The 'rgr' package and functions

Over 100 'rg' functions were written between 1994 and 2007 at the Geological Survey of Canada (GSC) for the S-Plus proprietary statistical software to support exploration and applied geochemical survey and research activities. Most of these function scripts were written from 'scratch', however, others were based on scripts shared within the S user community on S-News. The S language was developed at Bell Laboratories (now Lucent Technologies) by John Chambers and his colleagues, and later commercialized by a third party for distribution as S-Plus.

The 'rgr' functions are a subset of the 'rg' functions that run under the R system, a version of the S language, which is a free software environment for statistical computing and graphics. R is widely used in academia and other institutions, see http://www.r-project.org/, and the development of R and the contribution of packages continues by the development team and users, respectively. R may be downloaded from any of the CRAN (Comprehensive R Archival Network) sites listed at http://www.r-project.org/. Version 'rgr_1.1.9' comprises 103 functions, a combination of original 'rg' and newly developed functions. Most of these are exploratory data analysis (EDA) and data inspection tools, many of which rely on graphics as a way to communicate the structure of the data sets under investigation and interpretation. All current development is carried out in the R environment. The functions in include univariate and multivariate procedures to assist in visualizing data structure in both geochemical and geographic space, and in identifying multivariate outliers. Other functions support applied geochemical QA/QC and threshold selection. The multivariate procedures include Principal Component Analysis (PCA), 2-d projections of multivariate data, and Mahalanobis distance based procedures. The bivariate and multivariate tools include provision for the computation of log-ratio transformations for the processing of closed compositional, geochemical, data.

The 'rgr' Library was first built in 2007 at the Geological Survey of Canada, Ottawa, by Yiwen Chen working in collaboration with the author. In the past the GSC has undertaken specific geochemical compilations of National Geochemical Reconnaissance and other geochemical survey data sets held by the GSC for Other Government Departments (OGDs). The early compilations and reports were prepared using the original S-Plus 'rg' functions, latterly using 'rgr'. These are available as GSC Open Files from the Natural Resources Canada Geoscience Data Repository, see http://gdr.nrcan.gc.ca/geochem/index_e.php. One of the OGD clients requested that the 'rg' functions used in report preparation be made available to them for use by their staff and contractors. Additionally, a growing number of GSC staff were asking for access to the same functions. Not wishing to tie future users to a proprietary package it was decided to 'translate' the 'rg' functions into R. Thus the 'rgr' Library was prepared, documented and released through CRAN for use by GSC staff, OGDs, agencies and individuals wishing to use the Applied Geochemistry Section's graphical and other procedures.

The 'rgr' functions may be used to process applied geochemical data to generate statistical summaries, both numeric and graphical, in support of the estimation of the boundaries of ambient and natural geochemical background. Implicitly this includes the setting of threshold or action levels that may trigger further field activities to determine the causes of any outliers. That is, whether observations with above threshold or action levels, are due to natural or

anthropogenic causes. The 'rgr' functions have been developed for studying geochemical data in the 'geochemical-space'. If observation site coordinates are available, four functions are available to display simple spatial plots, 'maps', for data inspection. However, these do not replace the use of a Graphical Information System, GIS, for spatial data display and analysis. Provision is made in several 'rgr' functions to export results as '.csv' files for importing into GIS, or other, software. A further spatial function displays a concentration-area plot that assists in determining if multi-fractal patterns are present in the data. These can be used to identify boundaries between data populations related to different spatial - fractal - processes; e.g., background and anomalous.

The following notes are provided for those as yet unfamiliar with the use of R.

Although the examples in the help files and manual for the 'rgr' package use TRUE and FALSE, the capital letters T and F, respectively, may be used when running the functions.

Quotes, " ", are used to enclose character strings that will be printed or displayed. To obtain a Greek µ, use \265, thus to display (µg/kg) use "... (\265g/kg) ...", if a tab is required use \t, and \n forces a new line. The available codes for special characters such as the Greek µ may be displayed with function 'display.ascii.o'.

Where justification of text is an option, adj = 0 results in left justification, adj = 1 in right justification, and adj = 0.5 in centring. Defaults are always provided.

NA is an explicit way that the S language, and R, has of conveying the fact that there is no information. A blank numeric field, i.e. ', ,'(note the space between the commas) in a table entered into R by the 'read.table' or 'read.csv' commands is converted to a NA, for an actual value of zero a zero has to be explicitly entered. In some geochemical data files blanks have been converted to zeros by other software packages, in others they are set to a coded value, e.g. -9999. Tools are available in functions 'ltdl.fix.df' and 'ltdl.fix' to set these zeros or coded values to NAs if that is appropriate.

Many computational tasks cannot accept NAs, therefore software, function 'remove.na', is used internally within the 'rgr' functions to remove NAs from data when required.

It is common practice to set applied and exploration geochemical results less than the detection or quantification limit (<dl) to the negative value of the detection limit. Tools, functions 'ltdl.fix.df' and 'ltdl.fix' for data frames and vectors, respectively, are provided to convert these negative values to half the positive value of the detection limit. This is essential if logarithmic scaling is to be used in plots, or calculations are to be undertaken in logarithms. Fortunately, with the introduction of ever more sensitive methods of analysis, such as Inductively Coupled Plasma Mass Spectrometry (ICP-MS), data sets with large numbers of below detection limit analyses are becoming relatively rare. For data sets with a large proportion of below detection limit values alternate data processing procedures should be used, e.g., Kaplan-Meier methods, see Denis Helsel's 2005 (also the 2012 2nd edition) book 'Non Detects and Data Analysis: Statistics for Censored

Environmental Data'.

If logarithmic scaling is required for plots or computations ensure that the parameter log = T and, if required, logx = T. In some functions where two variables are plotted against each other, log = "x", log = "y" and log = "xy", control the scaling of the axes.

A common construct for storing data in S and R is the data frame, this includes not only the data but also the variable names (columns) and the observation identifiers (rows). The latter commonly known as the sample numbers or IDs. In this there is a difference between natural scientists and statisticians, to a natural scientist a sample is an individual 'something' that is collected, described and measured, whereas to a statistician the sample is the whole collection of individual 'somethings' and has some size N. The data for any one variable or measurement is a column vector. The data in a data frame may be made easily accessible by attaching the data frame with 'attach(dfname)'. The attached data frame may be removed by 'detach(dfname)'. The function 'df.test(dfname)' may be run to see if a data frame is attached or present in the R work space and identify the names of the variables it contains.

It is the author's usual practice to import data with the 'read.csv' command from a comma separated values, '.csv', file, with the first column containing sample IDs. However, the first record of the file containing the column (variable) names is modified by removing the first column descriptor for the ID. This results in the IDs being used as row identifiers in the resulting data frame.

The IDs of the observations and the names of the variables in a data frame, or matrix, may be displayed with the command 'dimnames(dfname)', or selectively the IDs by 'dimnames(dfname)[[1]]' and the variable names by 'dimnames(dfname)[[2]]'. If is required to change any variable names this can be achieved by, for example, 'dimnames(dfname)[[2]][6] <- "Cu"', where the name of 6th item in the variable list is changed to Cu, the quotes, " ", being necessary to identify the information as text.

The construct 'deparse(substitute(x))' is used to generate a default variable name label from the column variable name, e.g., Cu (a column variable name). Alternately, a user defined text string like "Cu (mg/kg) in <2 mm C-horizon soil" may be provided. The contents of the variable name label are variously defined as xlab, xname, with a similar parameter name for y or z variables, where required.

A construct useful in the execution time selection of a subset of the values for a variable is conditioning. While Cu leads to the processing of the entire vector of data for the column variable Cu, Cu[Cu<200] would result in the processing of only those data where the Cu value was <200. As an example, length(Cu[Cu<200]) would display the number of analyses with Cu values less than 200. Similarly, Cu[Cu>10 & Cu<200] would result in the processing of only those data with values between, exclusively, 10 and 200. The condition may be based on the values of any variable available in the data frame, thus Cu[Zn>200] would result in only those Cu values where the Zn value exceeded 200 being processed. Similarly, for a 'factor' (text string variable), Cu[PM =="Till"] (note the

double = signs) results in only those Cu values where PM (the soil parent material) was recorded as Till being processed. For generating more permanent subsets from a data frame or matrix the R function ‘subset’ or the rgr package function ‘gx.subset’ may be used.

In some cases it is required to split the data for a variable, a column vector, into subsets based on the value of some classificatory variable, factor, which appears as a column variable. For example, split(Cu, GSG) would split the data for the variable Cu into subsets on the basis of the values of GSG (Great Soil Group). The values of the criterion may be either character strings or integer numbers, there will be as many subsets as there are unique values of the criterion (factor). This technique may be used with functions ‘bwplots’ and ‘tbplots’. The value of the criterion also may be computed, for example, Distance%/%10 generates a truncated (integer) value of the Distance from a fixed point divided by 10. Thus all Distances between 0 and 9.99... have a value of 0, those from 10 to 19.99... have a value of 1, and so on. Thus if Distance is the distance from a point source of contaminants, e.g., a smelter stack, a Tukey boxplot display can be generated where the individual plots graphically summarize the data in 10 km groupings from the source.

Where options exist for the colour infill of polygons the default is grey, ‘colr = 8'. The following are the available default R colours: 1 = black; 2 = red; 3 = green; 4 = dark blue; 5= light blue; 6 = purple; 7 = yellow; and 8 = grey. Setting colr = 0 results in no infill. To display the actual colours use function ‘display.lty()’ that also displays line styles.

Most users find it convenient to use a ‘first’ function. Such a function can be used to load the rgr package for use along with other required R libraries, e.g., ‘fastICA’, set certain defaults to the user’s preferences, and set the R Working Directory to one appropriate for the data under investigation. In earlier versions of R the ‘MASS’ library had to be explicitly made available, currently it comes bundled with the standard R installation. ‘MASS’ and ‘fastICA’ are ‘lazy loaded’ by loading ‘rgr’, they just have to be in the R library folder in the user’s PC. The following is an example:

```
“first” <-
function (wd = NULL)
{
if(is.null(wd)) wd = "D:\\R\\Project 3\\WD"
setwd(wd)
library(rgr)
par(pty = "s", pch = 3)
cat("Default plot shape set to 'square' and plot symbol to 'plus'\n")
cat("Working directory set to:", wd, "\n")
options(warn = -1)
cat("R options set to warn = -1 to suppress unwanted graphics related messages from\n
rgr multi-panel functions and functions calling eqscplot in Library MASS\n")
}
```

Please note that ‘rgr_1.1.9’ has been changed to honour the licensing of package akima for not-for-profit use only so ‘rgr’ can retain its GPL-2 licensing. Now package ‘rgr’ only suggests akima, rather than depends, in its formal DESCRIPTION. As a result, non-for-profit users, e.g., users in governmental and academic institutions, need to explicitly make package akima available to be able to use function caplot. This may be done by adding the line library(akima) after library(rgr) in their ‘first’ function.

The above function assumes that a folder D:\R\Project 3\WD has been set up outside the Program Files for R as a location where data and output files are to be stored. In this example Project 3, or some other appropriate name, is a subdirectory where Project 3 data and files are kept. In general it is good practice to store the data and project specific files in a different place than the software. If a ‘data’ drive, e.g., D:\, is not available a subdirectory in My Documents can be used. The above ‘first’ function reflects the preference of the author for ‘square’ xy-plots and the use of a ‘+’ sign as the plotting symbol. Another plotting symbol can be chosen from the available plotting marks, see ‘display.marks’, and if the user consistently wants rectangular plots that make maximum use of the display space, change pty = “s” to pty = “m”.

To complete the set-up for a project an R icon should be placed on the desk top and edited so that R uses the defined Working Directory for storing the R files: .Rdata and .Rhistory. To place an extra R icon on the desktop for a particular project, go to the C:\Program Files\R\bin subdirectory and make a shortcut to R.exe and then drag it to the desktop, where it can be renamed appropriately, e.g., Project 3. In this manner different R sessions for different projects or data investigations may be set up, with the result that only relevant files are accessible and the R workspace is less cluttered. This is done by right clicking on the R icon placed on the desktop and selecting Properties and editing the ‘Start in:’ field to D:\R\Project3\WD and clicking on Apply.

If the user has set up a folder D:\R\Project3\WD for data and files for Project 3, then clicking on the ‘Project 3' R icon will start the session in the correct WD subdirectory. If a different Working Directory is required entering ‘first(“D:\\R\\Project 4\\WD") at the > in the R session will result in the rgr package being made available, the Working Directory being set to D:\R\Project 4\WD, and any other defaults the user wishes to set being implemented. Note two things: 1) the subdirectory D:\R\Project 4\WD must have been created; and 2) the use of \\ to cause the correct backslash for the file name in the execution of the ‘first’ function.

The last instruction in the ‘first’ function is ‘options(warn = -1)’. The use of multi-panel displays in some ‘rgr’ functions and the use of function ‘eqscplot’ from the ‘MASS’ Library in others causes warning messages to be displayed concerning certain graphics parameters. These are not relevant to the user and the ‘options(warn = -1)’ statement leads to their suppression.

In practice it is important to appropriately handle any -ve values due to the presence of <dl data and any zeros or coded values indicating missing data prior to undertaking any plotting or computations. The easiest way to achieve this is by use of the ‘ltdl.fix.df’

function. Thus, if the data are in a data frame 'dfname' the command 'dfname.fixed <- ltdl.fix.df(dfname)' is executed. The newly created object 'dfname.fixed' is attached, 'attach(dfname.fixed)' so that its column vectors are directly accessible. Any resulting NAs are handled appropriately in each 'rgr' function using the 'remove.na' function. An exception to this latter statement is when executing log-ratio transformations in the calls to multivariate EDA 'rgr' functions, where required. In those instances the functions na.omit or where.na should be used to clear data matrices of rows containing NAs.
The following list briefly describes the functions (94) available to the user, functions that are only called internally (9) are marked with an asterisk (*), together with the test data sets (15) used in the examples and provided for investigation by the user:

ad.plot1	Plot analytical duplicate results
ad.plot2	Plot analytical duplicate results, alternate input
ad.test	Data Frame of test data for function ad.plot2
alr	Undertake an Additive Log-Ratio transformation
anova1	Duplicate Sample Analysis of Variance (ANOVA)
anova2	Duplicate Sample Analysis of Variance (ANOVA), alternate input
bwplots	Plot Vertical Box-and-Whisker Plots
bwplots.by.var	Plot Vertical Box-and-Whisker Plots for Variables
bxplot	Plot a Horizontal Boxplot or Box-and-Whisker Plot
caplot	Prepare a Concentration-Area (C-A) Plot
cat2list *	Divides Data into Subsets by Factor
clr	Undertake a Centred Log-Ratio Transformation
cnpplt	Displays a Cumulative Normal Percentage Probability (CPP) Plot
crm.plot	Plot Control Reference Material (CRM) analyses
crm.test	Data Frame of test data for function crm.plot
cutter *	Function to Identify into which Interval a Value Falls
df.test	Check for the Existence of a Data Frame
display.ascii.o	Display the Windows Latin 1 Font Octal Table
display.lty	Display Available Line Styles and Colour Codes
display.marks	Display the Available Plotting Symbols
display.rainbow	Display the Colours of the Rainbow(36) Pallette
expit	Undertake an Inverse-Logit Transformation
fences	Generate and Display Fence Values
fences.summary	Generate and Save Fence Values for Data Subsets
fix.test	Test Data Frame for Function ltdl.fix.df - IDs in dimnames(fix.test)[[1]]
fix.test.asis	Test Data Frame for Function ltdl.fix.df - explicit IDs
framework.stats *	Compile Framework/Subset Summary Statistics
framework.summary	Generate and Save Framework/Subset Summary Statistics
gx.2dproj	Generate a 2-D Projection of P-Space Data
gx.2dproj.plot	Display the 2-D Projection computed by gx.2dproj
gx.add.chisq *	Adds and Displays Chi-square 'Fences' to Plots of Mahalanobis Distances
gx.adjr2	Compute Adjusted R^2 values for Multilinear Regression Models
gx.cnpplts	Plot up to nine CPP in a single Display
gx.cnpplts.setup	Define Symbology and Colours for use in gx.cnpplts
gx.ecdf	Plot an Emprical Cumulative Distribution Function (ECDF)

gx.fractile	Estimate the Fractile for a specified Quantile of a Distribution
gx.hist	Plot a Histogram
gx.hypergeom	Estimate the Probability of Anomaly Location is Informative
gx.ks.test	Plot ECDFs of two Distributions with a Kolmogorov-Smirnov Test
gx.lm.vif	Compute Variance Inflation Factors to Assess Collinearity in Linear Models
gx.md.display	Displays and/or Saves Mahalanobis Distances and Associated Information
gx.md.gait	Undertake a Graphical Adaptive Interactive Trimming (GAIT) exercise with Multivariate Data via Mahalanobis Distance Estimation
gx.md.gait.closed	Undertake a Graphical Adaptive Interactive Trimming (GAIT) exercise with Closed, Compositional, Multivariate Data via Mahalanobis Distance Estimation
gx.md.plot	Display Chi-square plots of Mahalanobis Distances
gx.md.plt0 *	'Engine' for preparing Chi-square plots
gx.md.print	Displays and/or Saves Mahalanobis Distances
gx.mva	Estimate the Covariance Matrix, Means, R-Q Principal Components and Mahalanobis Distances for a Matrix by Classical Methods
gx.mva.closed	Estimate the Covariance Matrix, Means, R-Q Principal Components and Mahalanobis Distances for a Matrix by Classical Methods specifically for Closed Compositional, Geochemical, Data sets
gx.mvalloc	Undertake a Multivariate Data Classification on the basis of Mahalanobis Distances for Reference Groups, and Simultaneously Identify Outliers
gx.mvalloc.closed	Undertake a Multivariate Data Classification on the basis of Mahalanobis Distances for Reference Groups of Closed, Compositional, Data, and Simultaneously Identify Outliers
gx.mvalloc.print	Displays and/or Saves the Results of a Multivariate Allocation exercise
gx.pairs4parts	Display a Graphical Matrix of log10 XY-Plots and Boxplots of the ilr Transforms annotated with their corresponding Robust ilr Stabilities
gx.pearson	Compute and Display Pearson Correlation Coefficients and Probabilities
gx.plot2parts	Display Four Plots to Support the Inspection of Two Parts of a Compositional Data Set
gx.quantile	Estimate the Quantile for a specified Fractile of a Distribution
gx.rma	Estimate the Reduced Major Axis Coefficients and Test for (0,1)
gx.robmva	Estimate the Covariance Matrix, Means, R-Q Principal Components and Mahalanobis Distances for a Matrix by Robust Methods (MCD, MVE, user's weights)
gx.robmva.closed	Compute R-Q Principal Components and Mahalanobis Distances for a Matrix by Robust Methods (MCD, MVE, user's weights) specifically for Closed Compositional, Geochemical, Data sets
gx.rotate	Undertake a Kaiser Varimax Rotation of Principal Components
gx.rqpca.loadplot	Graphical Display of PC Loadings
gx.rqpca.plot	Displays Bi-Plots arising from an R-Q Principal Component Analysis
gx.rqpca.print	Displays tables of PC Loadings and Scores
gx.rqpca.screeplot	Displays a Scree Plot for a Principal Components Analysis
gx.runs	Carry out a Wald-Wolfowitz, Runs, Test
gx.scores	Function to Compute Scores on the Basis of Threshold Estimates

gx.sort	Sort a Dataframe or Matrix on a single variable and Display
gx.sort.df	Sort a Dataframe on multiple variables and Display
gx.spearman	Compute and Display Spearman Correlation Coefficients and Probabilities
gx.sm	Compute and Display Robust ilr Stabilities and log-ratio Medians
gx.stats	Compute Summary Statistics and optionally Display
gx.subset	Extract a Subset of Rows from a Data Frame
gx.summary *	Compile Summary Statistics for other displays
gx.summary1	Display a Concise Single Line Summary Statistics Report
gx.summary2	Display a ten-line Summary Statistics Report
gx.summary.groups	Display gx.summary1 style Reports for Factors in a Dataframe
gx.summary.mat	Display gx.summary1 style Reports for Selected Columns of a Dataframe
gx.triples.aov	Undertake a Staggered 3-Level Design ANOVA for Sampling and Analytical Variability and Estimate Variance Components
gx.triple.fgx	Undertake ANOVAs to estimate the Regional Representivity of Triples
gx.vm	Compute and Display an Aitchison Variability Matrix
ilr	Undertake an Isometric Log-Ratio Transformation
ilr.stab	Estimate the Robust ilr Stability for Two Parts of a Compositional Data Set
inset	An EDA Graphical and Statistical Summary
inset.exporter	Saves an EDA Graphical and Statistical Summary
kola.c	Kola Project C-horizon Soil Data Frame
kola.o	Kola Project O-horizon Soil Data Frame
logit	Undertake a Logit Transformation
ltdl.fix	Replace Negative Values Representing Less Than Detects for a Vector
ltdl.fix.df	Replace Negative Values Representing Less Than Detects for a Dataframe
map.eda7	Display a Symbol Map of Numeric Data Based on the Tukey Boxplot
map.eda8	Display a Symbol Map of Numeric Data Based on their Percentiles
map.tags	Display a Map of Posted Values
map.z	Display a Map of Numeric Data using Proportional Symbols
ms.data1	Measurement Variability Test Data Frame
ms.data2	Measurement Variability Test Data Frame
ms.data3	Measurement Variability Test Data Frame
ogrady	O'Grady Pluton, NWT, Granitoid Test Data Frame
ogrady.mat2open	A matrix of selected major and minor element analyses from ogrady all expressed in mg/kg (ppm) for investigation of closure effects
orthonorm *	Computes Orthonormal Basis Matrices for ilr to clr Back-transformations
remove.na	Remove and Count NAs
rng	Undertake Range Transformations on the Columns of a Matrix
shape	An EDA Graphical Summary
sind	Howarth and Sinding-Larsen Test Data Frame
sind.mat2open	A matrix of the geochemical analyses from sind all expressed in mg/kg (ppm) for the demonstration and investigation of closure effects
syms *	Function to Compute the Diameters of Proportional Symbols
syms.pfunc	Function to Demonstrate the Effect of Different Values of p
tbplots	Plot Vertical Tukey Boxplots
tbplots.by.var	Plot Vertical Tukey Boxplots for Variables

thplot1	Display a Thompson-Howarth Plot of Duplicate Measurements
thplot2	Display a Thompson-Howarth Plot of Duplicate Measurements, alternate input
triples.test1	Test Data Frame for Function gx.triples.aov
triples.test2	Test Data Frame for Function gx.triples.fgx
var2fact *	Rearranges Data for Variables as Factors
where.na	Function to Identify Locations of NAs in a Vector or Matrix
wtd.sums	Computation of Weighted Sums
xyplot.eda7	Display a 'XY' Plot of Numeric Data Based on the Tukey Boxplot
xyplot.eda8	Display a 'XY' plot of Numeric Data Based on their Percentiles
xyplot.tags	Display a 'XY' Plot of Posted Values
xyplot.z	Display a 'XY' Plot of Numeric Data using Proportional Symbols

Full details are available through the on-line help files in the 'rgr' package.

Robert G. Garrett
Emeritus Scientist
Geological Survey of Canada
Earth Sciences Sector, Natural Resources Canada
Ottawa, Ontario K1A 0E8

E_mail: garrett@NRCan.gc.ca

March 18, 2013