The LWP-Trends library

December 2019

Title LWP-Trends library Version 1901: ***LWPTrends_v1901.r***

Date 13 December 2019

Authors Ton Snelder and Caroline Fraser LWP Ltd, Christchurch NEW ZEALAND.

Contacts: ton@lwp.nz or caroline@lwp.nz

Description Tools for analysing water quality trends

Requires: packages: plyr, NADA and gam are installed. (optionally ggplot2)

Contents:

Reference:

Snelder,T. and Fraser, C. (2019) *"The LWP-Trends Library; v1901 December 2019",* LWP Ltd Report, p35

Disclaimer

Software downloaded from the landwaterpeople (LWP) website (or provided by direct communication with an LWP employee) is provided 'as is' without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of fitness for a purpose, or the warranty of non-infringement. Without limiting the foregoing, LWP makes no warranty that:

1. the software will meet your requirements
2. the software will be uninterrupted, timely, secure or error-free
3. the results that may be obtained from the use of the software will be effective, accurate or reliable
4. the quality of the software will meet your expectations
5. any errors in the software obtained from the LWP web site will be corrected.

Software and its documentation made available on the LWP website:

1. could include technical or other mistakes, inaccuracies, or typographical errors. LWP may make changes to the software or documentation made available on its website.
2. may be out of date, and LWP makes no commitment to update such materials.

LWP assumes no responsibility for errors or omissions in the software or documentation available from its website.

In no event shall LWP be liable to you or any third parties for any special, punitive, incidental, indirect or consequential damages of any kind, or any damages whatsoever, including, without limitation, those resulting from loss of use, data or profits, whether or not LWP has been advised of the possibility of such damages, and on any theory of liability, arising out of or in connection with the use of this software.

The use of the software downloaded through the LWP site is done at your own discretion and risk and with agreement that you will be solely responsible for any damage to your computer system or loss of data that results from such activities. No advice or information, whether oral or written, obtained by you from LWP or from the LWP website shall create any warranty for the software.

Release notes: version 1901

1. Functions: InspectData, SeasonalityTest, AdjustValues, SenSlope and SeasonalSenSlope have been updated to optionally include plots to be returned in a list as ggplot objects.
2. A bug in the implementation of the HiCensor=TRUE switch for the SenSlope functions has been corrected.
3. When performing the seasonality tests, the minimum season requirements filter (as applied at the start of the seasonal Kendal and seasonal Sen slope tests) is used, and will return NA values (with a description of the issue), if the data has insufficient seasonal data to perform a seasonal trend analysis.
4. The HiCensor switch can now either be logical, or a numeric value specified by the user.
5. An additional season type of Bimonthly has been added to the GetDateInfo function.
6. A warning message related to the interpolation to determine the Sen slope has been resolved.
7. Notes have been added to the documentation to describe the warning messages produced when censored values are used to determine the Sen slope.
8. For users of R >3.6.1 a new error message arises when using the am flow adjustment – this does not affect results (it appears to be a bug that has been requested to be resolved by the R creators) – a message has been added to indicate that it is safe to ignore this warning message.

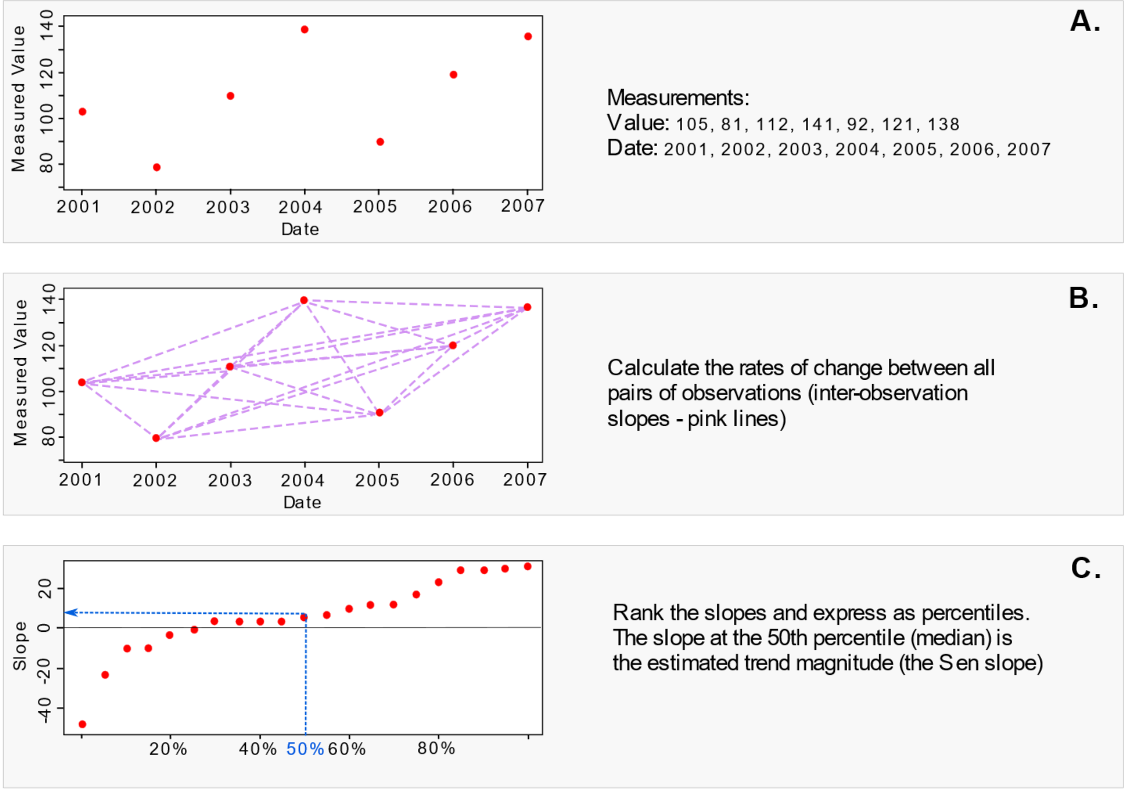
1 Introduction

This document describes how to use the functions in the LWP-Trends library to undertake water quality trend analysis in the R statistical computing environment. The functions were developed from scratch to provide up to date methods to evaluate trends.

2 Trend Analysis Method

The statistical analyses of trends involve the evaluation of (1) the magnitude of the trend and (2) the confidence in the trend direction.

Trend magnitude is characterised by the Sen slope estimator (SSE; Hirsch *et al.,* 1982). The SSE is the slope parameter of a non-parametric regression, which is calculated as the median of all possible inter-observation slopes (i.e., the difference in the measured observations divided by the time between sample dates; see Figure 1).



Measurements:
Value: 105, 81, 112, 141, 92, 121, 138
Date: 2001, 2002, 2003, 2004, 2005, 2006, 2007

Calculate the rates of change between all pairs of observations (inter-observation slopes - pink lines)

Rank the slopes and express as percentiles. The slope at the 50th percentile (median) is the estimated trend magnitude (the Sen slope)

Figure 1. Pictogram of the steps taken in the trend analysis to calculate the Sen slope, which is used to characterise trend magnitude in the time-series of data for each site × variable combination.

The seasonal version of the SSE is used in situations where there are significant (e.g., $p \leq 0.05$, as evaluated using a Kruskall Wallis test) differences in water quality measurements between ‘seasons’. Seasons are defined primarily by sampling intervals, which are commonly monthly or quarterly for water quality monitoring. The seasonal Sen slope estimator (SSSE) is the median of all inter-observation slopes within each season.

The Kendall test S and *p*-values are used by the LWP-Trends library to establish confidence in the trend direction. The Kendall test measures the rank correlation, which is a nonparametric correlation coefficient measuring the monotonic association between two variables, x and y. In water quality trend analysis, y is a sample of water quality measurements and x is the corresponding sample dates. Traditionally, the Kendall test is used to determine whether trends are statistically "significant" or "insignificant" (see Figure 2).

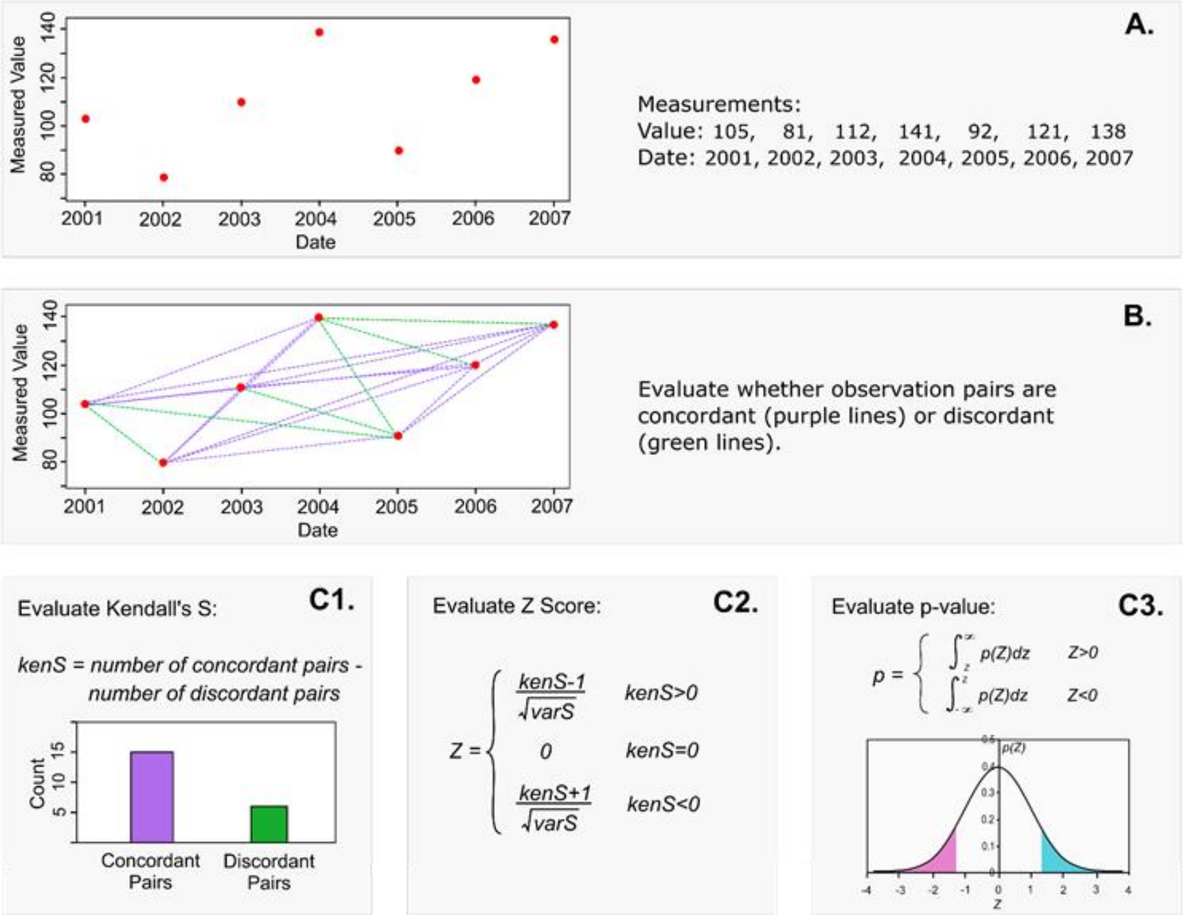


Figure 2.Pictogram of the steps taken in the trend analysis to calculate the Kendal statistic and its p-value, which is used to characterise the confidence in trend direction.

In the LWP-Trends library confidence in the direction of each trend is evaluated by interpreting the Kendall *p*-value as a probability that the trend was decreasing as follows:

$$P(S < 0) = 1 - 0.5 \ \times pvalue$$

$$P(S > 0) = 0.5 \ \times pvalue,$$

where $pvalue$ is the *p*-value returned by Kendall test (either seasonal or non-seasonal), *S* is the S statistic returned by Kendall test (either seasonal or non-seasonal) and *P* is the probability that the trend was decreasing. The trend direction is interpreted as decreasing when P > 0.5 and increasing when P < 0.5. Note that if data are seasonal (e.g., Kruskall Wallis p ≤ 0.05), a seasonal version of the Kendall test is used to evaluate the $pvalue$ and P.

The trend direction is established with a 95% level of confidence if the probability associated with S < 0 (i.e., a decreasing trend) is ≥ 95%, or the probability associated with S > 0 (i.e., an increasing trend) s ≤ 5%. In both, these cases the trend is categorised as 'established with confidence' and when the probability the trend is decreasing is between the 90% confidence limits (i.e., is ≥5% and ≤95%), the trend is categorised as being ‘indeterminant’.

2.1 Handling Censored Values

Censored values are those above or below a detection limit (e.g., >2.5 or <0.001). Values above the detection limit are described as right censored and values below the detection level are described as left censored. Trends are most robust when there are few censored values in the time-period of analysis.

When calculating point statistics such as means, standard deviations or quantiles, it is appropriate to replace censored values with imputed values. The LWP-Trends library included the functions Impute.lower() and Impute.upper(), which impute replacement values for lower (left) censored data and upper (right) censored data respectively. The Impute.lower() function is based on regression on order statistics (ROS) function from the NADA package. The Impute.upper() function is based on the survreg() function which is also from the NADA package. Both methods are based on fitting a distribution to the non-censored values and using that model to impute replacement values for the censored values and are described in detail in Nondetects and Data Analysis for Environmental Data[1] and Statistics for censored environmental data using MINITAB and R[2].

Censored values in the data used to calculate Kendall’s S and its p-value are robustly handled in the manner recommended by Hesel (2005, 2012). Briefly, for left-censored data, increases and decreases in a water quality variable are identified whenever possible. Thus, a change from a censored data entry of <1 to a measured value of 10 is considered an increase. A change from a censored data entry of <1 to a measured value 0.5 was considered a tie, as is a change from <1 to a <5, because neither can definitively be called an increase or decrease. Similar logic applied to right censored values. The information about ties is used in the calculation of the Kendall S statistic and its variance following Helsel (2012) and this provides for robust calculation of the p-value associated with the Kendall test.

Note that as the proportion of censored values increases, the proportion of ties increases and the confidence in the trend direction decreases. Therefore, trends calculated from data with high proportions of censored observations tend to be categorised as indeterminant.

The inter-observation slope cannot be definitively calculated between any combination of observations in which either one or both are censored. Therefore, when SSE and SSSE (i.e., Sen slopes) are calculated by the LWP-Trends library, the censored data entries are replaced by their corresponding raw values (i.e., the numeric component of a censored data entry) multiplied by a factor (0.5 for left-censored and 1.1 for right-censored values). This ensures that any measured value that is equal to a raw value is treated as being larger than the censored value if it is left-censored value and smaller than the censored value if it is right-censored. The inter-observation slopes associated with the censored values are therefore imprecise (because they are calculated from the replacements). However, because the Sen slope is the median of all the inter-observation slopes, the Sen slope is unaffected by censoring when a small proportion of observations are censored. As the proportion of censored values increase, the probability that the Sen slope is affected by censoring increases.

Helsel (1990) estimated that the impact of censored values on the Sen slope is negligible when fewer than 15% of the values are censored. However, this is a rule of thumb and is not always

---

[1] Helsel, D. R., 2005. Nondectects and Data Analysis; Statistics for censored environmental data. John Wiley and Sons, USA, NJ.

[2] Helsel, D. R., 2012. Statistics for censored environmental data using MINITAB and R, John Wiley & Sons.

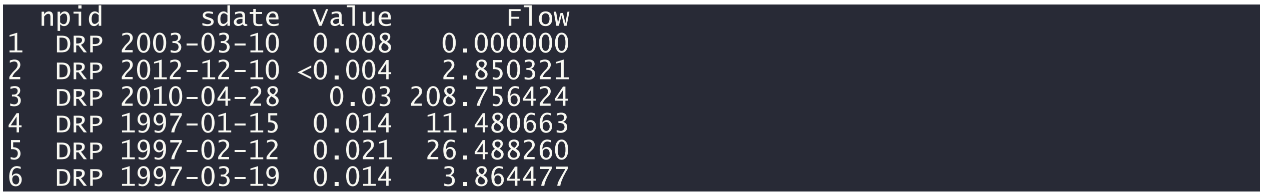
true. Depending on the arrangement of the data, a small proportion of censored values (e.g., 15% or less) could affect the computation of a Sen slope (Helsel, 2012). To provide information about the robustness of the SSE and SSSE values, the output from LWPTrends includes the proportion of observations that were censored and whether the Sen slope (i.e., the median of all inter-observation slopes) was calculated from observations that were censored. The estimate of the magnitudes (i.e., the SSE and SSSE values) and confidence intervals of individual site trends decreases in reliability as the proportion of censored values increases. In addition, when there are censored values, greater confidence should be placed in the statistics returned by the Kendall tests (including the trend direction and the probability the trend was decreasing).

It is noted that the functions have the option to scan the data to find the highest censoring limit (for left censored data), set any recorded values that are less than this value to the highest censoring limit and set these as censored (at the highest censoring limit). This may be appropriate if changing analytical methods over time have changed the reporting limit and thereby risk inducing a trend in the data. This is achieved by setting the argument HiCensor = TRUE in the Kendall test and Sen slope estimator functions described below.

3 Data and preliminary set up

These functions are designed to undertake trend analyses on data pertaining to a single site + variable. The functions can be used to analyse data that pertains to many sites + variable combinations by applying the functions to appropriately sub-setted data using (for example the ddply function from the plyr package).

It is expected that the data is in an R data frame format such that each water quality observation is a row. Each row must have columns that define the value, the date and the value of any covariate (e.g., flow). If the data pertains to many sites + variable combinations, a column must specify the variable name. An example of this type of data is shown in Figure 3.



```
  npid      sdate  Value       Flow
1  DRP 2003-03-10  0.008   0.000000
2  DRP 2012-12-10 <0.004   2.850321
3  DRP 2010-04-28   0.03 208.756424
4  DRP 1997-01-15  0.014  11.480663
5  DRP 1997-02-12  0.021  26.488260
6  DRP 1997-03-19  0.014   3.864477
```

Figure 3. Example of minimum water quality data for trend analysis. In this example, sID is the site name/identifier and npID is the water quality variable name.

3.1 Date Format

The first step is to add a column called myDate that represents the date of each observation as a vector of class “Date”. This is achieved for the above data (which are a data frame called WQData) with following command:

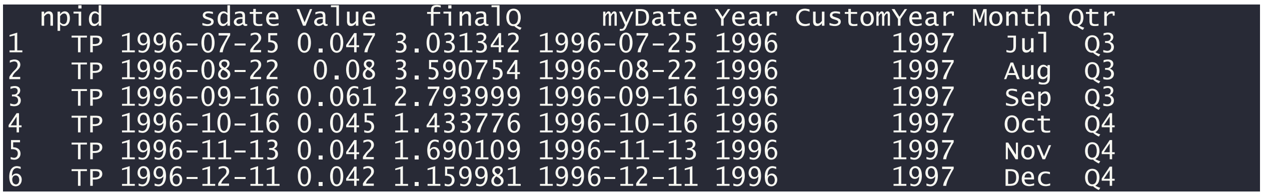
```
WQData_Ex1a$myDate <- as.Date(as.character(WQData_Ex1a$sdate),"%Y-%m-%d")
```

Note, the format “%d/%m/%Y” will need to be adjusted to match the format of the user input data.

3.2 Adding additional date labels

Additional date information used by the subsequent trend functions is added with the function GetMoreDateInfo(). This function uses myDate and has an optional argument “firstMonth” that can be used to choose an alternative start month for the analysis (i.e first month=7 would provide seasons that start in July an output a “custom year” that starts in July, default is that the year starts in January). It provides additional columns describing the months, quarters, and years – these are required for the analysis, and are common choices for the season of the dataset. These columns are factors. If the year is shifted, the factor levels for these additional columns are also shifted (this is useful for plotting purposes later).

```
WQData_Ex1b<-GetMoreDateInfo(WQData_Ex1b,firstMonth = 7)
```



```
  npid      sdate Value   finalQ     myDate Year CustomYear Month Qtr
1   TP 1996-07-25 0.047 3.031342 1996-07-25 1996       1997   Jul  Q3
2   TP 1996-08-22  0.08 3.590754 1996-08-22 1996       1997   Aug  Q3
3   TP 1996-09-16 0.061 2.793999 1996-09-16 1996       1997   Sep  Q3
4   TP 1996-10-16 0.045 1.433776 1996-10-16 1996       1997   Oct  Q4
5   TP 1996-11-13 0.042 1.690109 1996-11-13 1996       1997   Nov  Q4
6   TP 1996-12-11 0.042 1.159981 1996-12-11 1996       1997   Dec  Q4
```

The next step is to select the column that defines the time-period increment (or “season”). Note: the year must be a numeric field and the season must be a factor. Any user-defined season can be analysed. For example, selecting months as seasons, can be implemented by:

```
WQData$Season<-WQData_Ex1b$Month
```

A string describing the season names must also be specified.

```
SeasonString<-levels(WQData_Ex1b$Season)
```

3.3 Processing Censored Data

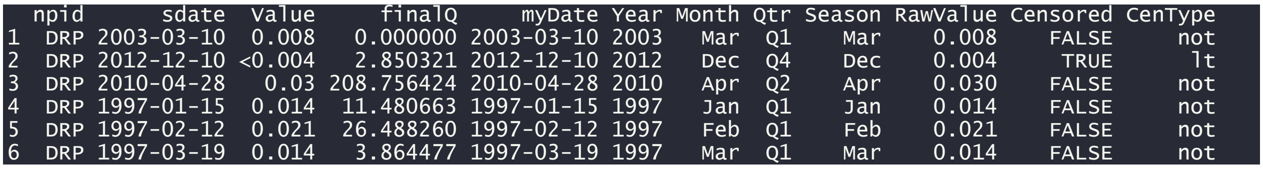
The next step assumes that the water quality measures contain less than and greater than signs that signify the data are censored (below detection limit is “<” and above the “reporting limit” “>”). These must be converted to their face values + information concerning censoring. This is achieved with the function RemoveAlphaDetect() as follows:

```
NewValues <- RemoveAlphaDetect(WQData_Ex1a$Value)
```

This output is a data frame with three columns and as many rows as the input data frame. The columns represent the face value of the water quality measures (named RawValue), a logical indicating if the observation was censored (named Censored) and the type of censoring (less than (lt), greater than (gt) or not censored (not). This data frame is then concatenated to the original data with the following command:

```
WQData_Ex1b <- cbind.data.frame(WQData_Ex1a, NewValues)
```

The modified WQData_Ex1b data frame is shown in Figure 4.



```
  npid      sdate  Value     finalQ     myDate Year Month Qtr Season RawValue Censored CenType
1  DRP 2003-03-10  0.008   0.000000 2003-03-10 2003   Mar  Q1    Mar    0.008    FALSE     not
2  DRP 2012-12-10 <0.004   2.850321 2012-12-10 2012   Dec  Q4    Dec    0.004     TRUE      lt
3  DRP 2010-04-28   0.03 208.756424 2010-04-28 2010   Apr  Q2    Apr    0.030    FALSE     not
4  DRP 1997-01-15  0.014  11.480663 1997-01-15 1997   Jan  Q1    Jan    0.014    FALSE     not
5  DRP 1997-02-12  0.021  26.488260 1997-02-12 1997   Feb  Q1    Feb    0.021    FALSE     not
6  DRP 1997-03-19  0.014   3.864477 1997-03-19 1997   Mar  Q1    Mar    0.014    FALSE     not
```

Figure 4. Water quality data after initial set up steps.

4 Performing a trend analysis

The following sections present the main functions of the LWPTrends library, providing examples from the accompanying demonstration dataset.

4.1 Inspecting the data

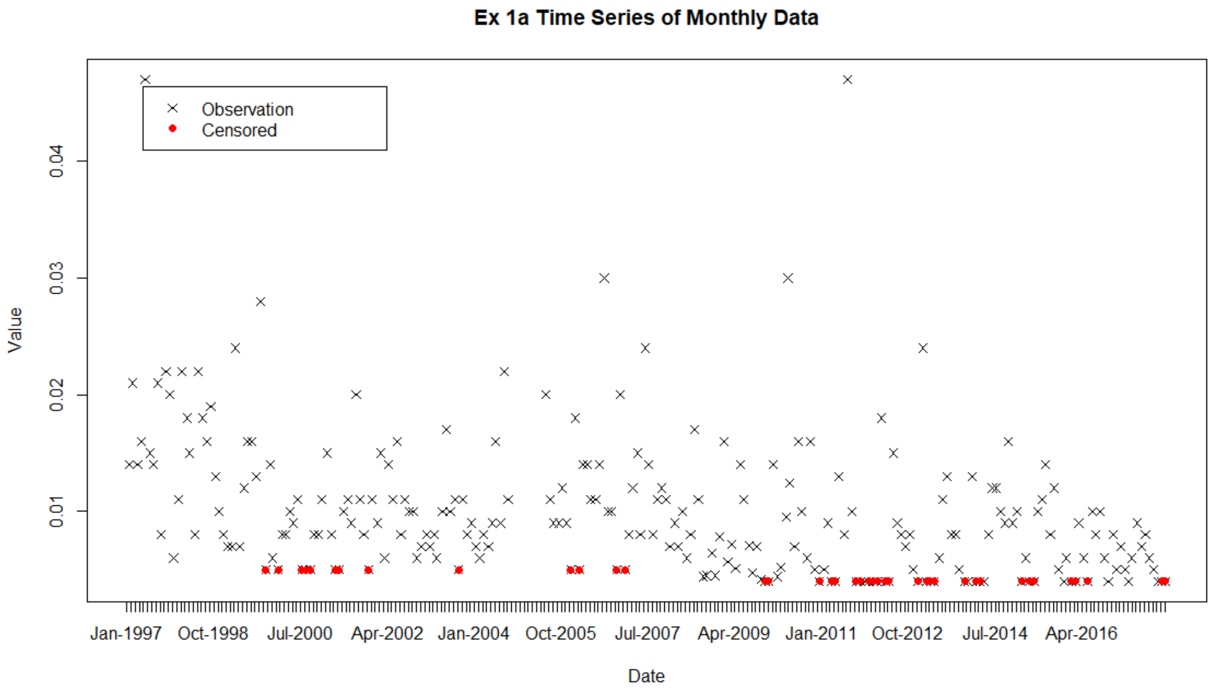
4.1.1 Inspect Data

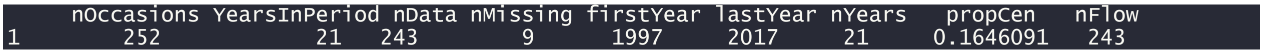
The function InspectData() assists with inspection of the data. It is assumed that the trend analysis is for a specific time-period that is defined by the arguments Year and StartYear and EndYear. The function outputs a data frame that defines the number of observation occasions (nOccasions – i.e. a specific Year + Season combination) in the specified time-period, the number of actual observations (nData), the number of missed occasions (nMissing), the proportion of observations that are censored (propCen) and the number of observations with flow data (nFlow). The function takes the median value of observations if there are more than one observation in Year + Season combination. When there is more than one value in a Year + Season combination and one or more of these is censored, the function assigns the observations as Censored if the median value is less than or equal to the maximum censored value within that Year + Season combination. Note, the Year can be specified as “CustomYear” (or other name) if, for example, the user wishes to perform the analysis on a water year rather than a calendar year.

The argument plotType is used to specify a plot of the data as either a time series or heat plot matrix (Years x Seasons). The InspectData() function is used to produce a time series plot with following commands:

```
x11();InspectData(WQData_Ex1a, StartYear = 1997, EndYear = 2017, FlowtoUse = "finalQ",plotType = "TimeSeries", main= "Ex 1a Time Series of Monthly Data")
```

Note, use Year=”CustomYear” if not using calendar years.





```
  nOccasions YearsInPeriod nData nMissing firstYear lastYear nYears   propCen   nFlow
1        252            21   243        9      1997     2017     21 0.1646091   243
```

Figure 5. Plot and data frame output from InspectData().

The InspectData() function is used to produce a matrix plot of the data with following commands:

```
x11();InspectData(WQData_Ex1a,  StartYear  =  1997,  EndYear  =  2017,  plotType  =
"Matrix",PlotMat="RawData",main="Matrix of Values: monthly data")
```

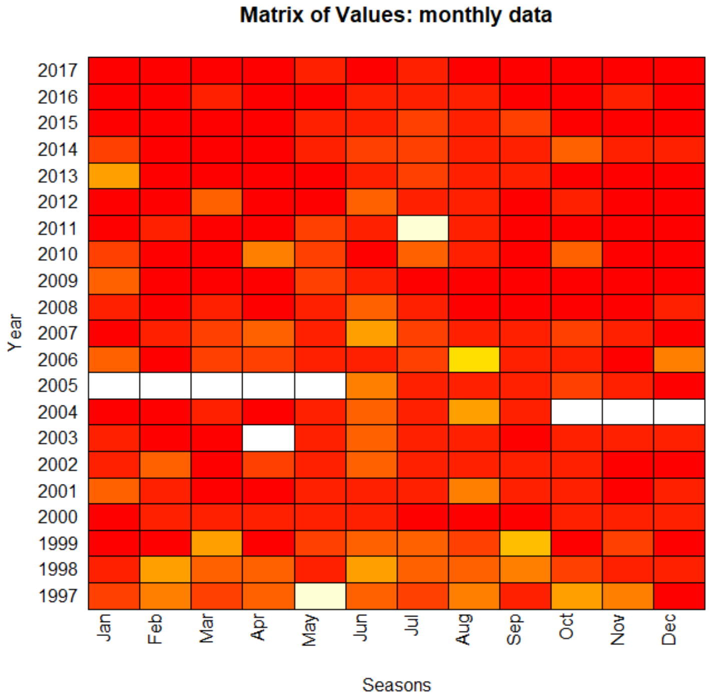


Figure 6. Heat matrix plot output from InspectData().

The InspectData() function is used to produce a matrix plot of the censoring occasions with following commands:

```
x11(); InspectData(WQData_Ex1a, StartYear = 1997, EndYear = 2017, plotType =
"Matrix",PlotMat="Censored",main="Matrix of censoring: monthly data")
```

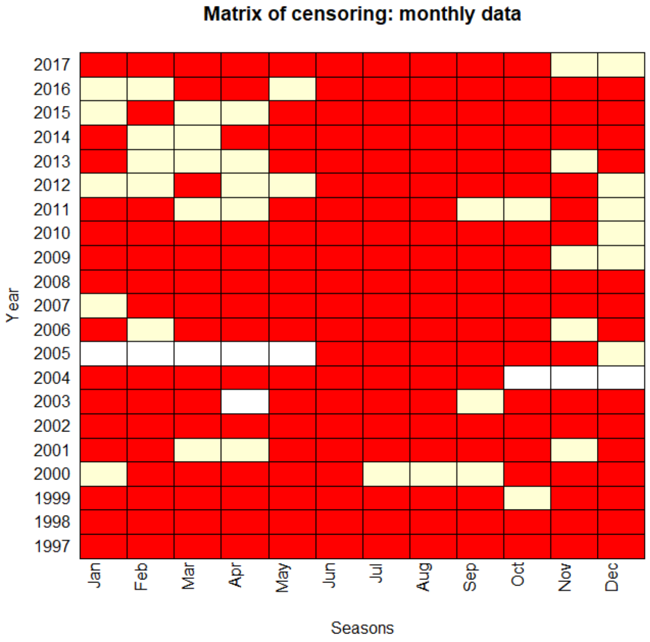
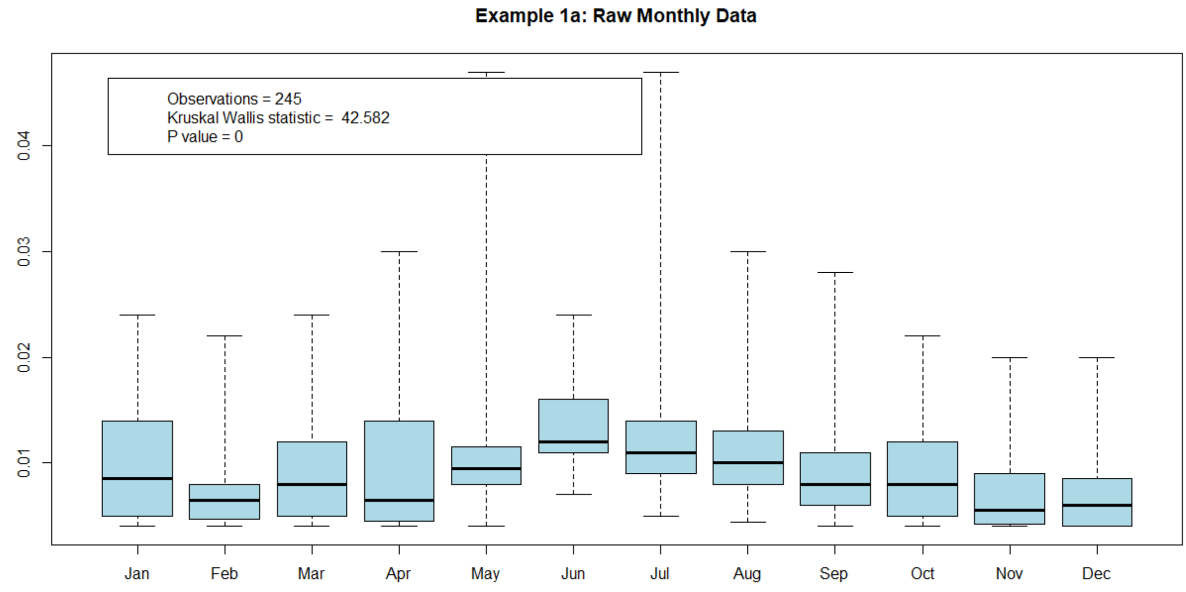


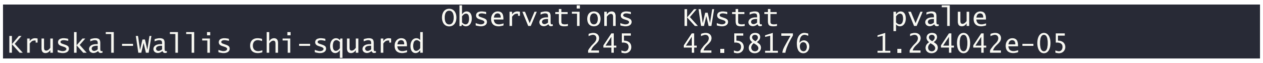
Figure 7. Heat matrix plot output showing the censored observations from InspectData().

Note, if there are no censored data the plot will be one colour; orange.

4.1.2 Seasonality Test

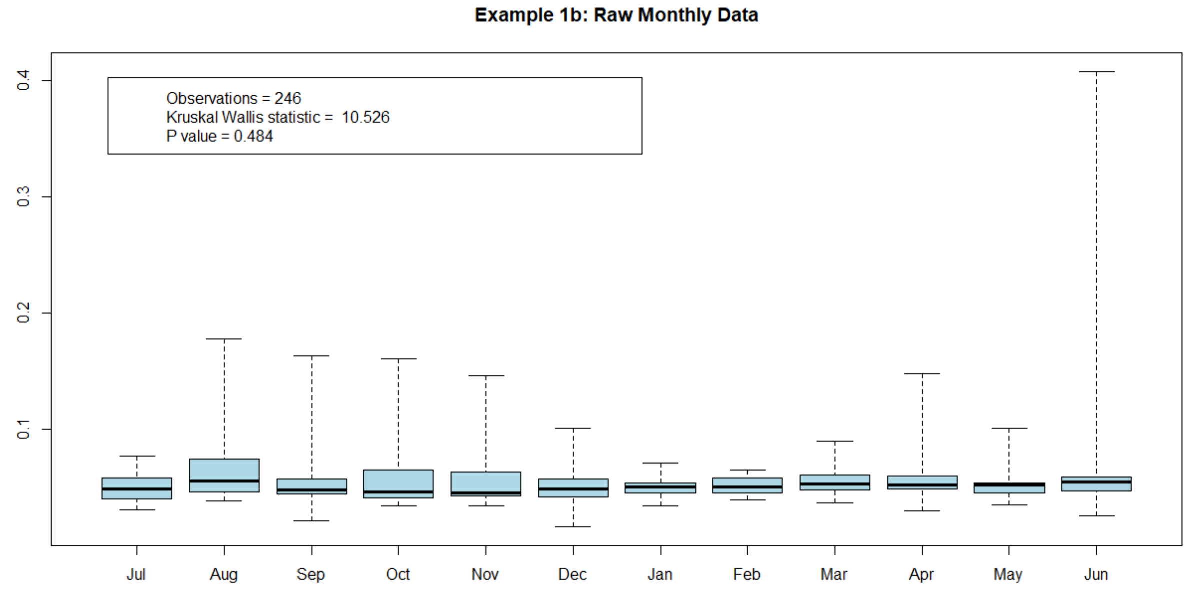
The function SeasonalityTest() tests if the data are seasonal. If the data are seasonal, the trend analysis should be seasonal (i.e. the Kendall test should be the seasonal Kendall test and the Sen slope should be a seasonal Sen slope). The SeasonalityTest() function performs a Kruskal Wallis test (non-parametric ANOVA) on the observations using season as the explanatory (categorical) variable. The function also produces a box plot of the data grouped by season (e.g. Figure 8 and Figure 9).

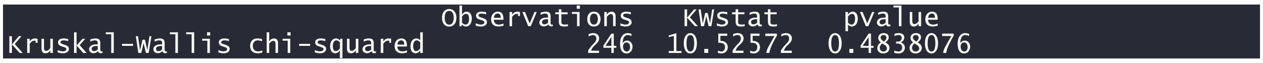




```
                          Observations    KWstat        pvalue
Kruskal-Wallis chi-squared          245  42.58176  1.284042e-05
```

Figure 8. Plot and data frame output from SeasonalityTest() for example seasonal data set.





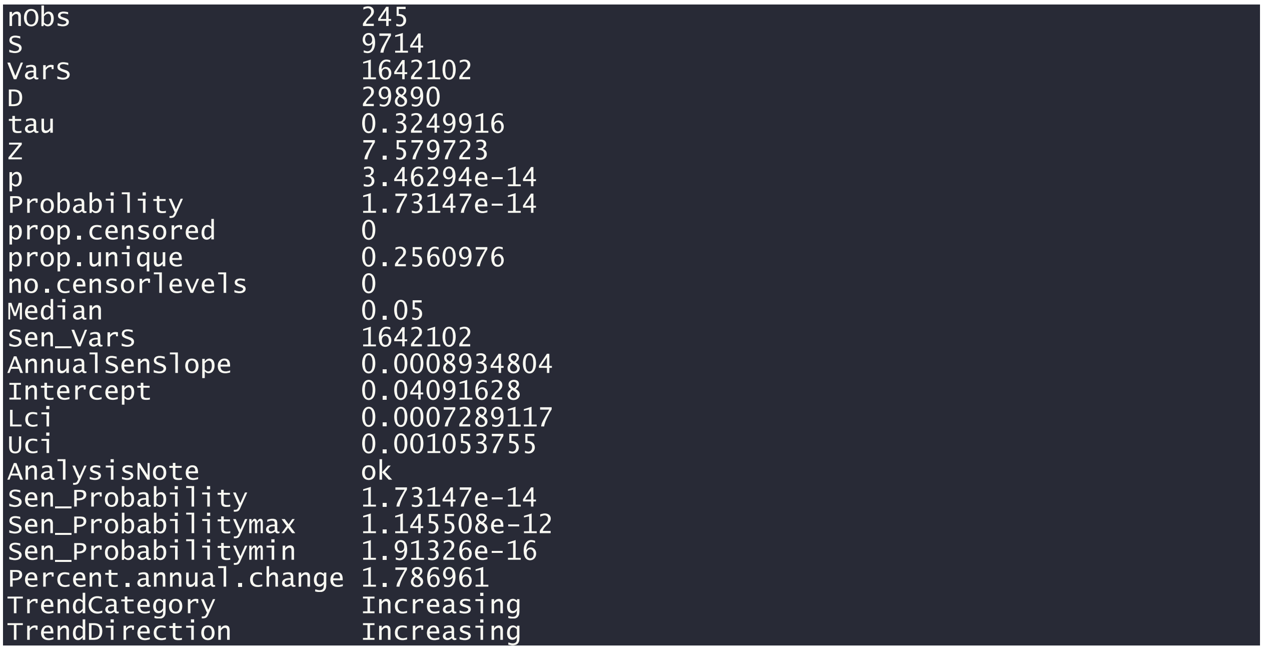
```
                          Observations    KWstat    pvalue
Kruskal-Wallis chi-squared          246  10.52572 0.4838076
```

Figure 9. Plot and data frame output from SeasonalityTest() for example non-seasonal dataset.

4.2 Non-seasonal trend analysis

The function NonSeasonalTrendAnalysis () performs a Mann Kendall test of correlation between the observations and time, followed by a non-parametric regression (Sen's slope estimator, also known as the Theil–Sen estimator) to the observations versus time. The function handles the censored values as described in Handling Censored Values section. The output is a data frame of the test results. Probability is the probability that the trend is decreasing.

```
NonSeasonalTrendAnalysis(WQData_Ex1b,mymain="Ex 1b Raw Trend: Monthly
Data",Year="CustomYear",doPlot=T)
```



```
nObs                   245
S                      9714
VarS                   1642102
D                      29890
tau                    0.3249916
Z                      7.579723
p                      3.46294e-14
Probability            1.73147e-14
prop.censored          0
prop.unique            0.2560976
no.censorlevels        0
Median                 0.05
Sen_VarS               1642102
AnnualSenSlope         0.0008934804
Intercept              0.04091628
Lci                    0.0007289117
Uci                    0.001053755
AnalysisNote           ok
Sen_Probability        1.73147e-14
Sen_Probabilitymax     1.145508e-12
Sen_Probabilitymin     1.91326e-16
Percent.annual.change  1.786961
TrendCategory          Increasing
TrendDirection         Increasing
```

Figure 10. Data frame output from the NonSeasonalTrendAnalysis function. Note the data frame has been transposed for display purposes.

The function is a master function that calls both the MannKendall() and SenSlope() functions. Each of these functions can be standalone, but note that (1) if the outputs from the MannKendall() function suggest that there are insufficient data, it is not necessary to go on to run the SenSlope() function and (2) the SenSlope() function requires the probability output from the MannKendall() function as an input, for the purposes of plotting.

Note, the NonSeasonalTrendAnalysis() (and internally the MannKendall() function) and the other trend assessment functions below, have the argument HiCensor, If the argument HiCensor = TRUE the functions find the highest censoring limit (for left censored data) and set any recorded values that are less than this value to the highest censoring limit and set these as censored (at the highest censoring limit).

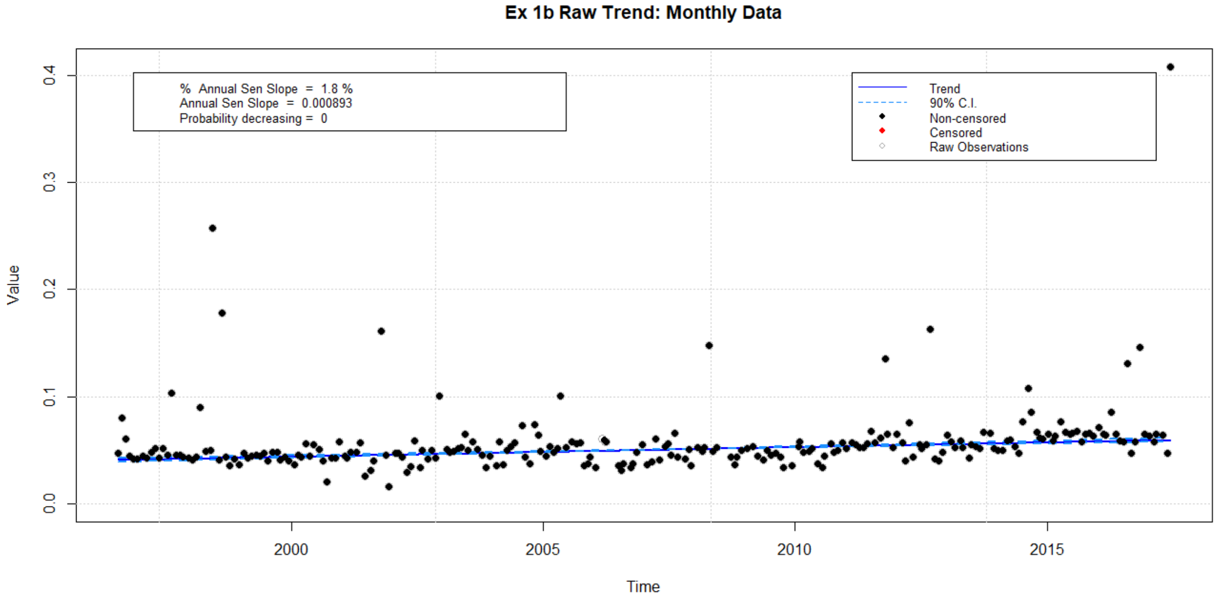


Figure 11. Plot and data frame output from SenSlope().

The Sen_Probability output is the probability that the true trend is decreasing derived as part of the Sen slope calculations. This is evaluated by interpolating from a value of zero on the cumulative distribution of the ranked inter-observation slopes. When there is low precision or large numbers of censored values, there can be many pairs of observations with the same value (i.e., ties) resulting in many inter-observation slopes equal to zero. This situation requires that choices are made when interpolating the probability. The LWP-Trends default is to return the median probability of all inter-observation slopes that are equal to zero (Figure 12). Other possible choices are to use either the maximum or minimum probability for which inter-observation slopes are equal to zero (Figure 12). These probabilities are included in the outputs of the SenSlope() and SeasonalSenSlope functions as “Sen_ProbabilitiyMin”, and “Sen_ProbabilityMax”.

For variables with low precision or large numbers of censored values, the Probability returned by the MannKendall() or SeasonalKendall() functions are the preferred estimates.

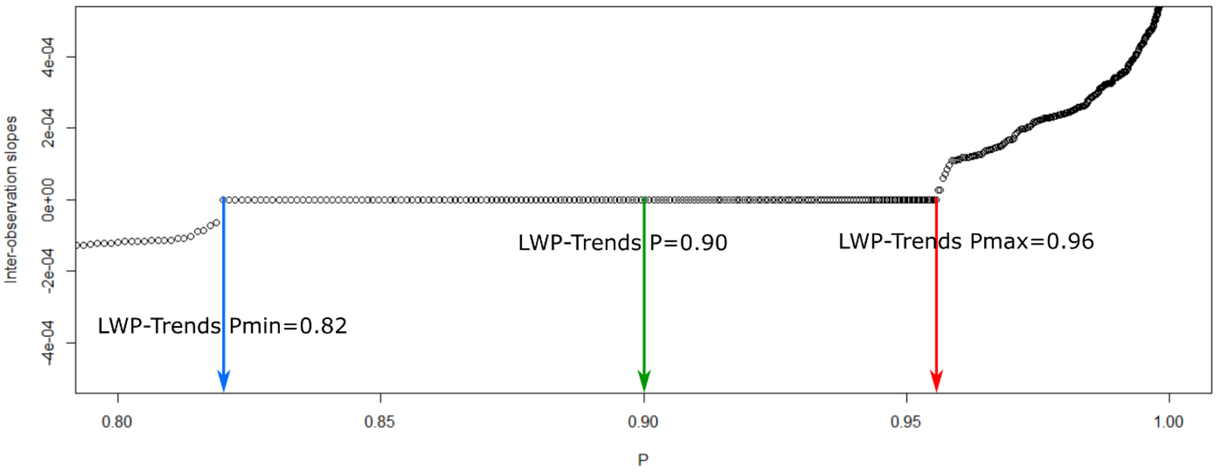


Figure 12: Example of cumulative distribution of the ranked inter-observation slopes – for site with negative Sen slope and many inter-observation ties.

The percentage annual change in trend slope is calculated as the Sen slope divided by the median, multiplied by 100. The default is to calculate the median using the face values of the RAW data (including censored data).

When there are insufficient data, the models will return NULL outputs – labelled “Not analysed”. Trends were classified as “not analysed” for two reasons:

1) When a large proportion of the values were censored (data has <5 non-censored values and/or <3 unique non-censored values). This arises because trend analysis is based on examining differences in the value of the variable under consideration between all pairs of sample occasions. When a value is censored, it cannot be compared with any other value and the comparison is treated as a “tie” (i.e., there is no change in the variable between the two sample occasions). When there are many ties there is little information content in the data and a meaningful statistic cannot be calculated.

2) When there is no, or very little variation in the data (<3 unique non-censored values, or a long run of identical values), because this also results in ties. This can occur because laboratory analysis of some variables has low precision (i.e., values have few or no significant figures). In this case, many samples have the same value resulting in ties.

4.2.1 Analysis Notes: warnings description

"WARNING: Sen slope based on two censored values"

This message occurs when the pair of observations that produce the median inter-observation slope are both censored. This can be > or < censoring, although generally, we find that this is a pair of < censored values. This is common at sites with high levels of censoring. The returned Sen slope will be zero, and this warning recognises that the ‘true’ Sen slope (i.e., if the water quality results were analysed and reported with very high precision) would not be zero. In other words, the ‘true’ Sen slope would be a small value that cannot be evaluated due to the lack of precision of the observations. Cases where the Sen slope is reported as zero and UCI and LCI are also zero, indicate that most observations are censored. In these cases, the ‘true’ Sen slope, UCI and LCI would be non-zero values if the precision of the observations was higher. Generally, in these cases the direction of trend is also uncertain.

"WARNING: Sen slope influenced by censored values"

This message occurs when the one of the two observations that produces the median inter-observation slope is censored. This can be > or < censoring, although generally, we find that this is associated with a < censored value. This can occur by chance, but it is more common at sites with high levels of censoring. This warning recognises that the estimate of the Sen slope uncertainty does not account for the imprecision associated with calculating Sen slopes using values that are below the detection limit. Theoretically, this imprecision adds to the uncertainty of the Sen slope estimate, but this is not accounted for in the calculations. If the detection limit is small relative to the range of observation values, the additional uncertainty will be small.

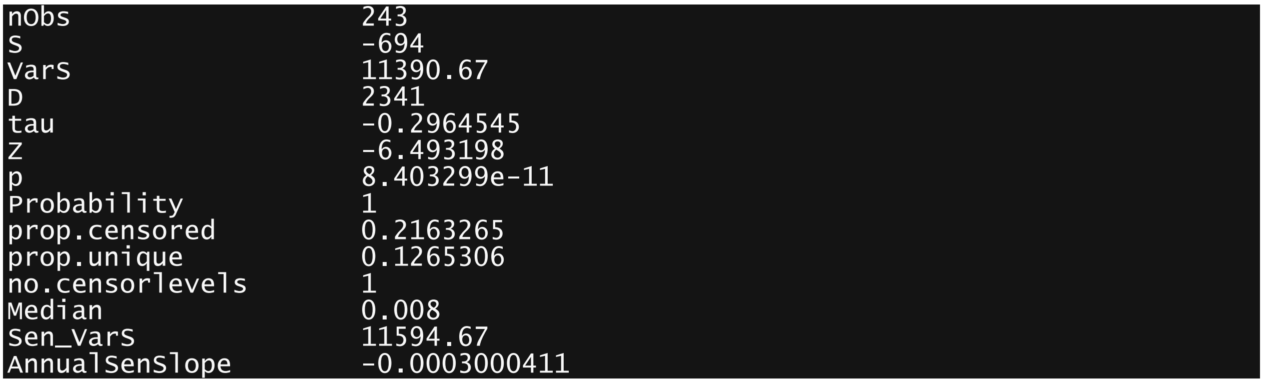
"WARNING: Sen slope based on tied non-censored values"

This message occurs when the pair of observations that produce the median inter-observation slope are the same but are not censored. In this case, the Sen Slope is reported as zero. This is common at sites where the observations comprise many repeated values, due to the imprecision of the observations. The returned Sen slope is not the ‘true’ Sen slope due to imprecision. The ‘true’ Sen slope would be a small value that cannot be evaluated due to the lack of sufficient precision of the observations.

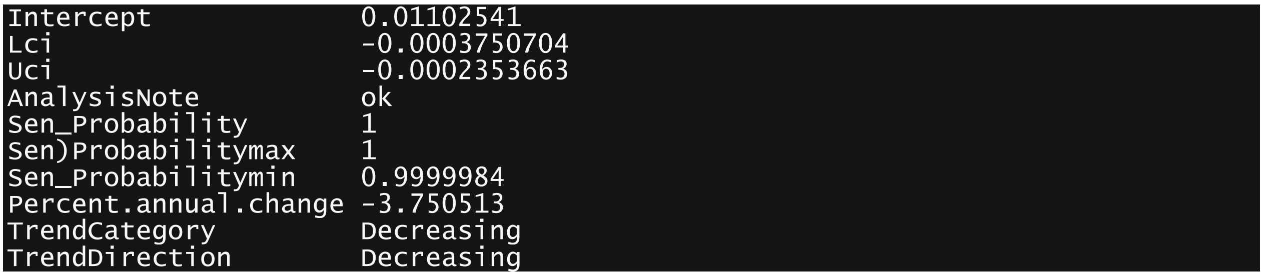
4.3 Seasonal Trend Analysis

The function SeasonalTrendAnalysis() performs a seasonal Kendall test of correlation between the observations and time and a seasonal version of the non-parametric regression (Sen's slope estimator) of the observations versus time. The function handles the censored values as described in Handling Censored Values section. The output is a data frame of the test results.

```
SeasonalTrendAnalysis(WQData_Ex1a,mymain="Ex 1a Raw Trend: Monthly Data",doPlot=T)
```



```
nObs                    243
S                       -694
VarS                    11390.67
D                       2341
tau                     -0.2964545
Z                       -6.493198
p                       8.403299e-11
Probability             1
prop.censored           0.2163265
prop.unique             0.1265306
no.censorlevels         1
Median                  0.008
Sen_VarS                11594.67
AnnualSenSlope          -0.0003000411
```



```
Intercept             0.01102541
Lci                   -0.0003750704
Uci                   -0.0002353663
AnalysisNote          ok
Sen_Probability       1
Sen)Probabilitymax    1
Sen_Probabilitymin    0.9999984
Percent.annual.change -3.750513
TrendCategory         Decreasing
TrendDirection        Decreasing
```

Figure 13. Data frame output from the SeasonalKendall function. Note the data frame has been transposed for display purposes.

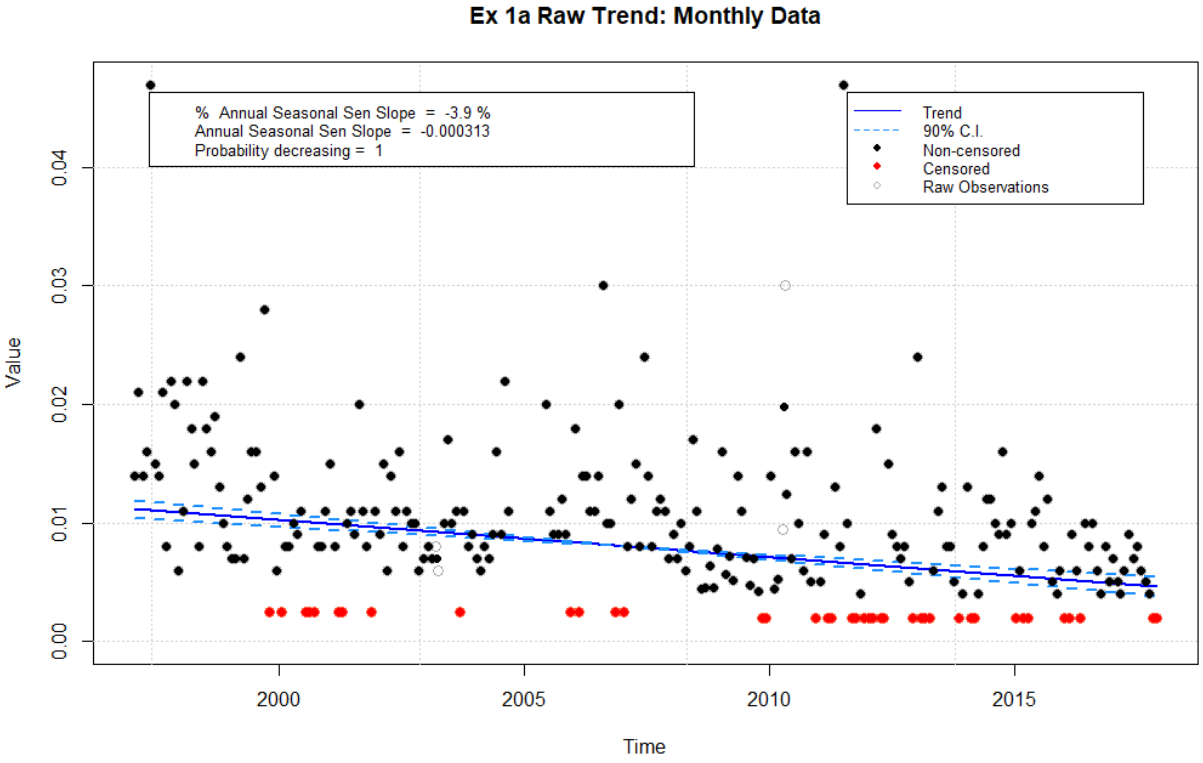


Figure 14. Plot and data frame output from SeasonalSenSlope().

The same additional outputs as described for the Sen slope are also provided for the seasonal Sen slope. Note, there are generally more not analysed sites for the seasonal Sen slope tests, as there must be sufficient data for each season.

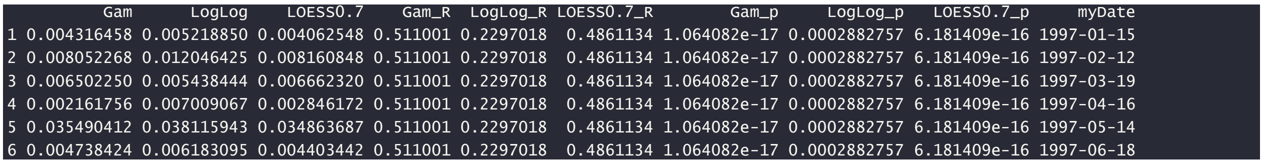
4.4 Flow adjustment

The function AdjustValues() performs an adjustment of the data based on a covariate (for example flow if the data represents river concentrations). The function fits a variety of models to the observation versus covariate relationship. The user needs to consider which of these models is the most appropriate basis for adjustment. The function returns a data frame with the adjusted values (regression residuals) for each of the models. The column that represents the chosen model in this data frame is then used in any of the functions above. The adjustment is achieved with following command:

```
FlowAdjusted<-AdjustValues(WQData_Ex1a, method = c(Gam, LogLog, LOESS), ValuesToAdjust
= RawValue, Covariate = Flow, Span = c(0.7, 0.79), doPlot = T)
```

The adjusted output is shown on Figure 15 and the plots are shown in Figure 16.

```
head(FlowAdjusted)
```



```
          Gam      LogLog   LOESS0.7    Gam_R  LogLog_R LOESS0.7_R        Gam_p     LogLog_p   LOESS0.7_p     myDate
1 0.004316458 0.005218850 0.004062548 0.511001 0.2297018  0.4861134 1.064082e-17 0.0002882757 6.181409e-16 1997-01-15
2 0.008052268 0.012046425 0.008160848 0.511001 0.2297018  0.4861134 1.064082e-17 0.0002882757 6.181409e-16 1997-02-12
3 0.006502250 0.005438444 0.006662320 0.511001 0.2297018  0.4861134 1.064082e-17 0.0002882757 6.181409e-16 1997-03-19
4 0.002161756 0.007009067 0.002846172 0.511001 0.2297018  0.4861134 1.064082e-17 0.0002882757 6.181409e-16 1997-04-16
5 0.035490412 0.038115943 0.034863687 0.511001 0.2297018  0.4861134 1.064082e-17 0.0002882757 6.181409e-16 1997-05-14
6 0.004738424 0.006183095 0.004403442 0.511001 0.2297018  0.4861134 1.064082e-17 0.0002882757 6.181409e-16 1997-06-18
```

Figure 15. Data frame output by the AdjustValues function. The suffix '_R' indicates the correlation coefficient of the fitted model and '_p' indicate the p-values of the fitted model

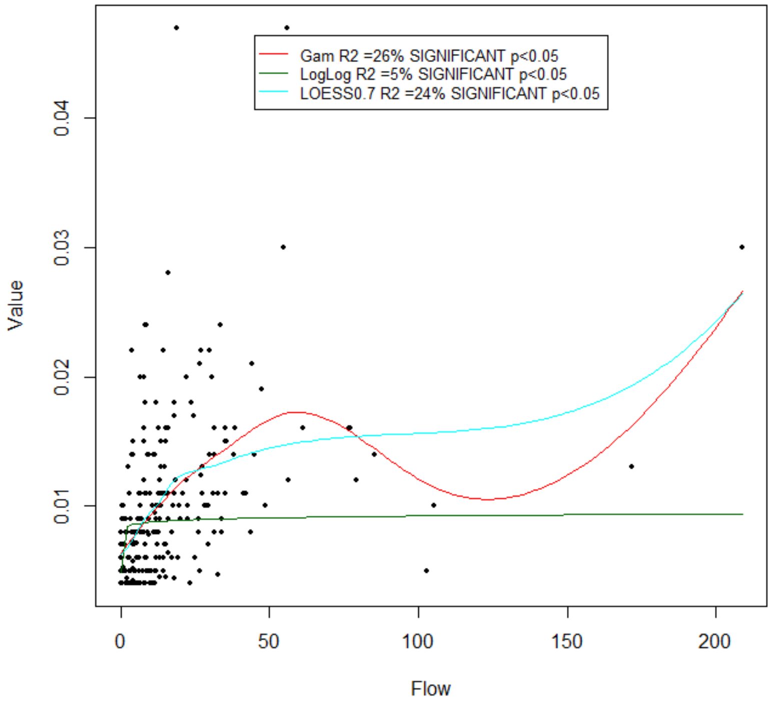
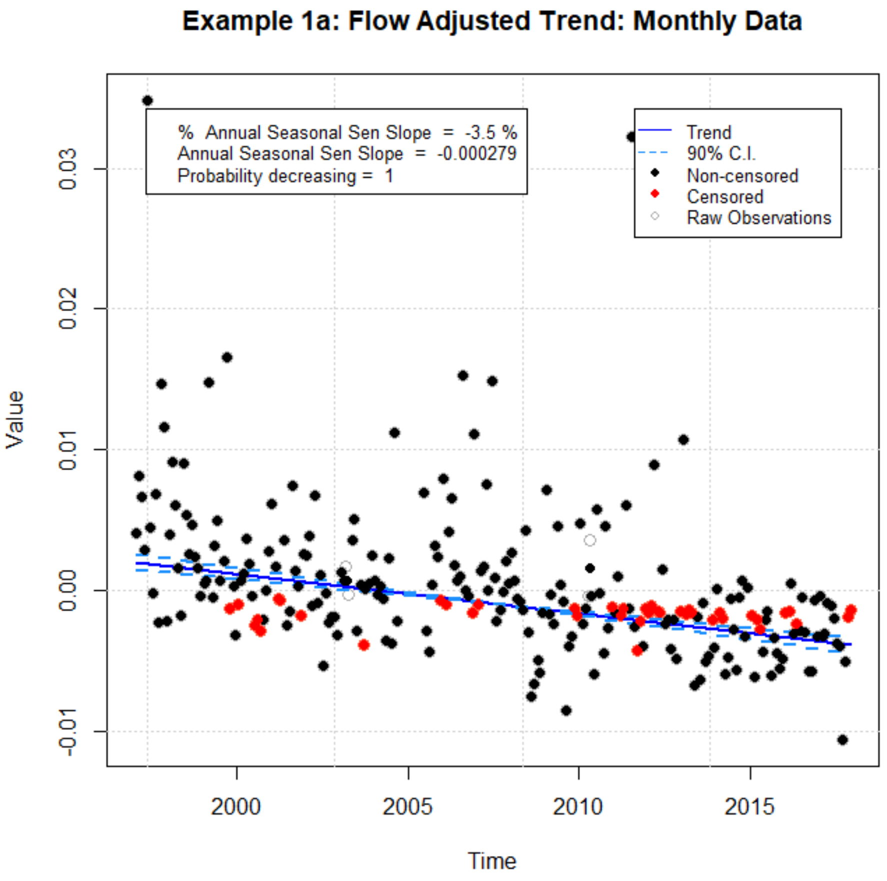


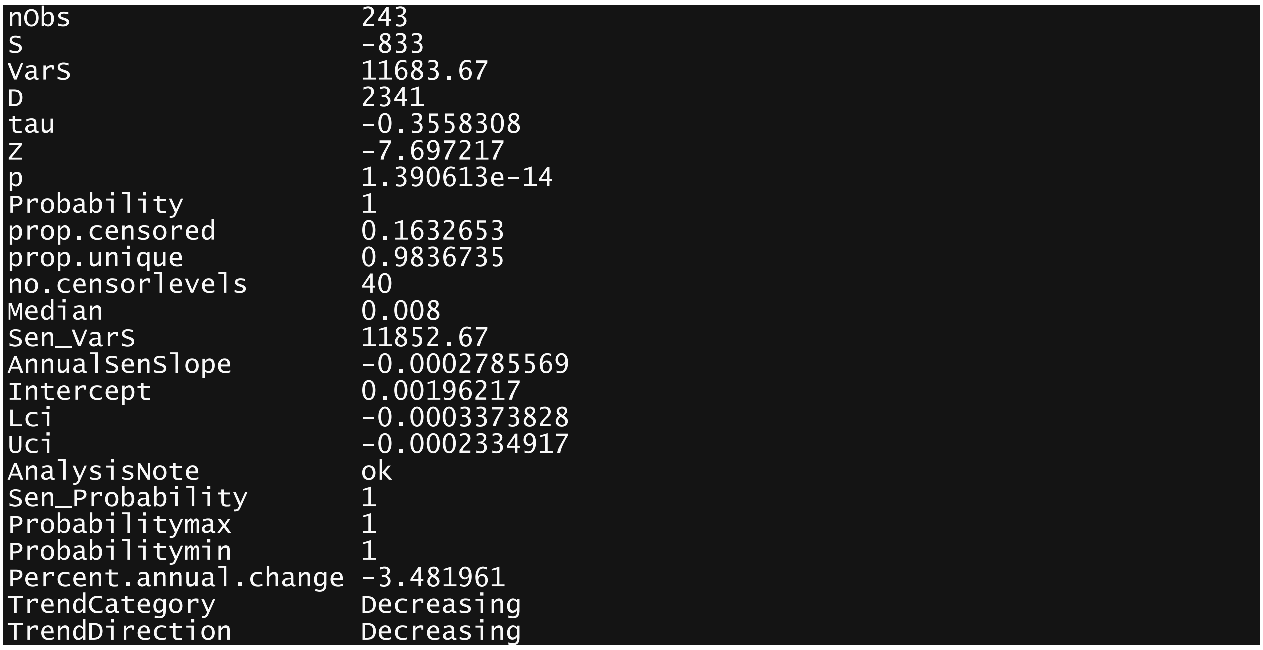
Figure 16. Output plots from the GetAdjustment function.

In the above example the LOESS model is probably the most appropriate, although no model is particularly robust in this case. The adjusted values that are output by the GetAdjustment function are used in any of the above four trend analysis functions with following commands as follows:

```
WQData_Ex1a<-merge(WQData_Ex1a,FlowAdjusted[,c(myDate,LOESS0.7)])

SeasonalTrendAnalysis(WQData_Ex1a,ValuesToUse=LOESS0.7,mymain=Example 1a: Flow
Adjusted Trend: Monthly Data,doPlot=T)
```





```
nObs                     243
S                        -833
VarS                     11683.67
D                        2341
tau                      -0.3558308
Z                        -7.697217
p                        1.390613e-14
Probability              1
prop.censored            0.1632653
prop.unique              0.9836735
no.censorlevels          40
Median                   0.008
Sen_VarS                 11852.67
AnnualSenSlope           -0.0002785569
Intercept                0.00196217
Lci                      -0.0003373828
Uci                      -0.0002334917
AnalysisNote             ok
Sen_Probability          1
Probabilitymax           1
Probabilitymin           1
Percent.annual.change    -3.481961
TrendCategory            Decreasing
TrendDirection           Decreasing
```

Figure 17. Plot and data frame output from SeasonalSenSlope() using flow adjusted values. Note the data frame has been transposed for display purposes.

5 Trend Aggregation

The aggregated results of analysis of water-quality trends are intended to provide an overview of recent water quality changes over a spatial domain of interest (e.g., environmental classes, regions, national). The LWP Trends functions provide two approaches for aggregating trend directions. Detailed descriptions of these approaches and comparisons with the previous approach are provided by Snelder and Fraser (2018).

For a given dataframe output with many rows, each being the outputs from the previously described trend analysis functions for a given site and variable combination. Confidence categories can be used to express the probability that a trend is improving (or its complement; degrading). The confidence categories can be assigned using the function ImprovmentConfCat():

```
AllTrends10FA$DirectionConf <- ImprovementConfCat(x=AllTrends10FA, Reverse=c(CLAR,MCI)
```

Note that the conversion of the probability that a trend is decreasing to the probability it is improving (and its complement, degrading) depends on whether decreasing values represent improvement or degradation and differs between variables. The Reverse argument is used to define those variables for which decreasing is degrading.

The categorical levels of confidence can be summarised using a colour coded bar chart (Figure 18). See the demonstration in the example file: RunLWPTrendsExample_Dec_19.R.

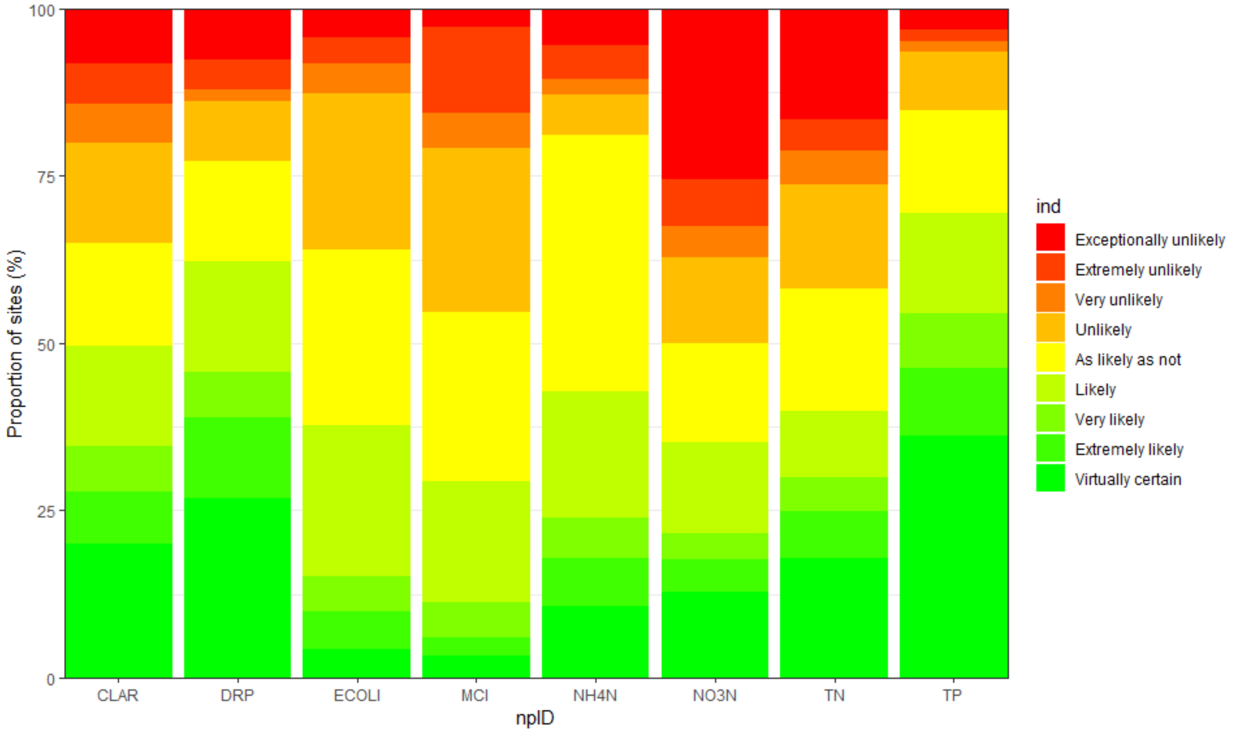
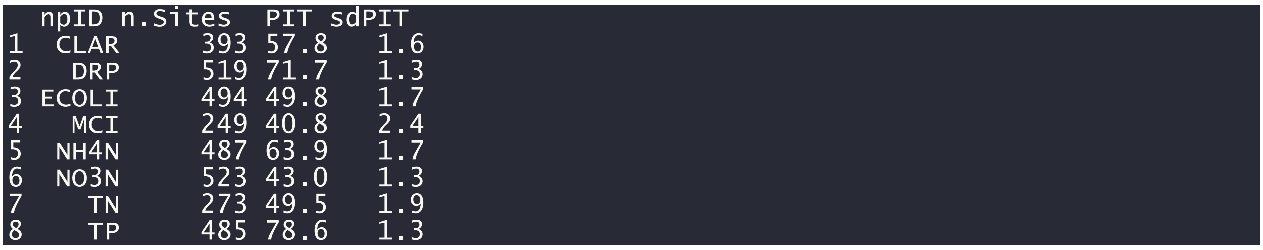


Figure 18. Example summary plot representing the proportion of water-quality monitoring sites with improving trends at each categorical level of confidence.

The second approach also utilises the probability that the true trend was decreasing to provide a probabilistic estimate of the proportion of improving trends (PIT) within a domain of interest. PIT and it's confidence can be calculated using the function AnalysePIT():

```
ddply(AllTrends10FA, “npID”, function(x) AnalysePIT(x,Reverse=c(CLAR,MCI))
```



```
  npID n.Sites  PIT sdPIT
1  CLAR     393 57.8   1.6
2   DRP     519 71.7   1.3
3 ECOLI     494 49.8   1.7
4   MCI     249 40.8   2.4
5  NH4N     487 63.9   1.7
6  NO3N     523 43.0   1.3
7    TN     273 49.5   1.9
8    TP     485 78.6   1.3
```

Figure 19. Output from the AnalysePIT() function.

The PIT statistics can be summarised using a plot (Figure 20). See the demonstration in the at the end of the example file.

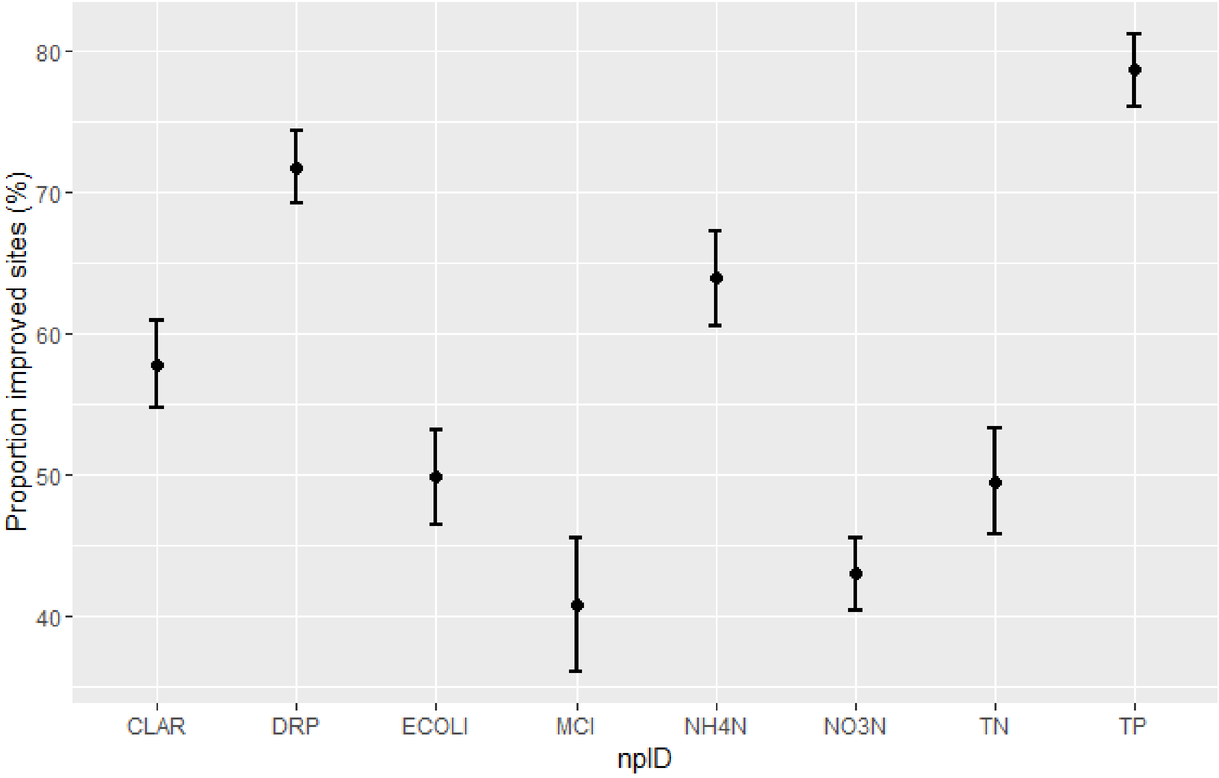


Figure 20. Example of PIT statistic plot.

6 Function descriptions

`GetMoreDateInfo`

Description

Takes dates and produces additional columns summarising, months, years, and quarters. If specified, the firstMonth can be used to shift the analysis year. This also automatically shifts the factor levels of the months and quarters to start from this month. The outputs from this function can be then used to specify a season for the analysis. Any other season definition would need to be manually shifted to reflect the custom year, by the user.

Usage

`GetMoreDateInfo(x,firstMonth=1)`

Arguments

`x`	Dataframe or vector containing myDate
`firstMonth`	Specify the first month (numeric (1:12)) for the analysis year

Value

A data frame with fields as follows (in addition to those columns in x)

`Year`	Calendar Year (numeric)
`Custom Year`	Custom Year (matches Calendar year of last month of the custom year) – only output if firstMonth!=1. numeric
`Month`	Month String (factor)
`BiMonth`	Bi-monthly String (factor)
`Qtr`	Quarter string (factor)

`RemoveAlphaDetect`

Description

Takes a character timeseries of observed values that includes censored values (specified as the prefix > or <) and returns face values and information about the nature of the censoring.

Usage

```
RemoveAlphaDetect(x)
```

Arguments

`x`	Dataframe or vector containing Value
`AdjustDL`	Option to adjust face value of low censored values by a factor (optional, default = 1)
`AdjustUpper`	Option to adjust face value of high censored values by a factor (optional, default = 1)

Value

A data frame with fields as follows

`RawValue`	Numeric face value from x$Value
`Censored`	Logical: whether the observation is censored or not
`CenType`	Factor indicating the censor type (lt: less than; gt: greater than, not: not censored)

`InspectData`

Description

InspectData provides an analysis of the input data, providing summary statistics and optional graphs.

Usage

```
InspectData (x, plotType = c("TimeSeries", "Matrix"), PlotMat =
c("RawData","Censored"),Year = "Year", StartYear = 1990, EndYear =
2015, doPlot = TRUE, ...)
```

Arguments

`x`	Input data frame must contain: myDate, RawData, Censored
`plotType`	Select type of plot time series or matrix, or “All_GGplot” (which returns all three plot types in a list as ggplot objects)
`PlotMat`	The column to be plotted in the matrix (RawData or Censored)
`Year`	Column name for Year data to be used
`StartYear`	Sets the start year of the time series. (If NA, will be selected from data)
`EndYear`	Sets the end year of the time series. (If NA, will be selected from data)
`doPlot`	Produce a plot if TRUE (not required if plottype==”All_GGPlot”)
`SiteName`	Site name – passed only to ggplots (which have automated titles)
…	Passed to base plot function

Value

A data frame and (optionally) a plot. Data frame fields as follows

`nOccasions`	Number of potential sampling occasions in the selected period
`YearsInPeriod`	Number of years in the period selected
`nData`	Number of actual sampling occasions in the period
`nMissing`	Number of missing observations for the selected period
`firstYear`	The year of the first observation
`lastyear`	The year of the last observation
`nYears`	The number of years with data in the sampling period
`propCen`	Proportion of observations that are censored
`nFlow`	The number of flow observations for the period

`SeasonalityTest`

Description

The function SeasonalityTest() tests if the data are seasonal. The SeasonalityTest() function performs a Kruskal Wallis test (non-parametric ANOVA) on the observations using season as the explanatory (categorical) variable. The function also produces a box plot of the data grouped by season[3].

Usage

```
SeasonalityTest(x, ValuesToUse=“RawValue”, HiCensor=FALSE,
do.plot=TRUE)
```

Arguments

`x`	Input data frame must contain, Season, ValuesToUse,Centype, Censored
`ValuesToUse`	Select Column with data to use in the test
`HiCensor`	If TRUE the function finds the highest censoring limit (for left censored data) and sets any recorded values that are less than this value to the highest censoring limit and sets these as censored (at the highest censoring limit).
`do.plot`	Produce a plot if TRUE. Return a ggplot object in a list if “doGGPlot”
…	Passed to plot function

Value

A data frame and (optionally) a plot. If do.plot set to “doGGPlot”, then a list is returned, with the first item the data frame and the second is the boxplot. Data frame fields as follows

`Observations`	Number of observations
`KWStat`	the Kruskal-Wallis rank sum statistic.
`pvalue`	The p-value for the test (null hypothesis is that the data is NOT seasonal)

---

[3] Note, the box plots between the base outputs (do.plot=TRUE) and the ggplot plots (do.plot=”doGGPlot”) vary slight in their treatment of the whiskers. For the base outputs, whiskers indicate the extent of the range of observations. For the ggplot box and whisker plots, the whiskers extend to a maximum of 1.5* the inter quartile range. Data beyond the whiskers are considered to outliers and plotted individually as points

`Impute.lower`

Description

Imputes replacement values for left-censored data using the regression on order statistics (ROS) function from the NADA package.

Usage

```
Impute.lower(x, ValuesToUse = "RawValue", forwardT="log", reverseT="exp", doPlot=F)
```

Arguments

`x`	Input data frame must contain: ValuesToUse, myDate, CenType
`forwardT`	See NADA function ros()
`reverseT`	See NADA function ros()
`doPlot`	Plot the ROS model

Value

The original data frame is returned with following fields added:

`i1Values`	Replacements for ValuesToUse
`LeftImpute`	Description of whether the observations were imputed or not

`Impute.upper`

Description

Impute replacement values for right censored observations (i.e., above (multiple) detection limits) uses the survreg function from (package::survival).

Usage

```
Impute.upper(x, ValuesToUse = "i1Values”)
```

Arguments

`x`	Input data frame must contain: ValuesToUse, myDate, CenType
`ValuesToUse`	The column of values for which right censored entries will be imputed. The default is i1Values as this is the column name for variables after imputation of left censored values.

Value

The original data frame is returned with following fields added:

`i2Values`	Replacements for ValuesToUse
`RightImpute`	Description of whether the observations were imputed or not

`NonSeasonalTrendAnalysis OR SeasonalTrendAnalysis`

Description

The function NonSeasonalTrendAnalysisl() performs a Mann Kendall test of correlation between the observations and time and a . The function SeasonalTrendAnalysis() performs a seasonal Mann Kendall test of correlation between the observations and time and a seasonal , and should be used if the data are determined to be seasonal by SeasonalityTest(). The function also returns some outputs relating to the censoring in the data. These can be used as diagnostic tools to determine the reliability of the estimate.

Usage

```
NonSeasonalTrendAnalysis(…)

OR

SeasonalTrendAnalysis(…)
```

Arguments

…	Variables to be passed to MannKendall() and SenSlope() functions

Value

If doPlot=”doGGPlot”, then a list is returned, with the dataframe as the first item and the plot as the second item. A data frame with fields as follows

`nObs`	Number of observations
`S`	S-statistic
`VarS`	Variance
`D`	n * (n - 1)/2
`tau`	Kendall’s tau
`Z`	Z-statistic
`p`	p-value
`AnalysisNote`	Notes about the analysis (see later description)
`prop.censored`	Proportion of observations that are censored
`prop.unique`	Proportion of unique observations
`No.censorlevels`	Number of unique left censor levels
`Median`	Data Median (based on data from ValuestoUseforMedian)
`Sen_VarS`	Variance used in the sen slope calculations
`AnnualSenSlope`	Annual Sen slope (attribute units/year)
`Intercept`	Predicted value at time t=t[1]
`Lci`	Lower confidence interval for annual Sen slope

`Uci`	Upper confidence interval for annual Sen slope
`AnalysisNote`	Note about why the dataset was not analysed, or warning about influence of censoring on sen slope.otherwise OK
`Sen_Probability`	Probability of a decreasing trend (mean) based on Sen method
`Sen_Probabilitymax`	Probability of a decreasing trend (max)
`Sen_Probabilitymin`	Probability of a decreasing trend (min)
`Percent.annual.change`	Percent annual change in Sen slope (value between 0 and 1)
`TrendCategory`	Trend direction at the 95% confidence level
`TrendDirection`	Face value trend direction

`MannKendall     OR     SeasonalKendall`

Description

The function MannKendall() performs a Mann Kendall test of correlation between the observations and time. The function SeasonalKendall() performs a seasonal Mann Kendall test of correlation between the observations and time, and should be used if the data are determined to be seasonal by SeasonalityTest(). The function also returns some outputs relating to the censoring in the data. These can be used as dagnostic tools to determine the reliability of the estimate.

Usage

```
MannKendall(x, ValuesToUse = "RawValue", Year = "Year",
HiCensor=FALSE)
```

OR

```
SeasonalKendall(x, ValuesToUse = "RawValue", Year = "Year",
HiCensor=FALSE)
```

Arguments

`x`	Input data frame must contain: myDate, Season,ValuesToUse
`ValuesToUse`	Select Column with data to use in the test
`Year`	Column name for Year data to be used
`HiCensor`	If TRUE the function finds the highest censoring limit (for left censored data) and sets any recorded values that are less than this value to the highest censoring limit and sets these as censored (at the highest censoring limit).

Value

A data frame with fields as follows

`nObs`	Number of observations
`S`	S-statistic
`VarS`	Variance
`D`	n * (n - 1)/2
`tau`	Kendall’s tau
`Z`	Z-statistic
`p`	p-value
`Probability`	Probability that the trend is decreasing
`AnalysisNote`	Note about why the dataset was not analysed, or warning about influence of censoring on sen slope.otherwise OK
`prop.censored`	Proportion of observations that are censored
`prop.unique`	Proportion of unique observations
`No.censorlevels`	Number of unique left censor levels

`SenSlope OR SeasonalSenSlope`

Description

The function SenSlope() performs a non-parametric regression (Sen's slope estimator, also known as the Theil–Sen estimator) of the observations versus time. The function SeasonalSenSlope() performs a seasonal version of the non-parametric regression (Sen's slope estimator) to the observations versus time, and should be used if the data are determined to be seasonal by SeasonalityTest(). If there are many censored values, the results of the Sen slope analysis should be treated with caution because the censored values are discarded. The Sen slope analysis in these cases may be inconsistent with the results of the Mann-Kendall or seasonal Kendall tests, which do use the censored values. The function returns the proportion of the observations that were censored to allow identification of Sen slopes that are likely to diverge strongly from the Mann-Kendall or seasonal Kendall tests.

Usage

```
SenSlope(x, ValuesToUse = "RawValue", ValuesToUseforMedian= "RawValue",
Year = "Year", HiCensor=FALSE, mymain=”My trend plot”,…)
```

OR

```
SeaonalSenSlope(x,  ValuesToUse  =  "RawValue",  ValuesToUseforMedian=
"RawValue", Year = "Year", HiCensor=FALSE, mymain=”My trend plot”,…)
```

Arguments

`x`	Input data frame must contain: myDate, Season,ValuesToUse,CenType
`ValuesToUse`	Select Column with data to use in the test
`ValuesToUseforMedian`	Default is to use RawValues. Alternatives are to name another column (e.g., ImputedData),
`Year`	Column name for Year data to be used
`HiCensor`	If TRUE the function finds the highest censoring limit (for left censored data) and sets any recorded values that are less than this value to the highest censoring limit and sets these as censored (at the highest censoring limit).
`doPlot`	Produce a plot if TRUE. Returns a plot object if “doGGplot”
`mymain`	Optional title for plot. Default uses npID*sID if they exist
`…`	Additional variables for plotting function:
`Probability`	Probability derived from Mann Kendall test, just for displaying on the plot
`legend.pos`	Position of legends (default is top, alternative is bottom)

Value

A data frame and (optionally) a plot. If doPlot=”doGGPlot”, then a list is returned, with the dataframe as the first item and the plot as the second item. Data frame fields as follows

`Median`	Data Median (based on data from ValuestoUseforMedian)

`Sen_VarS`	Variance used in the sen slope calculations
`AnnualSenSlope`	Annual Sen slope (attribute units/year)
`Intercept`	Predicted value at time t=t[1]
`Lci`	Lower confidence interval for annual Sen slope
`Uci`	Upper confidence interval for annual Sen slope
`AnalysisNote`	Note about why the dataset was not analysed, or warning about influence of censoring on sen slope.otherwise OK
`Sen_Probability`	Probability of a decreasing trend (mean) based on Sen method
`Sen_Probabilitymax`	Probability of a decreasing trend (max)
`Sen_Probabilitymin`	Probability of a decreasing trend (min)
`Percent.annual.change`	Percent annual change in Sen slope (value between 0 and 1)

`AdjustValues`

Description

Performs an adjustment of the data based on covariate data (for example flow if the data represents river concentrations). The function fits one or more models to the observation versus covariate relationship. The user needs to consider which of these models is the most appropriate basis for adjustment.

Usage

```
AdjustValues(x, method = c("Gam", "LogLog", "LOESS"), ValuesToAdjust =
"RawValue", Covariate = "Flow", Span = c(0.5, 0.75), doPlot = T)
```

Arguments

`x`	Dataframe or vector containing: myDate, ValuesToAdjust,Covariate
`method`	Method to fit relationship between the observations and covariate. One or more of `Gam, LogLog, LOESS),`
`ValuesToAdjust`	Column name of observations to perform adjustment on
`Covariate`	Column name of the covariate to be used
`Span`	Vector of spans to be tested for LOESS models
`doPlot`	Produce a plot if TRUE, return GGplot object as a list if “doGGPlot”
`plotpval`	If TRUE Include R^2 and pvalues in plot legend (default = FALSE)
`plotloglog`	If TRUE include a second Q-C plot in log-log space (default=FALSE). Does not apply wen returning the GGplot object
`SiteName`	String describing site name. Only used for the GGplot objects
…	Additional plotting variables to pass to the base plot

Value

A data frame and (optionally) a plot. If doPlot=”doGGPlot”, then a list is returned, with the dataframe as the first item and the plot as the second item. Data frame fields as follows:

`myDate`	Observation date
…	One column of adjusted observations (residuals) for each method selected
… `_R`	One column correlation coefficients (R) for each method selected
… `_p`	One column p-values for each method selected

`AnalysePIT`

Description

Utilises the probability that the true trend was decreasing to provide a probabilistic estimate of the proportion of improving trends (PIT) for a given variable across a number of sites.

Usage

```
AnalysePIT(x, myRound = 1, Reverse = c("CLAR","MCI"))
```

Arguments

`x`	Dataframe or vector containing: npID and Probability
`myRound`	Number of decimal points to include in the outputs
`Reverse`	List of npID variables for which an increasing trends indicates improvement

Value

A data frame with fields as follows

`npID`	Variable name
`n.Sites`	Number of sites
`PIT`	Proportion of Improving Trends statistic
`sdPIT`	The standard deviation of the PIT statistic

`ImprovementConfCat`

Description

Assigns a categorical description of the probability the trend is improving based on the values outlined in Table 1.

Usage

```
ImprovementConfCat(x, Reverse = c("CLAR","MCI"))
```

Arguments

`x`	Dataframe or vector containing: npID and Probability
`Reverse`	List of npID variables for which an increasing trends indicates improvement

Value

A vector of confidence categories (as per table1) as factors.