = \sum_{i=1}^{N} (X_i - \overline{X})^2$$

`t-pdf` *tt dof* [*Function*]

Returns the probability that the value of *t*, *tt*, could arise by chance.

`t-statistic` *sample-1 sample-2* [*Function*]

Returns the t-statistic for two independent samples, which should both be sequences of numbers. Does not say whether the statistic is significant—see the function `t-pdf` Returns `nil` if there is no variance in the two samples.

`t-statistic-matched` *sample-1 sample-2* [*Function*]

Returns the t-statistic for two dependent samples, which should both be sequences of numbers. Does not say whether the statistic is significant—see the function `t-pdf`. Returns `nil` if there is no variance in the two samples.

`t-test` *sample-1 sample-2 &optional (print? t)* [*Function*]
*(stream-name *clasp-report-stream*)*

Performs a t-test on two independent samples using pooled variance and returns the results. If *print?* is true, prints the analysis to *stream*, which defaults to the `*clasp-report-stream*`.

`t-test-matched` *sample-1 sample-2 &optional (print? t)* [*Function*]
*(stream-name *clasp-report-stream*)*

Performs a t-test on two dependent samples using matched pairs and returns the results. If *print?* is true, prints the analysis to *stream*, which defaults to the `*clasp-report-stream*`.

`two-way` *&optional (row-treatments nil)* [*Function*]
(column-treatments nil)
(variable-list nil)
*(plot-means? t) (report-stream *clasp-report-stream*)*

Takes some number of variable sequences and returns the two way ANOVA table for two seperate multi-level treatments. If the variables are not specified menus are presented for their selection. Reports Sum-of-Squares for each treatment and the interaction of treatments, Degrees of freedom, Mean square error, and the F statistic. Plots the cell means as the default. This method requires cell sizes to be the same size in order for the statistic to be meaningful. Unbalanced ANOVA is possible with regression techniques.

`u-test` *var1 var2* [*Function*]

Mann-Whitney U-test, for comparing the means of two independant samples. Compensates for small sample size and differences in the shape of the two distributions. Results similar to a t-test. Useful when a *t* test or *one way ANOVA* would be invalid due to size or distribution. If sample sizes are unequal the smaller sample is made the X variable and the larger the Y variable. A significant negative z indicates $\bar{X} < \bar{Y}$

and a positive z $\bar{X} > \bar{Y}$. $\sum R_X$ is the sum of the ranks of the X variable.
Formula:

$$z = \frac{\sum_{i=1}^{N} R_{X_i} - 0.5[n_X(n_X + n_Y + 1)]}{\sqrt{\frac{n_X n_Y (n_X + n_Y + 1)}{12}}}$$

`variance` *number-list* [*Function*]

Takes a sequence of numbers, returns the variance from the mean. Removes nil values from the sequence.
Formula:

$$s^2 = \frac{\sum_{i=1}^{N}(X_i - \overline{X})^2}{(N-1)}$$

5.2 Graph Functions

`histogram` *attributes data-set &optional (number-of-subdivisions 100)* [*Function*]

Plots frequencies of a single variable on the `*clasp-plot-window*`. If called with a list of variables creates multiple plot windows, sized by the mouse. Also plots lines at the mean and 1 and 1.96 standard deviations above and below the mean.

`plot-cell-means` *cell-means-list num-cols title* [*Function*]

Plots the cell means from a two way anova.

`plot-regression-line` *x-list int b-list &optional (x-msg nil) (y-msg nil)* [*Function*]

Plots a single line from a multi-variate regression at the intercept and the max-value for the sum of the X variables.

`plot-y-hat` *x-var-list y-var-symbol* [*Function*]

Plots Y values versus the predicted values $\hat{Y}$.

`plot-y-on-x` *x-var y-var &optional (graph-window *clasp-plot-window*)* [*Function*]

Draws a simple scatter plot with the line of best fit. Will not work on data that contains missing values, i.e. `nil`.

5.3 Initialization and Internal Functions

`bind-data-points` *&optional (data-set-name *current-data-set*)* [*Function*]

Binds internal symbols to values in the given data set. This is used at load time by `load-data-set`.

`clasp-init` *&optional (datafile-list nil)* [*Function*]

Initializes system and sets default data directory values. Creates the main window and plotting window. When given a datafile-list a single data-set is created from all of the files in the list.

`clasp-mouse-window-function` *function* [*Function*]

Executes the argument function followed by a forced keyboard input to return to the prompt of the clasp window.

`clasp-reset` [*Function*]

Use to release data-sets from memory and reset global variables.

`drop-data-set` *&optional (data-set nil) (gc? t)* [*Function*]

Removes a data-set from memory. Deletes resources associated with it. Prompts for deleting the log buffer of the data set. If no data set is specified a menu is presented to choose one.

`drop-plot-windows` [*Function*]

Clears the `*clasp-plot-window-list*` and deletes the associated resources.

`rtm-init` [*Function*]

Defines the domains `float-with-nil`, `symbol-or-num`, and `list` for use by RTM.

`set-more-processing` *status &optional (window *clasp-window*)* [*Function*]

Set the status of more processing for the *Clasp Interaction* window, `True` to enable, `Nil` to disable. When the optional *window* is provided more processing status for that window is set.

5.4 Allocation Functions

`make-clasp-window` [*Function*]

Initializes an instance of the primary window.

`make-data-set-attribute-list` [*Function*]

Generates the attribute list for construction of the data set menu.

`make-fun-menu` [*Function*]

Creates the Statistical functions menu.

`make-header-list` *data-list* [*Function*]

Takes a list of lists of variable symbols and data-set names and returns a list of the concatenated data-set names and variable name strings.

`make-help-window` [*Function*]

Instantiates an instance of the help-window flavor.

`make-indices` *var-symbols-list rtm-table* [*Function*]

Builds indices on the selected variables in the RTM table.

`make-log-symbol` *data-set* [*Function*]

Generates a buffer name for the data-set log buffer.

`make-main-user-action-menu` [*Function*]

Initializes the primary user interface menu.

`make-new-buffer-symbol` [*Function*]

Generates a buffer name for the current edit buffer.

`make-new-rtm-table-symbol` [*Function*]

Generates a unique symbol for the RTM table and pushes it onto `*rtm-table-list*`.

`make-partial-multiple-data-set` *&optional (data-set-name nil)* [*Function*]

Presents the multiple data-set choice menus and builds a new data-set from the selected variables. The only values in the new data set are the selected variables. If data-set-name is not provided an internal symbol is generated.

`make-partitioned-data-set` *&optional (data-set-name nil)* [*Function*]

Creates a new data-set by selecting on some property from the RTM table and inserting all variable values into the new data-set. Useful for seperating treatments i.e. 'Burn first?' If data-set-name is not provided a symbol is generated for it.

`make-plot-window` *&optional (main? nil)* [*Function*]

Creates an instance of a plotting window. If *main?* is `Nil` the window is positioned by the mouse.

`make-plot-window-list` [*Function*]

Generates the selection list for the plot window menu.

`make-rtm-attribute-list` *&optional (data-set *current-data-set*)* [*Function*]

Generates the list of attribute names and types for use by `make-rtm-table`.

`make-rtm-table` *&optional (data-set *current-data-set*)* [*Function*]

Creates an RTM table using a generated name and the current data-set. Uses `make-rtm-attribute-list` to define the table attributes.

`make-variable-menu` *&key (data-set *current-data-set*) (label nil)* [*Function*]

Binds a menu to a data-set for selecting values from it.

`make-variable-menu-attribute-list` *&optional (data-set *current-data-set*)* [*Function*]

Generates the attribute list for construction of the variable menu for each data set.

`make-variable-symbol` *prefix-string* [*Function*]

Interns a symbol of the form <prefix-string>-<counter>.

`make-variable-symbols` *data-set* [*Function*]

Generates a list of internal symbol names for the values of a data-set. This is used at load time by `load-data-set`.

5.5 File Handling Functions

`check-same-items` *filenames* [*Function*]

Before loading data from two or more files, this checks to see if the column ID strings at the front of each file are the same, i.e., whether the columns we are about to load have the same name-strings in each file. There is no other check for accuracy, so two different variables which have the same name and position could be combined into a single data set.

`load-data-set` *&key (filenames-list nil)* [*Function*]
*(extension *clasp-data-extension*)*

Loads a data-set from a file. Generates a unique name for the data and adds it to the menu list of data-sets. Allows for multiple data-sets in use at one time. Sets the value of `*current-data-set*` and pushes the data-set name onto the `*data-set-list*`. If filenames-list is nil `select-data-files` is called, presenting the directory menu.

`load-data-from-files` *&key (filenames-list nil)* [*Function*]
*(extension *clasp-data-extension*)*

Loads multiple files into a single data-set, requires that all of the files have exactly the same strings for the variable names, otherwise an error is returned. If *filenames-list* is `nil` `select-data-files` is called presenting the directory menu for data file selection.

`with-conditional-open-file` *(stream filename &rest options) &body body* [*Macro*]

WITH-CONDITIONAL-OPEN-FILE (stream filename [{option}*]) {form}*
A slightly modified version of the Common Lisp standard. Executes *form*'s with the *stream* variables bound to a stream for the corresponding file *filename*. *filename* is opened using *options*, which are the same as for the `OPEN` function. If *filename* is `nil`, the `OPEN` is bypassed and *stream* is bound to `nil`.

`with-multiple-open-streams` *(stream stream-list &rest options) &body body* [*Macro*]

Iterates over a list of streams executing the forms of body for each stream. Allows echoing of output to a log buffer or file. This is a multiple stream version of `with-conditional-open-file`.

`write-ascii-file-or-buffer` *&key (data-set nil) (padchar " ")* [*Function*]

Write the data-set to an ASCII file delimited by a user specified character. The default is a space character. If no data-set is given `select-current-data-set` is called.

`write-lisp-forms-file-or-buffer` *&optional (data-set nil)* [*Function*]

Write the data-set to a Lisp form file, with the same format as the raw data files. Partitioned data sets have a title string stating which file it came from and the selection criterion. If no *data-set* is given `select-current-data-set` is called.

5.6 Selection Functions

`enter-name` [*Function*]

Displays a menu for setting the name of a multiple file data-set.

`enter-value` [*Function*]

Displays a menu for setting the value to use for the RTM table selection.

`expand-where-clause` *list &optional (data-set *current-data-set*)* [*Function*]

Catches the close of an .and. or .or. allowing the building of complicated queries.

`get-data-or-function` *&key (data-p t) (data-set *current-data-set*)* [*Function*]

Displays a menu of either the available variables or functions and returns the selected value(s).

`get-data-set-filename` *var-symbol* [*Function*]

Takes an internal variable symbol and returns the filename associated with its data set.

`get-data-set-fun-data` [*Function*]

Presents three menus. The first selects the active data set, the second the function(s), and the third the variable(s). The functions are applied to the variables and the results returned.

`get-key-item` *&optional (data-set *current-data-set*)* [*Function*]

Selects a variable from the *current-data-set*.

`get-levels` *&optional (data-set *current-data-set*)* [*Function*]

Returns a list of old values for the selected variables for transforming scores.

`get-multiple-data-set-fun-data` [*Function*]

Presents menus to select a function, a data-set, a variable value, and then repeats data-set variable until complete is selected. Evaluates selected functions on all selected data.

`get-multi-set-data` *&optional (funcall-p t)* [*Function*]

Presents a menu for selecting a data-set for partitioning or multiple data-set function calls.

`get-rtm-data` *&optional (data-set *current-data-set*)* [*Function*]

Select a table from a menu and do a selection of specified variables. Limited functionality.

`get-rtm-modifier` [*Function*]

Displays a menu for selecting an RTM modifier.

`get-treatment` *&optional (data-set *current-data-set*)* [*Function*]

Returns a list of the levels of a discrete variable for partitioning a variable. With the partitioned variable an *ANOVA* can be performed.

`get-user-action` [*Function*]

Displays a menu of possible actions, covers most basic uses of the system.

`get-variable` *&optional (data-set nil) (label nil)* [*Function*]

Prompts for variable selection from the data set *data-set* with the label *label*. If no *data-set* is given `select-current-data-set` is called.

`get-var-name-string` *var-symbol &optional (data-set nil)* [*Function*]

Takes an internal variable symbol and returns the name string of the variable.

`get-var-symbol` *name-string data-set* [*Function*]

Takes a name string and a data-set name and returns the internal symbol for that name.

`make-where-clause` *&optional (label nil) (data-set *current-data-set*)* [*Function*]

Presents the variable selection menu then presents the RTM selection operators menu. Returns a where-clause for selection from the RTM table.

`select-current-data-set` [*Function*]

Displays a menu of the available data sets and returns the selected data set. This does not set the value of `*current-data-set*`.

`select-data-files` *&key (pathname *clasp-data-directory*)* [*Function*]
*(extension *clasp-data-extension*)*

Presents a directory window for selecting data files with the mouse.

`select-partition-data` *&optional (data-set *current-data-set*)* [*Function*]

Generates the partition data by selecting from the RTM table based on menu choices. Creates appropriate variable names indicating origin.

`select-plot-window` [*Function*]

Presents a menu for selection of a previously prepared plot to view.

`set-global-variable-values` [*Function*]

Presents an edit menu for entering the values of the default directory paths for raw data files and RTM table data files.

5.7 Help Functions

`get-topic` [*Function*]

Presents a menu for selecting a topic.

`help-me` *&optional (topic nil)* [*Function*]

Goes to the help file on disk and returns the appropriate section for the *topic*. If no *topic* is given the list of all available topics is presented for selection.

`read-from-help-file` *topic* [*Function*]

Probes for the help file. If found reads to the topic header and then sends the topic text to the help-window for display.

`show-help` *topic* [*Function*]

Opens the help file and displays the section in a buffer window.

`sorry-no-help` *topic* [*Function*]

No help available for the requested topic.

`topic-p` *topic* [*Function*]

Determines if a string is a valid topic name.

5.8 Flavor Methods

`:after :deselect` *&rest ignore* [*Method of* `help-window`]

Causes the help window to return to the caller when it is deselected, ending the `:edit-string` process.

`:display-help` *topic* [*Method of* `help-window`]

Ensures an empty io-buffer before editing a help topic.

`:mouse-click` *button x y* [*Method of* `clasp-window`]

Process mouse clicks to bring up different system function menus in the *Clasp Interaction* window.

`:mouse-click` *button x y* [*Method of* `help-window`]

Processes mouse clicks in the *Help* window, making the help menu and the *SYSTEM* menu available.

`:wrapper :who-line-documentation-string` *&body body* [*Method of* `clasp-window`]

Sets up the who-line documentation for the *Clasp Interaction* window.

`:wrapper :who-line-documentation-string` *&body body* [*Method of* `help-window`]

Sets up the who-line documentation for the *Help* window.

5.9 Reporting Functions

`check-stream-status` [*Function*]

Reports which streams are being echoed to, or CLASP interaction window if none.

`column-number` *var-symbol* [*Function*]

Takes a variable symbol and returns the data-set array index and the data set name.

`column-values` *column &optional (data-set *current-data-set*)* [*Function*]

Returns a list of the values for a single variable.

`disable-echoing-report-stream` [*Function*]

Sets `*reports-to-log-p*` and/or `*reports-to-ed-buffer-p*` to `nil` based on menu choice.

`echo-report-stream-to-ed-buffer` [*Function*]

Sets `*reports-to-ed-buffer-p*` to `True`, echoing reports to `*current-ed-buffer*`.

`echo-report-stream-to-log-buffer` [*Function*]

Sets `*reports-to-log-p*` to `True` echoing reports to the data set log buffer.

`log-actions` *data-set &rest forms* [*Function*]

Writes the *forms* to the log buffer for the *data-set*, recording activity.

`print-all-data` *&key (data-set nil) (stream-name *clasp-report-stream*)* [*Function*]

Prints variable names and data for all variables in a *data-set* in tabular format. If no *data-set* is given `select-current-data-set` is called.

`print-column-data` *column-number &key (stream-name *clasp-report-stream*)* [*Function*]
*(data-set *current-data-set*)*

Displays the data for an individual column of a data set. Optionally sends to an alternative stream.

`print-row-data` *row &key (data-set *current-data-set*)* [*Function*]
*(stream-name *clasp-report-stream*)*

Sends the data for a specific row to the specified stream.

`print-variable-symbol-name` *&key (data-set nil)* [*Function*]
*(buffer *clasp-report-stream*) (var-symbol nil)*

Displays or sends to a buffer the listing of internal symbols and the associated variable name strings. With no arguments the values are displayed for all loaded data-sets. An optional selection of an individual variable symbol and/or data-set is available.

`report` *results &optional (stream-name *clasp-report-stream*)* [*Function*]

Returns a sentence describing each evaluation of a group of functions on a group of data. Can be sent to an optional file or buffer.

`report-regression-results` *x-list y-sym int coefs r-list t-bs betas r-square f* [*Function*]
ss-percent-list ss-res ss-reg mse
*&optional (stream-name *clasp-report-stream*)*

Formats the results of a multi-variate linear regression for display. Correlations given in sets of two. Can be redirected to a different buffer, or the log buffer.

`report-results` *results &optional (stream-name *clasp-report-stream*)* [*Function*]

Returns a sentence describing each evaluation of a group of functions on a group of data. Can be sent to an optional file or buffer.

`row-values` *row &optional (data-set *current-data-set*)* [*Function*]

Returns a list of the values of a given row. This is the same as the original format of the data file.

5.10 Transformation Functions

`delete-columns` *&optional (data-set nil)* [*Function*]

Deletes one or more variables from the data set and rebinds the variable symbols. This is a destructive operation which resizes the data array. If no *data-set* is specified a menu is presented to choose one.

`delete-row` *row-number data-set* [*Function*]

Deletes a row from the data set and rebinds the variable symbols. This is a destructive operation which resizes the data array.

`delete-rows-with-missing-values` [*Function*]

Deletes all rows in which the selected variables have missing values. Updates the RTM table and resizes the data array.

`hours-to-minutes` [*Function*]

Replaces all selected values in the selected variable with minutes for hours. Updates the RTM table for the data set.

`minutes-to-hours` [*Function*]

Replaces all selected values in the selected variable with hours for minutes. Updates the RTM table for the data set.

`nreplace-column-value` *(column-number old-value new-value &key (row nil) (data-set *current-data-set*) (test (quote equal)))* [*Macro*]

Changes the values in a column, does not rebind the variable-symbols. This is a destructive operation. Does not update the RTM table for the data set.

`nreplace-with-mean` [*Function*]

Replaces all selected values in the selected variable with that variable's mean. Updates the RTM table for the data set.

`ntransform-column-values` [*Function*]

Takes a list of lists of pairs of values for transforming a column's values. Updates the RTM table for the data set.

`ntransform-raw-score-to-log-score` *var-symbols* [*Function*]

Destructively replaces raw-scores with the natural log of the scores. Updates the RTM table for the data set.

`ntransform-raw-score-to-z-score` *var-symbols* [*Function*]

Destructively replaces raw-scores with standardized 'z' scores. Preserves the distribution of the original data. Updates the RTM table for the data set.

`replace-nil-with-mean` *varlist &optional (new-value nil)* [*Function*]

Non-destructively replaces missing values, `nil` with the mean of the sequence or, if one is given, an optional alternate value. Returns the new sequence.

`update-rtm-table` *attribute old-value new-value data-set* [*Function*]

Update the values in the RTM table after a transformation.

5.11 Data Types

`clasp-window` [*Flavor*]

Window flavor for the interaction window. Composed of the flavors `w:lisp-listener` and `w:stream-mixin`.

`data-set` [*Structure*]

Primary data structure for raw data. Contains pointers to the RTM table, internal symbol names list, and the variable choice menu.

`help-window` [*Flavor*]

Standalone editor window for displaying the help text. Automatically ends the edit sequence when deselected.

5.11.1 data-set Slot Functions

`data-set-data-array` *data-set* [*Function*]

Array of variable values.

`data-set-filename` *data-set* [*Function*]

File from which the data was loaded.

`data-set-header-string` *data-set* [*Function*]

First string from the data file, containing identification information.

`data-set-log-buffer` *data-set* [*Function*]

Buffer for logging actions on file.

`data-set-name-array` *data-set* [*Function*]

Array of variable name strings.

`data-set-rtm-table` *data-set* [*Function*]

Internally generated symbol naming the RTM table associated with the data set.

`data-set-title` *data-set* [*Function*]

Internally generated symbol which names the data set.

`data-set-variable-menu` *data-set* [*Function*]

Menu resource for selecting variable values by name string.

`data-set-variable-symbols` *data-set* [*Function*]

List of internally generated symbols associated with column values and used as the RTM table attribute names.

5.12 Variables

`*buffer-counter*` [*Variable*]

Next value to use for generating a *data-buffer* symbol.

`*clasp-data-directory*` [*Variable*]

Default directory for data files.

`*clasp-data-extension*` [*Variable*]

Default extension for searching for data files.

`*current-data-set*` [*Variable*]

The data set which is currently in use.

`*clasp-error-stream*` [*Variable*]

Stream for error messages.

`*clasp-notification-stream*` [*Variable*]

Stream for status messages.

`*clasp-plot-window*` [*Variable*]

Instance of a plotter window to which plots are sent.

`*clasp-plot-window-list*` [*Variable*]

List of multiple plot window instances for deallocation.

`*clasp-report-stream*` [*Variable*]

Stream for reports.

`*clasp-who-line-documentation*` [*Variable*]

Who line documentation string for `*clasp-window*`.

`*clasp-window*` [*Variable*]

The instance of the main interaction window.

`*current-data-set*` [*Variable*]

The data set which is currently in use.

`*current-ed-buffer*` [*Variable*]

Name of buffer to which reports will be sent.

`*data-set-counter*` [*Variable*]

Next value to use for generating a 'data-set' symbol.

`*data-set-list*` [*Variable*]

The list of available data set names.

`*fun-menu*` [*Variable*]

Menu of statistical functions.

`*help-file*` [*Variable*]

Filename of the help file.

`*help-topics-list*` [*Variable*]

List of available help topics in the help file.

`*help-window*` [*Variable*]

Instance of the help display window.

`*main-user-action-menu*` [*Variable*]

Menu of user actions.

`*null-counter*` [*Variable*]

Next value to use for generating a symbol for a string which does not have its own counter.

`*reports-to-ed-buffer-p*` [*Variable*]

Predicate for sending reports to the `*current-ed-buffer*`.

`*reports-to-log-p*` [*Variable*]

Predicate for sending reports to the log buffer of a data set.

`*rtm-table-counter*` [*Variable*]

Next value to use for generating a RTM symbol.

`*rtm-table-list*` [*Variable*]

The list of available RTM tables.

`*value-to-select-on*` [*Variable*]

For entering a data value to partition a data set.

`*var-counter*` [*Variable*]

Next value to use for generating a *var* symbol.

Appendix A

Regression Analysis: Matrix Method*

To implement *linear regression* in CLASP the decision was made to use matrix algebra methods for the calculation of the statistics. The basis of the method is in the ability to calculate the *Sum of Squares* and *cross products* for the independent and dependent variables through matrix multiplication, then using these values to calculate the *regression coefficients*, *correlation coefficients*, and percentage *variance* of the dependent variable, Y. The advantage of using matrix algebra over direct raw score calculation is not obvious when considering an example with only two independent variables, however the complexity of the direct calculations becomes apparent as the number of independent variables increase. The following formulae illustrate the calculations involved using the direct raw score method, for two independent, X, variables, with $\sum x_i$ the sum of deviation scores for the i'th X variable, and $\sum x_i y$ the sum of the cross products of the deviation scores of Y and the i'th X variable.

$$b_1 = \frac{(\sum x_2^2)(\sum x_1 y) - (\sum x_1 x_2)(\sum x_2 y)}{(\sum x_1^2)(\sum x_2^2) - (\sum x_1 x_2)^2}$$

$$b_2 = \frac{(\sum x_1^2)(\sum x_2 y) - (\sum x_1 x_2)(\sum x_1 y)}{(\sum x_1^2)(\sum x_2^2) - (\sum x_1 x_2)^2}$$

As can be seen these equations expand with the introduction of each new independent variable, making them very unwieldy. In order to perform the calculations with matrices we first introduce a new variable, X_0, a unit vector (a vector of ones), of the same size as the vectors X_i, with this the matrix $\mathbf{Z}$ is built, with the columns being defined as $X_0 \cdots X_i$ and Y. Using the equation

$$Y = a + b_1 X_1 + b_2 X_2 + \cdots + b_k X_k + e$$

in matrix form, where $\mathbf{Y}$ is a $N \times 1$ column vector, $\mathbf{X}$ is a $N \times k$ matrix with each column corresponding to X_i, where X_0 is the unit vector described above, a is the first element of the $1 \times k$ row vector of regression coefficients, $\mathbf{b}$, and $\mathbf{e}$ is the $1 \times k$ column vector of the error terms of the equation, given by the form $Y = \hat{Y} + e$, where $\hat{Y}$ is the predicted value of Y

*This discussion is a condensation of chapters 2 and 3 from Elazar J. Pedhazur, *Multiple Regression in Behavioral Research*, 1973.

from the regression equation, produces the following equation:

$$\begin{matrix} Y \\ \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ \vdots \\ Y_N \end{bmatrix} \end{matrix} = a + \begin{matrix} b \\ \begin{bmatrix} a \\ b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix} \end{matrix} \begin{matrix} X \\ \begin{bmatrix} X_{10} & X_{11} & X_{12} & \cdots & X_{1k} \\ X_{20} & X_{21} & X_{22} & \cdots & X_{2k} \\ X_{30} & X_{31} & X_{32} & \cdots & X_{3k} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ X_{N0} & X_{N1} & X_{N2} & \cdots & X_{Nk} \end{bmatrix} \end{matrix} + \begin{matrix} e \\ \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ \vdots \\ e_N \end{bmatrix} \end{matrix}$$

To calculate the values of $\mathbf{b}$ and $\mathbf{e}$ we use the sum of squares and cross product matrix produced by multiplying the transpose of the matrix $\mathbf{Z}$, $\mathbf{Z}'$, by $\mathbf{Z}$, $\mathbf{Z}'\mathbf{Z}$. As can be seen the result of this multiplication contains the summed squares of the individual variables along the diagonal, with $\sum X_0^2 = \sum X_0$, and the cross products along the off diagonals.

$$\mathbf{Z}'\mathbf{Z} = \begin{bmatrix} \sum X_0 & \sum X_1 & \sum X_2 & \cdots & \sum X_k & \sum Y \\ \sum X_1 & \sum X_1^2 & \sum X_1 X_2 & \cdots & \sum X_1 X_k & \sum X_1 Y \\ \sum X_2 & \sum X_1 X_2 & \sum X_2^2 & \cdots & \sum X_2 X_k & \sum X_2 Y \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum X_k & \sum X_k X_1 & \sum X_k X_2 & \cdots & \sum X_k^2 & \sum X_k Y \\ \sum Y & \sum Y X_1 & \sum Y X_2 & \cdots & \sum Y X_k & \sum Y^2 \end{bmatrix}$$

From this we get the two component matrices $\mathbf{X}'\mathbf{X}$, X transpose X, and $\mathbf{X}'\mathbf{Y}$, X transpose Y,

$$\mathbf{X}'\mathbf{X} = \begin{bmatrix} \sum X_0 & \sum X_1 & \sum X_2 & \cdots & \sum X_k \\ \sum X_1 & \sum X_1^2 & \sum X_1 X_2 & \cdots & \sum X_1 X_k \\ \sum X_2 & \sum X_1 X_2 & \sum X_2^2 & \cdots & \sum X_2 X_k \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum X_k & \sum X_k X_1 & \sum X_k X_2 & \cdots & \sum X_k^2 \end{bmatrix} \quad \mathbf{X}'\mathbf{Y} = \begin{bmatrix} \sum Y \\ \sum X_1 Y \\ \sum X_2 Y \\ \vdots \\ \sum X_k Y \\ \sum Y^2 \end{bmatrix}$$

which are then used to determine the matrix $\mathbf{b}$, using the following equation:

$$\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$$

where $(\mathbf{X}'\mathbf{X})^{-1}$ is the inverse of the matrix $\mathbf{X}'\mathbf{X}$. Recall that the inverse of a matrix is a matrix such that $\mathbf{X}^{-1}\mathbf{X} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. So multiplying by the inverse of a matrix is equivalent to division. The significance of the coefficients can be tested using a standard t test, using the formula $t = \frac{b}{s_b}$, where s_b is the error term of the coefficient. The value of t indicates whether or not the coefficient is significantly different from zero, with a critical value at the 0.05 level of 1.96 for a two tailed test. A two tailed test is used as it does not matter whether the coefficient is positive or negative. With $\mathbf{b}$ the sum of squares of the regression and the sum of squares of the residual are calculated, with:

$$ss_{reg} = \mathbf{b}'\mathbf{X}'\mathbf{Y} - \frac{(\sum Y)^2}{N}$$

$$ss_{res} = \mathbf{e}'\mathbf{e} = \sum e^2 = \mathbf{Y}'\mathbf{Y} - \mathbf{b}'\mathbf{X}'\mathbf{Y}$$

where $\mathbf{e'e}$ is the sum of squared error terms for the regression. The sum of squares of the regression indicate what portion of the total squared deviation scores, $\sum y^2$, of the dependent variable, are due to the effects of the independent variables. The sum of squares of the residual indicate the proportion which is unaccounted for by the independent variables. Recall that

$$ss_{total} = ss_{reg} + ss_{res} = \sum y^2 = \mathbf{Y'Y} - \frac{(\sum Y)^2}{N}$$

using these values the significance of the regression can be tested.

The key value of interest from the regression is the F statistic of the R^2 value, which is the percentage of the variance accounted for by the regression. The value of R^2 can be calculated using either of the following two equations:

$$R^2 = \frac{\mathbf{b'X'Y} - \frac{(\sum Y)^2}{N}}{\mathbf{Y'Y} - \frac{(\sum Y)^2}{N}} = \frac{ss_{reg}}{\sum y^2}$$

$$R^2 = 1 - \frac{\mathbf{e'e}}{\mathbf{Y'Y} - \frac{(\sum Y)^2}{N}} = 1 - \frac{ss_{res}}{\sum y^2}$$

where $\sum y^2$ is the sum of squared deviations for the dependent variable Y. The value of F indicates the probability that such a value could occur due to chance. F is calculated as:

$$F = \frac{ss_{reg}/df_{reg}}{ss_{res}/df_{res}} = \frac{R^2/k}{(1-R^2)/(N-k-1)}$$

where k is the number of independent variables and N is the number of observations. Additionally the coefficients of *correlation* between each X_i and Y, as well as between the X_i's can be calculated, with the squared coefficient for each of the X_i's indicating its contribution to the *variance* of Y.

It is possible, as with other statistics, to produce results which are significant, but not meaningful, as well as meaningful, but not significant. Evaluating the result of a *linear regression*, especially when there is interaction between the independent variables, can be a very difficult task. Often a small value of R^2 can discourage a researcher, even though its value is significant, and, by the same token, a large value of R^2 does not immediately confer meaning to the results of the regression. In cases where the results of the analysis are not clear, or appear to be at a border line, a useful strategy can be to reduce the number of independent variables being used, looking for a change in the analysis as the interactive effects are reduced. The *covariance* of the independent variables, which is indicated by the *correlation* between the X_i's, is a good indicator of the interaction between the independent variables. In general it is best to refer to a statistics text when evaluating the results of a linear regression.

Appendix B

Automated Testing of CLASP*

B.1 The Automated Tester

The design of the data structures is extremely simple and neat. Structures and files are the only data-structures. Associated with each statistic there is a structure to store the output of executing each statistic. The second structure, called *test-suite*, defines the name of the statistical function, the arguments to test the function, the name of the data-file, the name of the key-file, the name of the results-file, and the transform slot. The automated tester uses three files; a *data-file* with extension *".clasp"*, a *key-file* with extension *".key"*, and a *results-file* with extension *".results"*.

The key files are created only once in the very beginning, using the available data set. The function `create-clasp-key-files` iterates over the list of statistics and creates the "**.key*" files for each of them by calling the function `create-key-file`.

Once the data-files and the key-files are available, the automatic tester can be executed by calling the function `test-clasp`. The tester executes each of the statistic, and fills the slots with the results. The stuctures are then written into the corresponding results-file. Once the results-files are available, they are tested against the key-files, and the results of the validation are reported. The results can be reported in one of two ways, depending on the user's choice. One way is to simply report whether or not each statistic passed the validation test. Another way is to give a detailed slot-wise comparison report for each statistic. In either case if there is a discrepancy in the comparison, the contents of both, the results and the key files are reported. The user can change the required precision by setting `*epsilon*`.

To use the automated tester the file "*clasp:testing:automated-testing*" must be loaded. To test all of the statistics execute the form `(test-clasp)`, which takes an optional key argument `:verbose`. This tests the software against the already available key files, and reports the results. If `:verbose` is `nil` it simply reports whether or not a statistic is validated. If `:verbose` is `t`, it gives a detailed slot-wise comparison report.

*This chapter was authored with Sameen Fatima, who implemented the automated testing routines.

B.2 Functions

B.2.1 Data Structures Macros

`defstatstruct` *stat &rest slots* [*Macro*]

Executes a `defstruct` for the statistic data structure and generates a macro for positional construction of an instance of the structure, a BOA constructor. The structure is given the name *stat*-`struct` with slots *slots*. The constructor is of the form:
`make-the-'stat'-struct` *slot-values-list*
where the *slot-values-list* contains a filler for each slot in the order they were declared in the original `defstatstruct`. The declaration must order the slots according to the order of the values returned by the statistic *stat*.

`call-stat-struct` *statistic arg-list* [*Macro*]

Given the name of a statistic and a list of its arguments, makes an instance of the stat-struct with the slots filled with the result of applying the statistic to the arguments. Each of the statistics returns multiple values.

B.2.2 Comparison Functions

`compare-stat-struct` *struct-1 struct-2 report-stream &optional (epsilon *epsilon*)* [*Function*]

Compares two stat-structs for a match. Used to compare the key struct with the result struct. Performs slot by slot comparison, reporting success or failure. Uses *epsilon* as the required precision for numeric comparisons, differences less than this are ignored.

`struct-slot-value` *slot-name struct* [*Function*]

Returns the value of the slot named *slot-name* from the structure *struct*.

`slot-equal` *slot struct1 struct2* [*Function*]

Slot comparison function, handles any Lisp data type.

`statstruct-slots` *struct-type* [*Function*]

Calls the internally generated function for the structure of type *struct-type*, which returns the list of slot-names for it.

`structure-equal` *struct1 struct2* [*Function*]

Performs an `equalp` comparison of two structures. For clarity of code.

`compare-results` *key-file results-file* [*Function*]

Predicate on the contents of the two files. Returns `True` if and only if the two items are `structure-equal`.

`compare-the-results` *key-file results-file report-stream* [*Function*]

Predicate on the contents of the two files. Performs a slot-wise comparison, using the limits described in `compare-stat-struct`. Returns `True` if and only if every slot is within the *epsilon* for comparison.

B.2.3 Input/Output Functions

`write-results-of-the-test` *stat arg-list file-name* [*Function*]

Evaluates the application of the statistic to the *arg-list*, writing the resultant structure to the file *file-name*.

`write-structure-to-file` *structurename filename* [*Function*]

Writes a structure to the file *filename*.

`read-structure-from-file` *filename* [*Function*]

Reads a structure from the file *filename* and returns it.

B.2.4 Translation Functions

`quoted-symbol-p` *arg* [*Macro*]

Returns `True` if and only if *arg* is a quoted symbol, i.e. of the form `(quote foo)`. Evaluates its argument only once.

`quoted-list-p` *arg* [*Macro*]

Returns `True` if and only if *arg* is a quoted list, i.e. of the form `(quote (foo))`. Evaluates its argument only once.

`unquote` *sym-list* [*Function*]

Hack to unquote symbols in a sublist. Used for `anova2way`.

`eval-arg-list` *arg-list* [*Function*]

Replaces each instance of an internal variable symbol with the values represented by that symbol. Quotes the resultant sequences as needed.

`translate-arg-list` *arg-list* [*Function*]

Replaces the string name of each of the arguements in arg-list with the data-set symbols generated at the time the data set is loaded for the test. Handles sublists of args, quotes symbols and the top level list of any sublist, removes quotes from within any sublist.
Examples:

```
("long jump" "year")  ⇒  ('var-1 'var-2)
(("x" "y") "z")       ⇒  ('(var-3 var-4) 'var-5)
```

B.2.5 CLASP Testing Functions

`test-clasp` *&key verbose* [*Function*]

Checks if a detailed slot-wise validation report is required or not, and then goes ahead and tests CLASP.

`test-clasp-yes-no` *&optional (report-stream *clasp-report-stream*)* [*Function*]

Iterates over the list of statistics, reporting whether or not the statistic passed the test.

`report-error` *stat* [*Function*]

Reports the failure of a test for the statistic stat.

`test-statistic` *stat report-stream* [*Function*]

Takes a test-suite structure, *stat*, and tests the given statistic.

`test-statistic-and-report` *stat report-stream* [*Function*]

Takes a test-suite and performs a slot-wise test. Reports results for each slot.

`test-clasp-and-report` *&optional (report-stream *clasp-report-stream*)* [*Function*]

Iterates over the stat-list, testing each statistic in a slot-wise fashion.

B.2.6 Key File Creation Functions

`create-key-file` *stat* [*Function*]

Creates a key file by executing the statistic, writing the results to a *".key"* file.

`create-clasp-key-files` [*Function*]

Iterates over the stat-list creating *".key"* files.

B.3 Data Structures

B.3.1 statstruct

There are 11 structures defined corresponding to each of the 11 statistics listed in the variable `stat-list`. These structures are meant to output the results of executing the statistic functions. The macro `defstatstruct` is used for defining these 11 structures. Below is an example for defining the structure to output the results of the test t-statistic-matched. This generates a structure by the name t-statistic-matched-struct.

```
(defstatstruct  t-statistic-matched
                t-statistic
                sample-error
                degrees-of-freedom
                significance)
```

B.3.2 test-suite

This structure defines the test-suite. Below is the definition, along with the comments for each slot.

```
(defstruct  test-suite
            statistics    ;; the function name used in Clasp
            arg-list      ;; arguments list
            data-file     ;; the data set for the test
            key-file      ;; the key or the solution
            results-file  ;; the results of the function
            transform)    ;; function for translating the args.
```

B.4 Variables

`stat-list` [*Variable*]

Defines list of statistics which can be tested with the automated tester.

`*epsilon*` [*Variable*]

Defines the required precision depending on the computer and the user's choice. Set to double-float-epsilon by default.

Appendix C

Testing and Validation*

C.1 Regression

C.1.1 Univariate

Univariate regression was tested with a set of related variables and a set of random variables. Test Data Set 1 (from DataDesk sample data) is the related set and is shown in Figure C.1. This is the gold medal performance in the long jump for the Olympic games from 1900 to 1984. Test Data Set 2, Figure C.1, contains randomly generated data.

Figures C.3 and C.5 compare the performance of CLASP and Statview for univariate regression. The results are the same to at least 3 significant figures. The only difference is the sum of squares of the regression. This is probably due to rounding error.

*Sample test sets are taken from DataDesk sample data, by hand, and from **Biometry** by Sokal and Rohlf. This section was authored with Sameen Fatima and Victor Ng of EKSL, who performed all of the testing.

Test Data Set 1

Year Since 1900	Long Jump Distance
−4	249.75
0	282.875
4	289
8	294.5
12	299.25
20	281.5
24	293.125
28	304.75
32	300.75
36	317.3125
48	308
52	298
56	308.25
60	319.75
64	317.75
68	350.5
72	324.5
76	328.5
80	336.25
84	336.25

Test Data Set 2

x	y
5	41
8	25
10	37
11	42
19	28
19	10
23	40
30	20
25	8
40	8
38	22
34	23
22	21

Figure C.1: Test Data Sets 1 & 2

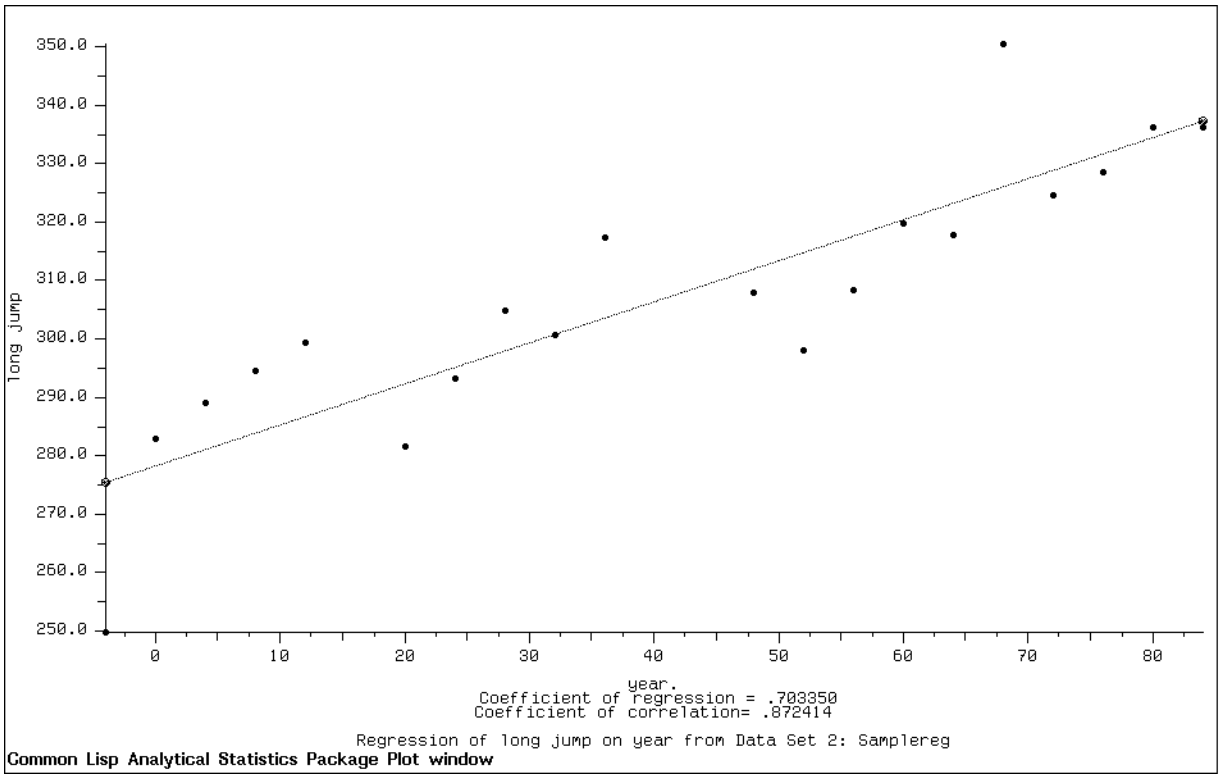

Figure C.2: Scatter Plot of Regression *long jump* on *year* from CLASP

The regression of long jump on year from data-set Samplereg

		CLASP Result	Statview Result
The intercept	=	278.191	278.191
The regression coefficient b	=	0.703	0.703
The mean square error	=	127.322	
The error of the coefficient	=	4.484	
The error of the intercept	=	0.090	
The variance of long jump	=	506.319	
The sum of squares of the regression	=	7707.250	7707.437
The F statistic for the regression	=	57.347	57.346
The correlation coefficient	=	0.872	

Figure C.3: Univariate Regression Test Data Set 1

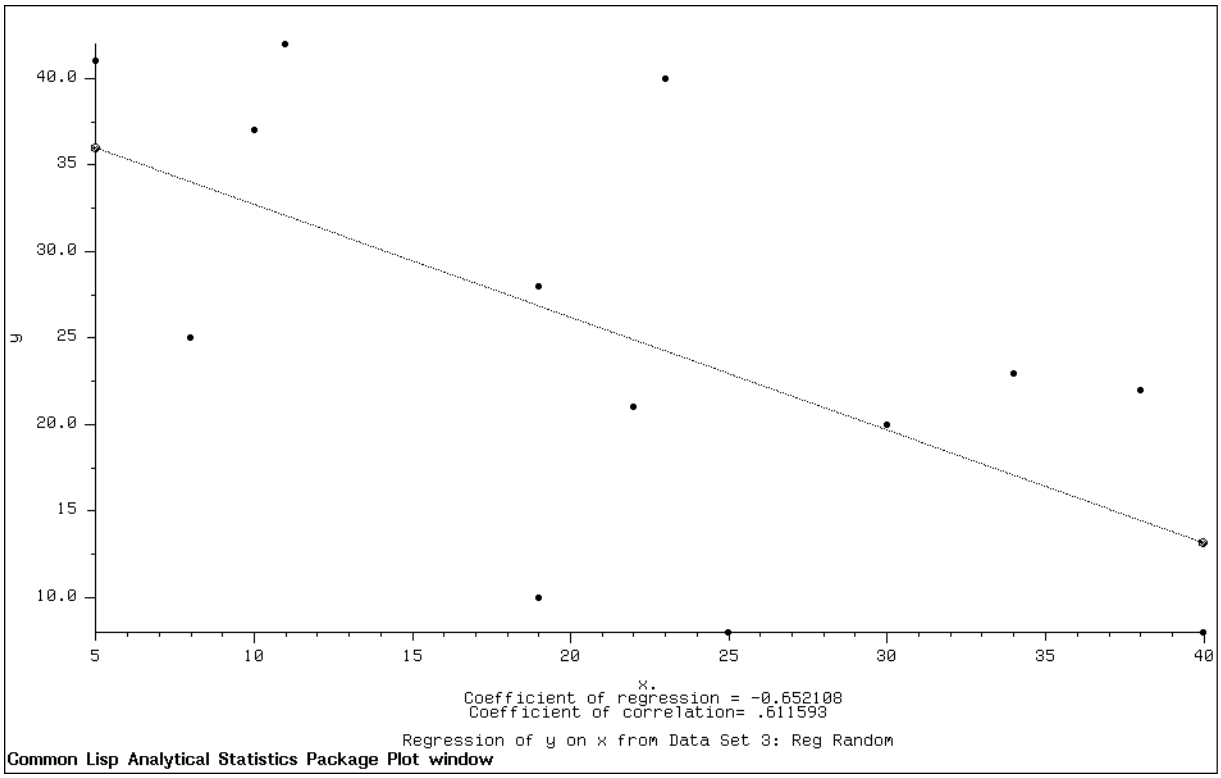

Figure C.4: Scatter Plot of Regression y on x from CLASP

The regression of y on x from data-set Reg-Random

		CLASP Result	Statview Result
The intercept	=	39.246	39.246
The regression coefficient b	=	−0.652	−0.652
The mean square error	=	92.850	
The error of the coefficient	=	5.954	
The error of the intercept	=	0.244	
The variance of y	=	136.923	
The sum of squares of the regression	=	665.802	665.802
The F statistic for the regression	=	6.573	6.573
The correlation coefficient	=	0.612	

Figure C.5: Univariate Regression Test Data Set 2

Test Data Set 3

Unemployment	FRB Index	Year From 1950
3.1	113	1
1.9	123	2
1.7	127	3
1.6	138	4
3.2	130	5
2.7	146	6
2.6	151	7
2.9	152	8
4.7	141	9
3.8	159	10

Test Data Set 4

x	y	z	zz
5	41	6	32
8	25	9	9
10	37	10	37
11	42	17	35
19	28	21	21
19	10	20	11
23	40	27	32
30	20	27	15
25	8	30	4
40	8	38	13
38	22	0	40
34	23	1	9
22	21	5	40

Figure C.6: Test Data Sets 3 & 4

C.1.2 Multivariate

Test Data Set 3 (from DataDesk sample data) is shown in Figure C.6. This is the set of related variables. Three economic variables during the decade of the 1950's. FRB Index is a measure of industrial production. Unemployment is the dependent variable. Test Data set 4 in Figure C.6 shows randomly generated data, with y as the dependent variable.

The results generated by CLASP are compared with those of DataDesk in Figures C.7 and C.8. The intercept is the constant term in the linear formula, b is the coefficient of terms.

The results for the sum of squares of the regression and residual, the F statistics, and the intercepts are the same to three significant figures. The precision of R^2 is three decimal places in both CLASP and DataDesk, with the results within 0.001 of each other. The results for the coefficient and the t statistics are equivalent with rounding.

dependent variable: unemployment
independent variables: FRB, year from 1950

	Analysis of Variance			
	DF	Sum of Squares		Mean Square Error
		CLASP	DataDesk	
Regression	2	7.249	7.24976	0.125
Residual	7	1.127	1.12624	0.057

	CLASP	DataDesk
R-square:	0.865	0.866
The F statistic:	22.501	22.5
The intercept:	13.454	13.4539

Variable	b		Beta	t statistic of the b	
	CLASP	DataDesk		CLASP	DataDesk
Index year	0.659	0.659417	2.069	6.318	6.32
FR	−0.103	−0.103339	−1.562	−4.769	−4.77

Figure C.7: Multivariate Regression Test Data Set 3

dependent variable: y
independent variables: x, z, zz

Analysis of Variance

	DF	Sum of Squares		Mean Square Error
		CLASP	DataDesk	
Regression	3	1159.860	1159.86	51.678
Residual	9	620.140	620.15	0.922

	CLASP	DataDesk
R-square:	0.652	0.652
The F statistic:	5.611	5.611
The intercept:	25.384	25.384

Variable	b		Beta	t statistic of the b	
	CLASP	DataDesk		CLASP	DataDesk
zz	0.489	0.488785	0.537	2.433	2.43
z	−0.009	−0.008673	−0.009	−0.039	−0.039
x	−0.524	−0.524014	−0.491	−2.402	−2.40

Figure C.8: Multivariate Regression Test Data Set 4

C.2 Correlation

Test sets 3 and 4 are used for comparisons. Results from CLASP and that from DataDesk are identical. The correlation coefficients are the Spearman's Product Moment Correlation coefficients or the R-ratios. They are computed when selecting "multivariate linear regression" (as part of the information) or "correlation" from the main user menu. The results from CLASP match that from DataDesk.

C.3 Spearman's Rank Order Correlation

Test Data Set 5 is the same as Test Data Set 2. Test Data Set 6, in Figure C.11, introduces a column with many duplicates. The results are identical and are shown in Figures C.12 and C.13.

The correlation coefficents for				
	Index year		FRB	
	CLASP	DataDesk	CLASP	DataDesk
Unemployment	0.654	0.654	0.313	0.313
Index year	1.000	1.000	0.906	0.906
FRB	0.906	0.906	1.000	1.000
Percentage variance	0.428		0.098	

Figure C.9: Correlation Coefficients Test Data Set 3

The correlation coefficents for						
	zz		z		x	
	CLASP	DataDesk	CLASP	DataDesk	CLASP	DataDesk
y	0.648	0.648	−0.363	−0.363	−0.612	−0.612
zz	1.000	1.000	−0.437	−0.437	−0.220	−0.220
z	−0.437	−0.437	1.000	1.000	0.244	0.244
x	−0.220	−0.220	0.244	0.244	1.000	1.000
Percentage variance	0.420		0.131		0.374	

Figure C.10: Correlation Coefficients Test Data Set 4

x	y	same
5	41	5
8	25	5
10	37	5
11	42	5
19	28	5
19	10	7
23	40	5
30	20	5
25	8	5
40	8	5
38	22	5
34	23	5
22	21	5

Figure C.11: Test Data Set 6

	Unemployment		FRB		Index-year	
	CLASP	DataDesk	CLASP	DataDesk	CLASP	DataDesk
Unemployment	1					
FRB	0.297	0.297	1			
Index-year	0.564	0.567	0.915	0.915	1	

Figure C.12: Spearman's Rho Test Data Set 2

	x		y		same	
	CLASP	DataDesk	CLASP	DataDesk	CLASP	DataDesk
x	1					
y	−0.639	−0.639	1			
same	−0.309	−0.309	−0.116	−0.116	1	

Figure C.13: Spearman's Rho Test Data Set 6

P-Pupation Site (Depth)
S-Sex (Row)
H-Healthy, Po-Poisoned (Column)

		Mortality	
P	S	H	Po
IM	M	23.000	1.000
	F	15.000	5.000
AM	M	7.000	4.000
	F	3.000	5.000
OW	M	8.000	3.000
	F	5.000	3.000
OM	M	55.000	6.000
	F	34.000	17.000

Figure C.14: Loglinear Analysis Test Data Set

C.4 Log Linear Analysis

The Test Set in Figure C.14 is from **Biometry** by Sokal, Rohlf p.750. Intermediate results were validated against those in the book.

All of the intermediate steps except the three-factor interaction are the same in **Biometry** and CLASP. CLASP executes one less iteration in the calculation of γ. This is due to allowable difference which is used to control the iterations. When this value is less than 0.001 iterations stop.

After 4 iterations, $\gamma = 1.3655$ with $df = 3$, $p < 0.7136$. The iterations are shown in Figure C.15. CLASP stopped at iteration 4 so the expected frequencies, in Figure C.16 are slightly different than those given in **Biometry**.

Two Factor Interactions

Test of independence of Pupation site & Sex given the level of Mortality

G-AB[C]: $G = 2.8694$ [2.869] with $df = 6$, $p < 0.8251$

Test for 3-factor interaction

		CLASP	Biometry
Iteration 1,	$\gamma =$	2.193	2.193
Iteration 2,	$\gamma =$	1.392	1.392
Iteration 3,	$\gamma =$	1.366	1.366
Iteration 4,	$\gamma =$	1.3655	1.365
	$\gamma =$		1.365

Figure C.15: Iterations to find γ

CLASP		Biometry	
22.398	1.608	22.394	1.605
15.602	4.393	15.605	4.393
7.308	3.681	7.313	3.688
2.692	5.318	2.685	5.314
8.873	2.124	8.875	2.125
4.128	3.876	4.125	3.875
54.421	6.587	54.418	6.582
34.578	16.414	34.584	16.417

Figure C.16: Expected Frequencies 3-factor Interaction

Decrease in fit due to dropping Pupation Site-Sex {AB} terms from the model is

$$2.869 - 1.366 = 1.504 \text{ with } 6 - 3 = 3 \text{ degrees of freedom, } p < 0.825$$

Expected frequencies G-AB[C]:	
23.560	1.909
14.440	4.091
6.200	2.864
3.800	6.136
8.060	1.909
4.940	4.091
55.180	7.318
33.820	15.682

Test of independence of Pupation site & Mortality given the level of Sex

$$\text{G-A[B]C: } G = 11.6845 \text{ [11.684] with } df = 6,\ p < 0.0694$$

Decrease in fit due to dropping Pupation Site-Mortality {AC} terms from model is

$$11.684 - 1.366 = 10.319 \text{ with } 6 - 3 = 3 \text{ degrees of freedom, } p < 0.069$$

Expected frequencies G-A[B]C:	
20.860	3.140
13.103	6.897
9.561	1.439
5.241	2.759
9.561	1.439
5.241	2.759
53.019	7.981
33.414	17.586

Test of independence of Sex & Mortality given the level of Pupation Site

$$\text{G-[A]BC:} G = 15.3385[15.334] \text{ with } df = 4, p < 0.0040$$

Decrease in fit due to dropping Sex-Mortality {BC} terms from model is

$$15.338 - 1.366 = 13.973 \text{ [13.973] with } 4 - 3 = 1 \text{ degrees of freedom, } p < 0.004$$

Expected frequencies G-[A]BC:	
20.727	3.273
17.273	2.727
5.789	5.211
4.211	3.789
7.526	3.474
5.474	2.526
48.473	12.527
40.527	10.473

Test for complete independence of Pupation Site from Sex & Mortality

$$\text{G-A,BC: } G = 11.8285 \text{ [11.828] with } df = 9,\ p < 0.2232$$

Expected frequencies G-A,BC:	
21.093	3.175
12.928	6.804
9.108	1.371
5.582	2.938
9.108	1.371
5.582	2.938
53.691	8.082
32.907	17.320

Test for complete independence of Sex from Pupation Site & Mortality

$$\text{G-B,AC: } G = 15.4825 \text{ [not shown in book] with } df = 7,\ p < 0.0303$$

Expected frequencies G-B,AC:	
20.959	3.309
17.041	2.691
5.515	4.964
4.485	4.036
7.170	3.309
5.830	2.691
49.088	12.686
39.912	10.314

Test for complete independence of Mortality from Pupation Site & Sex

$$\text{G-C,AB: } G = 24.2976 \text{ [not shown in book] with } df = 7,\ p < 0.0010$$

Expected frequencies G-C,AB:

18.557	5.443
15.464	4.536
6.186	1.814
8.505	2.495
6.186	1.814
47.165	13.835
39.433	11.567

Test for complete independence of all 3 factors from each other. This is only meaningful if there were no two factor interactions.

G-A,B,C: $G = 24.4416$ [24.442] with $df = 10$, $p < 0.0065$

Expected frequencies G-A,B,C:

18.764	5.504
15.257	4.475
8.103	2.377
6.588	1.933
8.103	2.377
6.588	1.933
47.763	14.010
38.835	11.392

Results from CLASP
For dependent Data and independent Groups
from data-set Anova1 the one way ANOVA results are

Sum of Squares between groups:	1269.923
Sum of Squares within groups:	1641.053
Sum of Squares total:	2910.976
Mean Square Error between groups:	634.961
Mean Square Error within groups:	28.790
Mean Square Error total:	49.339
F statistic:	22.055
Degrees of freedom:	(2,57)

Results from DataDesk
Analysis of Variance For Data:

Source	df	Sum of Squares	Mean Square	F-ratio	Prob
Grp	2	1269.92	634.961	22.055	0.0000
Error	57	1641.05	28.7904		
Total	59	2910.98			

Figure C.17: One Way Anova CLASP and DataDesk

C.5 One Way ANOVA

Input Data

Three categories of data were generated with **DataDesk** and then merged together to serve as input to CLASP. All three data sets were generated using normal distributions. with differing means but same variance, the number of observations in each case was 20.

In datadesk, "Source = Grp" means, "between groups", and "Source = Error" means "within groups." As seen from Figure C.17, the df, sum of squares, Mean square and the F-ratio are consistent.

$$\begin{array}{lll} \text{Category 1} & \mu = 45 & \sigma = 5 \\ \text{Category 2} & \mu = 50 & \sigma = 5 \\ \text{Category 3} & \mu = 55 & \sigma = 5 \end{array}$$

From data-set Anova2waydata
Row variable: Column 2
Column variable: Column 1
For dependent variable: Column 3

Source	df	SS	s^2	F
Column 1	2	665.666	332.833	255.698
Column 2	3	260.209	86.736	66.635
Column 2 × Column 1	6	1460.478	243.413	187.001
Within cells	12	15.620	1.302	
Total	23	2401.973		

Figure C.18: Two Way Anova CLASP

C.6 Two Way ANOVA

Input Data

The input to this file contains 3 columns, namely the row number, column number, and the dependent variable. Please see enclosed data file for CLASP. The incidence table given under the the results from Statview, gives an idea of the input data. From this incidence table we can see that each of the cells have the same number of observations, which is a requirement imposed by CLASP, as mentioned in the CLASP manual. Two-way ANOVA was tested on CLASP and Statview using the above input data. The results, i.e., df, sum of squares, mean square and the F-ratio, that were obtained, were consistent as far as the integral part was concerned. The first two decimal digits were also mostly consistent, except in one or two cases.

C.7 Summary Statistics

Input Data

Data was generated for the 3 distributions, namely mormal, binomial, and poisson using *Generate Random Numbers* on DataDesk. The summary statistics were calculated on the above 3 input data sets, using CLASP and Datadesk. The results i.e., number of cases, minimum, maximum, range, median, mean, variance, and standard deviation that were obtained, were consistent up to 2 decimal digits. The third decimal digit was also mostly consistent, except in one or two cases.

Normal $\mu = 50$ $\sigma = 5$

Binomial $n = 500$ $p = 0.1$

Poisson $\lambda = 50$

Normal Distribution

CLASP		DataDesk		
Number of cases:	100	NumNumeric	=	100
Minimum:	38.668	Minimum	=	38.668
Maximum:	59.849	Maximum	=	59.849
Range:	21.181	Range	=	21.181
Median:	49.850	Median	=	49.850
Mean:	49.613	Mean	=	49.613
Variance:	20.844	Variance	=	20.844
Standard Deviation:	4.566	Standard Deviation	=	4.5655

Binomial Distribution

CLASP		DataDesk		
Number of cases:	100	NumNumeric	=	100
Minimum:	31.000	Minimum	=	31.000
Maximum:	64.000	Maximum	=	64.000
Range:	33.000	Range	=	33.000
Median:	48.500	Median	=	48.500
Mean:	48.680	Mean	=	48.680
Variance:	49.795	Variance	=	48.796
Standard Deviation:	7.057	Standard Deviation	=	7.0566

Poisson Distribution

CLASP		DataDesk		
Number of cases:	100	NumNumeric	=	100
Minimum:	33.000	Minimum	=	33.000
Maximum:	69.000	Maximum	=	69.000
Range:	36.000	Range	=	36.000
Median:	49.000	Median	=	49.000
Mean:	49.100	Mean	=	49.100
Variance:	61.343	Variance	=	61.343
Standard Deviation:	7.832	Standard Deviation	=	7.8322

Figure C.19: Summary Statistics

C.8 χ^2

C.8.1 χ^2 2×2

Input Data	
50	50
70	30

CLASP Results

The hypothesis of independence can be rejected at the .0100 level ($\chi^2 = 7.521$).

Statview Results

The χ^2 value obtained in CLASP is the same as the χ^2 with continuity correction value in Statview. The null hypothesis in CLASP assumes that the expected frequencies for each of the conditions will be the same. But if the equal frequencies condition is not assumed and the expected frequencies are calculated based on the independence criteria, the Total Chi-Square value of Statview results. Further, the significance level obtained in CLASP is .0100, whereas in Statview it is .0061.

C.8.2 χ^2 $m \times n$

CLASP Results

The input file to CLASP contains 3 columns, namely the row number, column number, and the observed frequency. The row number and the column number define the different categories of the 2 attributes that are being tested for independence.
For Row: method
Column: failure
The hypothesis of independence can be rejected at the $p < 0.513$ level $\chi^2 = 1.333$

The observed frequencies:

2 1
2 1
2 4

The significance level and the value of χ^2 tally between CLASP and Statview.

C.9 T test

C.9.1 T-Test Matched Pairs

The test was performed on two different sample sizes

1. For large n i.e., $n = 100$
2. For small n i.e., $n = 20$

Further, for each of the above sample sizes, three data sets, were used

CLASP Results

Data Set 1 : (Normal Binomial)		
For Large N: Norm and Large N: Bino.		
The standard error of the difference	=	0.842
With degrees of freedom	=	99
The value of t	=	1.107
Which is significant at	$p <$	0.2151

DataDesk Results

Data Set 1 : (Normal Binomial)		
Norm:1 - Bino:1: Test $H_0 : \mu = 0$ vs $H_a : \mu \neq 0$		
Sample mean	=	0.93253
with		99 d.f.
t-statistic	=	1.107
Fail to reject H_0 at	$\alpha =$	0.05
Prob	<=	0.4321

Figure C.20: T test Matched Pairs Large N 1

1. Sample pair (normal binomial) consisting of 2 independent samples
2. Sample pair (normal poisson) consisting of 2 independent samples
3. Sample pair (binomial poisson) consisting of 2 independent samples

The t-test matched pairs was tested on the above data, using CLASP and Datadesk. The results are shown in Figures C.20 – C.25. The value of t, and the degrees of freedom obtained from both tallied. However, the p value at which t is significant does not tally.

C.9.2 T-Test Pooled Variance

The test was performed on two different sample sizes

1. For large n i.e., $n = 100$
2. For small n i.e., $n = 20$

Further, for each of the above sample sizes, three data sets, were used

1. Sample pair (normal binomial) consisting of 2 samples that are not independent.
2. Sample pair (normal poisson) consisting of 2 samples that are not independent.
3. Sample pair (binomial poisson) consisting of 2 samples that are not independent.

The t-test with pooled variance was tested on the above data, using CLASP and Datadesk. The results are shown in Figures C.26–C.31. The value of t, and the degrees of freedom obtained from both tallied. However, the p value at which t is significant did not tally.

CLASP Results

Data Set 2 : (Normal Poisson)

For Large N: Norm and Large N: Poiss.		
The standard error of the difference	=	0.862
With degrees of freedom	=	99
The value of t	=	0.595
Which is significant at	$p <$	0.3330

DataDesk Results

Data Set 2 : (Normal Poisson)

Norm:1 - Poiss:1: Test $H_0 : \mu = 0$ vs $H_a : \mu \neq 0$		
Sample mean	=	0.51253
with		99 d.f.
t-statistic	=	0.595
Fail to reject H_0 at	$\alpha =$	0.05
Prob	<=	0.5535

Figure C.21: T test Matched Pairs Large N 2

CLASP Results

Data Set 3 : (Binomial Poisson)

For Large N: Bino and Large N: Poiss.		
The standard error of the difference	=	1.070
With degrees of freedom	=	99
The value of t	=	−0.393
Which is significant at	$p <$	0.3682

DataDesk Results

Data Set 3 : (Binomial Poisson)

Bino:1 - Poiss:1: Test $H_0 : \mu = 0$ vs $H_a : \mu \neq 0$		
Sample mean	=	−0.42000
with		99 d.f.
t-statistic	=	−0.393
Fail to reject H_0 at	$\alpha =$	0.05
Prob	<=	0.6954

Figure C.22: T test Matched Pairs Large N 3

CLASP Results

Data Set 1 : (Normal Bimomial)

For T Test Nsmall: Norm-s and T Test Nsmall: Bino-s.		
The standard error of the difference	=	1.312
With degrees of freedom	=	19
The value of t	=	1.810
Which is significant at	$p <$	0.0802

DataDesk Results

Data Set 1 : (Normal Bimomial)

Norm-s - Bino-s: Test $H_0 : \mu = 0$ vs $H_a : \mu \neq 0$		
Sample mean	=	2.3736
with		19 d.f.
t-statistic	=	1.809
Fail to reject H_0 at	$\alpha =$	0.05
Prob	<=	0.0861

Figure C.23: T test Matched Pairs Small N 1

CLASP Results

Data Set 2 : (Normal Poisson)

For T Test Nsmall: Norm-s and T Test Nsmall: Poiss-s.		
The standard error of the difference	= 1.846	
With degrees of freedom	=	19
The value of t	=	0.500
Which is significant at	$p <$	0.3454

DataDesk Results

Data Set 2 : (Normal Poisson)

Norm-s - Poiss-s: Test $H_0 : \mu = 0$ vs $H_a : \mu \neq 0$		
Sample mean	=	0.92367
with		19 d.f.
t-statistic	=	0.500
Fail to reject H_0 at	$\alpha =$	0.049
Prob	<=	0.6224

Figure C.24: T test Matched Pairs Small N 2

CLASP Results

Data Set 3 : (Binomial Poisson)

For T Test Nsmall: Bino-s and T Test Nsmall: Poiss-s.		
The standard error of the difference	=	2.189
With degrees of freedom	=	19
The value of t	=	−0.662
Which is significant at	$p <$	0.3133

DataDesk Results

Data Set 3 : (Binomial Poisson)

Bino-s - Poiss-s: Test $H_0 : \mu = 0$ vs $H_a : \mu \neq 0$		
Sample mean	=	−1.4499
with		19 d.f.
t-statistic	=	−0.662
Fail to reject H_0 at	$\alpha =$	0.049
Prob	<=	0.5156

Figure C.25: T test Matched Pairs Small N 3

CLASP Results

Data Set 1 : (Normal Binomial)

For Large N: Norm and Large N: Bino.		
The standard error of the difference	=	0.840
With degrees of freedom	=	198
The value of t	=	1.109
Which is significant at	$p <$	0.2151

DataDesk Results

Data Set 1 : (Normal Binomial)

Test $H_0 : \mu$(Norm:1) − μ(Bino:1) = 0 vs. $H_a : \mu$(Norm:1) − μ(Bino:1) ≠ 0		
Sample mean(Norm:1)	=	49.613
Sample mean(Bino:1)	=	48.680
with		198 d.f.
t-statistic	=	1.110
Fail to reject H_0 at	$\alpha =$	0.05

Figure C.26: T test Pooled Variance Large N 1

CLASP Results

Data Set 2 : (Normal Poisson)

For Large N: Norm and Large N: Poiss.		
The standard error of the difference	=	0.907
With degrees of freedom	=	198
The value of t	=	0.565
Which is significant at	$p <$	0.3394

DataDesk Results

Data Set 2 : (Normal Poisson)

Test $H_0 : \mu(\text{Norm:1}) - \mu(\text{Poiss:1}) = 0$ vs. $H_a : \mu(\text{Norm:1}) - \mu(\text{Poiss:1}) \neq 0$

Sample mean(Norm:1)	=	49.613
Sample mean(Poiss:1)	=	49.100
with		198 d.f.
t-statistic	=	0.565
Fail to reject H_0 at	$\alpha =$	0.05

Figure C.27: T test Pooled Variance Large N 2

CLASP Results

Data Set 3 : (Binomial Poisson)

For Large N: Bino and Large N: Poiss.		
The standard error of the difference	=	1.054
With degrees of freedom	=	198
The value of t	=	−0.398
Which is significant at	$p <$	0.3679

DataDesk Results

Data Set 3 : (Binomial Poisson)

Test $H_0 : \mu(\text{Bino:1}) - \mu(\text{Poiss:1}) = 0$ vs. $H_a : \mu(\text{Bino:1}) - \mu(\text{Poiss:1}) \neq 0$

Sample mean(Bino:1)	=	48.680
Sample mean(Poiss:1)	=	49.100
with		198 d.f.
t-statistic	=	−0.398
Fail to reject H_0 at	$\alpha =$	0.05

Figure C.28: T test Pooled Variance Large N 3

CLASP Results

Data Set 1 : (Normal Binomial)

For Small N: Norm-s and Small N: Bino-s.		
The standard error of the difference	=	1.639
With degrees of freedom	=	38
The value of t	=	1.448
Which is significant at	$p <$	0.1390

DataDesk Results

Data Set 1 : (Normal Binomial)

Test $H_0 : \mu$(Norm-s:1) $- \mu$(Bino-s:1) $= 0$ vs. $H_a : \mu$(Norm-s:1) $- \mu$(Bino-s:1) $\neq 0$		
Sample mean(Norm-s)	=	51.423
Sample mean(Bino-s)	=	49.049
with		38 d.f.
t-statistic	=	1.447
Fail to reject H_0 at	$\alpha =$	0.04

Figure C.29: T test Matched Pairs Small N 1

CLASP Results

Data Set 2 : (Normal Poisson)

For Small N: Norm-s and Small N: Poiss-s.		
The standard error of the difference	=	1.730
With degrees of freedom	=	38
The value of t	=	0.534
Which is significant at	$p <$	0.3426

DataDesk Results

Data Set 2 : (Normal Poisson)

Test $H_0 : \mu$(Norm-s:1) $- \mu$(Poiss-s:1) $= 0$ vs. $H_a : \mu$(Norm-s:1) $- \mu$(Poiss-s:1) $\neq 0$		
Sample mean(Norm-s)	=	51.423
Sample mean(Poiss-s)	=	50.500
with		38 d.f.
t-statistic	=	0.533
Fail to reject H_0 at	$\alpha =$	0.04

Figure C.30: T test Matched Pairs Small N 2

CLASP Results

Data Set 3 : (Binomial Poisson)		
For Small N: Bino-s and Small N: Poiss-s.		
The standard error of the difference	=	1.983
With degrees of freedom	=	38
The value of t	=	−0.731
Which is significant at	$p <$	0.3018

DataDesk Results

Data Set 3 : (Binomial Poisson)		
Test $H_0 : \mu$(Bino-s:1) $- \mu$(Poiss-s:1) $= 0$ vs. $H_a : \mu$(Bino-s:1) $- \mu$(Poiss-s:1) $\neq 0$		
Sample mean(Bino-s)	=	49.049
Sample mean(Poiss-s)	=	50.500
with		38 d.f.
t-statistic	=	−0.731
Fail to reject H_0 at	$\alpha =$	0.04

Figure C.31: T test Matched Pairs Small N 3

Index