Help File

This file serves as a basic introduction to the functions available in this software.

Value Types: These value types are used in this piece of software to represent data and information; they are the common argument types for the functions provided.

1) Floating-Point: An arithmetic value for representing real numbers such as positive or negative decimal or integer values.
2) Integer: A whole number without decimal points.
3) Boolean: A True or False Value.
4) Matrix: A rectangular array/grid of Floating-Point values.
5) Function: A basic equation consisting of the previous value types and appropriate operands.

Functions:

Function Name	**Summary**	**Argument Types**	**Return Value**
backSub()	Performs backwards substitution	1) backSub(Matrix, "NULL") - Assumes Matrix value is an nx(n+1) value where the last column is the known as the B matrix. 2) backSub(Matrix, Matrix) - Assumes the first Matrix value is square and the second Matrix value is the B matrix.	Returns a Matrix value of the solution to the backwards substitution.
bisection()	Performs the bisection method onto a given function, with starting point intervals	1) bisection(Function, Floating Point, Floating Point) - Assumes the Function of the first Floating Point value times the Function of the second Float Point value is negative; that is to say, the given interval requires a change of signs	Returns a Floating Point value of the estimated root value of the function
cov()	Calculates the covariance Matrix on a Matrix dataset	1) cov(Matrix) - A matrix value of any size	Returns a Matrix value of the covariance of the dataset
cor()	Calculates the correlation Matrix on a Matrix dataset	1) cor(Matrix) - A matrix value of any size	Returns a Matrix value of the correlation of the dataset
delCol()	Deletes a specified column index on a Matrix	1) delCol(Matrix, Integer) - The Integer value represents the column index from which will be deleted in the Matrix value	Does not return a value
delRow()	Deletes a specified row index on a Matrix	1) delRow(Matrix, Integer) - The Integer value represents the row index from which will be deleted in the Matrix value	Does not return a value

eigen()	Calculates eigen values and eigen vectors on a square symmetric Matrix	1) eigen(Matrix) - Assumes Matix value that is square and symmetric 2) eigen(Matrix)$values -Assumes Matrix value that is square and symmetric 3) eigen(Matrix)$vectors -Assumes Matrix value that is square and symmetric	1) The return value cannot be used inside a calculation as it returns two Matrix values: Values and Vectors 2) Returns a Matrix value of the eigen values of the parameter Matrix 3) Returns the Matrix value of the eigen vectors of the parameter Matrix
forwSub()	Performs forwards substitution	1) forwSub(Matrix, “NULL”) - Assumes Matrix value is an nx(n+1) value where the last column is the known as the B matrix. 2) forwSub(Matrix, Matrix) - Assumes the first Matrix value is square and the second Matrix value is the B matrix.	Returns a Matrix value of the solution to the forwards substitution.
gamma()	Calculates the gamma value given a positive Integer value	1) gamma(Integer) - Assumes the Integer value is positive	Returns a Floating Point value of the gamma function
getCol()	Returns the column from the column index of the parameter Matrix	1) getCol(Matrix, Integer) -The Integer value represents the column index from which will be returned from the Matrix value	Returns a Matrix value of the column index from the parameter Matrix
getRow()	Returns the row from the row index of the parameter Matrix	1) getRow(Matrix, Integer) -The Integer value represents the row index from which will be returned from the Matrix value	Returns a Matrix value of the row index from the parameter Matrix
houseHolder()	Performs a House Holder Transformation on the symmetric Matrix	1) houseHolder(Matrix) -Assumes Matrix value is symmetric	Returns a Matrix value of the House Holder transformation
integrate()	Performs an Adaptive Composite Simpsons single integration method onto a given Function with bounds (Note:) Requires the use of "x" as the variable for integration	1) integrate(Function, Floating Point, Floating Point) - The Floating Point values represent the given interval over which the Function value will be integrated over 2) integrate(Function, Floating Point, "infty") - Performs integration to infinity over of the Function value (Note:) Spell out "infty" without quotes when performing	Returns a Floating Point value of the estimated integral

isPosDef()	Returns whether or not the Matrix value is Positive Definite; that is, all its eigenvalues are positive	1) isPosDef(Matrix) -Matrix value	Returns a Boolean Value- True or False
isSquare()	Returns whether or not the Matrix value is Square	1) isSquare(Matrix) -Matrix value	Returns a Boolean Value- True or False
isSymmetric()	Returns whether or not the Matrix value is Symmetric	1) isSymmetric(Matrix) -Matrix value	Returns a Boolean Value- True or False
LU()	Performs LU decomposition onto the Matrix value	1) LU(Matrix) -Assumes Matrix value is symmetric 2) LU(Matrix)$L -Assumes Matrix value is symmetric 3) LU(Matrix)$U -Assumes Matrix value is symmetric	1) The return value cannot be used inside a calculation as it returns two Matrix values: The L and U matrices 2) Returns a Matrix value of the Lower Triangular Matrix from LU decomposition 3) Returns the Matrix value of the Lower Triangular Matrix from LU decomposition
mean()	Calcuates the mean of each column in the parameter Matrix and stores it in the repsective column index of a single row Matrix	1) mean(Matrix) -Matrix value	Returns a Matrix value
naiveGauss()	Performs Naïve Gaussian Elimination on the parametered Matrices	1) naiveGauss(Matrix, NULL) -Assumes the Matrix value is nx(n+1) 2) naiveGauss(Matrix, Matrix) -Assumes the first Matrix value is square and the second Matrix value is nx1	Returns the row reduced Naïve Gaussian form of the parameterized Matrices
newton()	Performs Newton Raphsmon's method onto a given function, initial value, and derivative	1) bisection(Function, "NULL" , Floating Point) - As of current, the user cannot enter the derivative function so please enter "NULL" without the bracktets; the Floating Point value is the initial point	Returns a Floating Point value of the estimated root value of the function
pchisq()	Calcuates the Chi-Square probability value of the given a quantile function and degrees of freedom	1) pchisq(Floating Point, Floating Point) - First Floating Point value is the quantile value and the second Floating Point value is the degrees of freedom	Returns a Floating Point value of the Chi-Square probability given a quantile value

pnorm()	Calcuates the Normal probability of the given quantile value, mean, and standard deviation	1) pnorm(Floating Point, Floating Point, Floating Point) - First Floating Point represents the quantile, second Floating Point value represents the mean value, and third Floating Point value represents the standard deviation	Returns a Floating Point value of the Normal probability up to the quantile value
qchisq()	Calcuates the Chi-Square quantile function of the given a probability value and degrees of freedom	1) qchisq(Floating Point, Floating Point) - Requires that the first Floating Point value is between 0 and 1, as it represents the probability, second Floating Point value is the degrees of freedom	Returns a Floating Point value of the Chi-Square quantile up to the probability value
qnorm()	Calcuates the Normal quantile function of the given a probability value, mean, and standard deviation	1) qnorm(Floating Point, Floating Point, Floating Point) - Requires that the first Floating Point value is between 0 and 1, as it represents the probability, second Floating Point value represents the mean value, and third Floating Point value represents the standard deviation	Returns a Floating Point value of the Normal quantile up to the probability value
QR()	Performs QR decomposition onto the Matrix value by the Gram Schmidt process	1) QR(Matrix) -Assumes Matrix value is symmetric 2) QR(Matrix)$Q -Assumes Matrix value is symmetric 3) QR(Matrix)$R -Assumes Matrix value is symmetric	1) The return value cannot be used inside a calculation as it returns two Matrix values: The Q and R matrices 2) Returns a Matrix value of the orthogonal Q Matrix from QR decomposition 3) Returns the Matrix value of the lower triangular R Matrix from QR decomposition
setRow()	At the given row index, switch out the row with the entered Matrix	1) setRow(Matrix, Matrix, Integer) - The first Matrix value is the matrix from which the Integer value will be used as an index to be replaced with the second Matrix value; assumes the second Matrix value is 1xn	Returns a Boolean Value- True or False
setCol()	At the given column index, switch out the column with the entered Matrix	1) setCol(Matrix, Matrix, Integer) - The first Matrix value is the matrix from which the Integer value will be used as an index to be replaced with the second Matrix value; assumes the second Matrix value is nx1	Returns a Boolean Value- True or False
t()	Performs the transpose of a Matrix value	1) t(Matrix) - Matrix value	Returns a Matrix value, the transpose of the parameter Matrix

zscore()	Calculates the zscores of every entry in a Matrix value data set by subtracting each column by its mean and dividing it by its standard deviation	1) zscore(Matrix) - Matrix value	Returns a Matrix value of equivalent values in zscore form