The LWP-Trends library

March 2025

Title LWP-Trends library Version 2502: ***LWPTrends_v2502.r***

Date 10 March 2025

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

Contacts: caroline@lwp.nz

Description Tools for analysing water quality trends

Requires: packages: plyr, NADA, lubridate, ggplot2 and gam

Contents:

Reference:

Fraser, C. and Snelder, T. (2025) *“The LWP-Trends Library; v2502 March 2025”,* LWP Ltd Report, p53

Disclaimer

Software downloaded from the(LWP Ltd website (or provided by direct communication with an LWP Ltd 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 Ltd 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 Ltd may make changes to the software or documentation made available on its website.
2. may be out of date, and LWP Ltd makes no commitment to update such materials.

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

In no event shall LWP Ltd 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 Ltd 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 Ltd 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 Ltd or from the LWP Ltd website shall create any warranty for the software.

Release notes: version 2502

1. Removed the use of HiCensor filter. The Kendall test as implemented did not require the filter and the Sen slope calculation has adopted the same treatment of censored values as for the Kendall test.
2. Removed previous trend aggregation functions: AnalysePIT, MonteCarloPIT and FaceValueCounter. Added the new trend aggregation function getTau and associated plotting function PlotRegionalConfDir.
3. Added a new wrapper function, doMyTrends_2502, that will assess seasonality, select an appropriate season, and run the appropriate seasonal or non-seasonal trend functions.
4. Upgrade of InpsectData to new function InspectTrendData. As well as providing summary plots of data, also determines the minimum possible analysis frequency (TimeIncr) that meets user defined minimum data requirements.
5. Options now to define the analysis time increment (TimeIncr) and the season (SeasIncr) separately.
6. Options to have an appropriate "season" to be selected by the new function "GetSeason(), which assesses which season to use and whether a seasonal trend analysis should be performed. Seasons that do not meet further data requirements for the trend assessment that would lead to a "not analysed" result (i.e., insufficient unique values within a season) are excluded at this stage. If that data are seasonal, but no seasons will meet minimum data requirements, the data will labelled as non-seasonal and trends will be evaluated with the non-seasonal trend assessment.
7. Adjustments to seasonal trend assessments to account for possibility that the seasonal increments are larger than the time increments. In the calculation of the S and the variance of S, observations in the same season-year are treated as ties in time. In the calculation of the Sen Slope, the inter-observation slopes are excluded for pairs of observations that are in the same season-year.
8. All options for base R plots have been removed, retaining only ggplot output options.
9. The median of observations used to normalise the Sen Slope to a percentage has been changed to be the median over the observations AFTER the mid or median within time increment calculation has been performed (rather than using ALL observations).
10. The output data frames for the SeasonalSenSlope and SenSlope, also now include strings to specify the analysis time increment (TimeIncr) and the time increment for the season (Season).
11. MannKendall and SeasonalKendall functions now output nObs as the total number of observations (i.e., including duplicates within time increments) and nTimeIncr as the number of time increments with observations.
12. Updates to "GetMoreDateInfo" so that bimonths, quarters, biannual time increments can be variable depending on the "first month" (i.e. quarters where first month is 1 would be "J-F-M,…", but where first month is 2 would be "F-M-A"). New biannual (twice yearly) time increment option included. Also makes an estimate of the appropriate TimeIncr for the analysis.
13. New function ShiftTempDate () included to process monthly timeseries data to identify observations that probably are supposed to represent a sample from the previous or following months, and assign them time metadata (i.e., time increment options) associated with the previous or following month if there is more than one observation in the month. This function is called from within GetMoreDateInfo. The default is to undertake the date shift, but it can also be turned off by setting FindDateShifts=FALSE.
14. For consistency all plot title options are specified by the argument mymain (rather than SiteName or other). Plot titles also include a second row indicating the time increment and season time increment for the analysis.
15. Updated RemoveAlphaDetec()t function so it reads the entire dataset and returns that dataset with the required additional columns (rather than having to bind outputs onto original dataset after calling the function).
16. GetMoreDateInfo(); previous versions had an error when start month was not specified as either 1 or 7. Yearshift (an internal variable) has also been adjusted so that CustomYear is named after the calendar year associated with the last day of the custom year (previously this was the first)
17. AdjustValues() function. Factorise "Method" to improve plotting.
18. MedianForSeason() function is now called ValueForTimeIncr() (generalised the name as the function can evaluate the median or mid-point for either a season or a time increment).
19. Updates to GGplotTrend to ensure that the data type is coloured correctly in the plots. Updates to title.

20. Added new generic categorical assignment function: AssignConfCat(). Replaces previous version DecreasingConfCat(). Can now provide confidence categories for trend direction, confidence trend is improving and confidence trend is decreasing. Two options for the number of confidence categories are provided.
21. Updated string list of months to reflect a change in base R which now abbreviates September as Sept (rather than Sep).

HISTORIC Release notes: version 2102

1. There was an error in line 1854 of the AnalysePIT function in the calculation of the variance of the PIT statistic (C was used instead of C_d) that led to an overestimation of the variance. This has been corrected
2. We have changed the "Proportion of Improving Trends" (PIT) to now analyse the "Proportion of decreasing trends" (Pd). The functions "AnalysePIT" and "MonteCarloPIT" are replaced with "AnalysePd" and "MonteCarloPd". The difference is that the function no longer "reverses" variables for which decreasing trends are associated with degradation (i.e., Clarity and MCI). We did this to allow the function to be more flexible, particular to accommodate variables where no particular direction of change is associated with improvement or degradation.
3. We have changed the function "ImprovingConfCat" to now be "DecreasingConfCat" to remove the subjective decision required to decide whether any given variable improves or degrades with decreasing values.
4. There was an error in the "GetKendall" and "SeasonalKendall" functions at lines 1512 and 1823, respectively, where the special case of Z==0 was assigned a p value of 0.5, which then incorrectly assigned C and C_d. This has been corrected so that the p value assigned where Z==0 is 1.

HISTORIC Release notes: version 2101

1. Updates to description of trends methodology to align with recent trend assessment guidance (Snelder et al. 2021). IN particular we no longer use "Probability of trend direction", rather "Confidence in trend direction".
2. Sen slopes derived from flow adjusted values now require an argument to be included so that the censored values are not adjusted (by 0.5 for values below detection limit or 1.1 for values above the reporting limit). This adjustment is handled as part of the flow adjustment function and does not need to be repeated.
3. An option has been added to allow the observation closest to the middle of the season to be used as the season observation (previously the only option was to use the median value of the season). This is a suitable option to use when the sampling frequency has changed throughout time, and the season is defined as the coarser sampling frequency (Helsel 2020).
4. The confidence intervals for the Sen slope are now reported as "Sen_Lci" and "Sen_Uci", rather than "Uci" and "Lci" (to avoid ambiguity)
5. Option for HiCensor filter to now include a numeric input for a user defined hicensor value (rather than previously which took the highest detection limit value as the high censor value).
6. Flow adjustment functions now allow for a high censor limit to be applied at the flow adjustment step. New instructions about how to apply the high censor limit when using flow adjustment are described in the manual.

HISTORIC 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 using non-parametric methods based on the Mann-Kendall correlation coefficient and the Theil–Sen estimator.

An update to the methods description and output was made in April 2021 in order to align the methodology description and terminology to recent New Zealand guidelines for trend assessment (Snelder et al. 2021). We recommend the reader refers to these guidelines for a more detailed description of the methods; only a summarised description is provided within this help document.

Updates in this release are to provide some corrections, incorporate some additional utilities, as well as to:

1. Provide new functions to evaluate aggregate trend assessments, following Snelder et al. (2022). The method assesses the modal direction of individual trends at multiple sites (i.e., the aggregate trend direction) and assesses the confidence associated with this determination while taking into account the spatial autocorrelation between sites. Previous functions that evaluated aggregate trends are superseded and removed from the library.

2. Allow for separate specification of the time increment of analysis (TimeIncr) and the season (Season) of the analysis. Previous versions required the time increment and season to be the same, which was specified in the data by a variable called Season. This change was for two reasons. First, the terminology was confusing when dealing with non-seasonal trend assessments. Second, the flexibility to assign TimeIncr and Season separately should generally increase the statistical power of the trend assessments by simultaneously reducing variability associated with seasonality but retaining as many observations as possible.

3. Removal of the high censor filter. The inclusion of the high censor filter had been a pragmatic choice following recommendations of Helsel et al. (2020. p. 357) to account for changes in censor level. However, it is unnecessary to apply this filter to the evaluation of Kendall's S as implemented in LWPTrends, as the formulation already robustly accounts for varying censor levels (following cenken() from the NADA package and implemented in LWPTrends by the GetKendal() function).

4. The calculation of the Sen Slope now accounts for varying censor levels in the same manner as the GetKendal(), ensuring that slopes are treated as ties between higher left censored data and lower non-censored data (or conversely lower right censored data and higher non-censored data). This removes the need to have a high censor filter. More details are provided in section 4.2.

5. Make adjustments to date metadata for sampling dates that should be associated with a previous or following month. This occurs by default in the function GetMoreDateInfo(), but can be turned off by setting the argument FindDateShifts=FALSE. Sampling might, for example, be consistently carried out towards the end of the month. However, on occasion, a sample occasion associated with a monthly monitoring programme for a specific month might occur on the first day of the following month, leading to two samples in the following month, and none in the previous month. If this happens frequently, the

site may not comply with filtering rules (i.e., minimum data requirements for a trend assessment to be judged robust). However, by reassigning observations at the end of the month to the following month there would be sufficient approximately monthly observations. We have added an option to make adjustments to the date metadata in these situations; the method is described in section 3.2.

The changes listed above have relatively minor implications for the evaluated trend confidence and trend rates. The more significant effect is that fewer analyses will have insufficient data and therefore fewer analyses will be categorised as “not analysed”.

2 Trend Analysis Method

The purpose of trend assessment is to evaluate the direction (i.e., increasing or decreasing) and rate of the change in the central tendency of the observed water quality values over the period of analysis (i.e., the trend). Because the observations represent samples of the water quality over the period of analysis, there is uncertainty about the conclusions drawn from their analysis. Therefore, statistical models are used to determine the direction and rate of the trend and to evaluate the uncertainty of these determinations.

2.1 Trend direction assessment

The trend direction and the confidence in the trend direction were evaluated using either the Mann Kendall assessment or the Seasonal Kendall assessment. Although the non-parametric Sen slope regression also provides information about trend direction and its confidence, the Mann Kendall assessment is recommended, rather than Sen slope regression, because the former more robustly handles censored values. However, Sen slope regression is the recommended method for assessing the trend rate (see Section 2.2).

The Mann Kendall assessment requires no *a priori* assumptions about the distribution of the data but does require that the observations are randomly sampled and independent (no serial correlation) and that there is a sample size of ≥ 8. Both the Mann Kendall and Seasonal Kendall assessments are based on calculating the Kendall *S* statistic, which is explained diagrammatically in Figure 4.

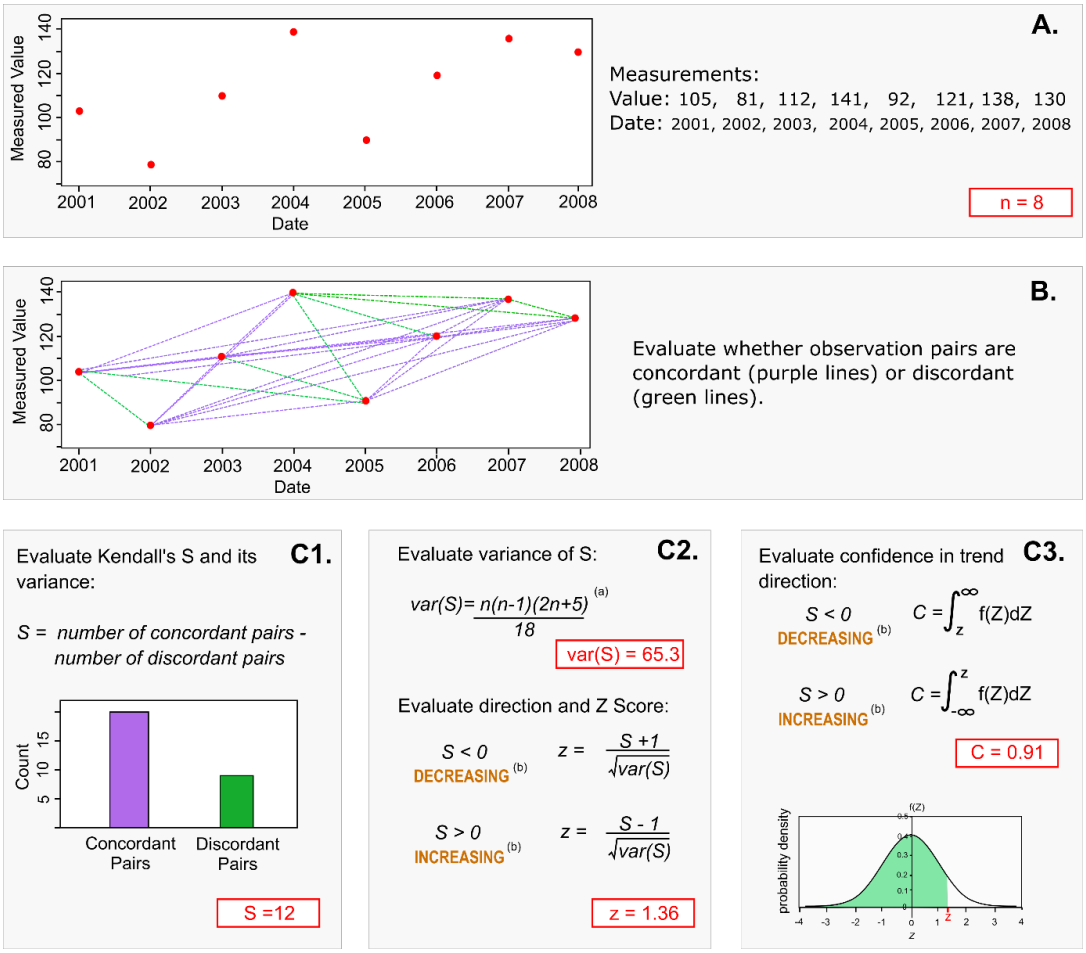


Figure 1. Pictogram of the steps taken in the trend direction assessment to calculate the Kendall S statistic and its confidence in trend direction. Notes: [a] the calculation of the variance in S has some adjustments to account for ties (numerically equal values, or ties in time) and censored values. Details of these adjustments can be found in (Gilbert 1987; Helsel 2005, 2012). [b] There is a third alternative, where S=0. In this case C is 0.5, and the trend direction is classified as "indeterminate". Values of S equal to -1 or 1 will also result in a Z value of 0, and a C value of 0.5 and the trend direction is similarly classified as "indeterminate".

The Kendall *S* statistic is calculated by first evaluating the differences between all pairs of water quality observations (Figure 1, A and B). Positive differences are termed 'concordant' (i.e., the observations increase with increasing time) and negative differences are termed 'discordant' (i.e., the observations decrease with increasing time). Pairs of observations that are tied in value or tied in time[1] (i.e., have the same Season-Year) are assigned differences of zero (Gilbert, 1987, p. 218). The Kendall *S* statistic is the number of concordant pairs minus the number of discordant pairs (Figure 1, C1). The water quality trend direction is indicated by the sign of *S* with a positive or negative sign indicating an increasing or decreasing trend, respectively (Figure 1, C2).

The seasonal version of the Kendall *S* statistic *S* is calculated in two steps. First, for each season, the *S* statistic is calculated in the same manner as shown in Figure 1, but for data pertaining to

[1] Accounting for ties in time is an addition to the code since version 2502. This addition is to accommodate seasonal analyses where the Season time increment is larger than the data TimeIncr (time increment).

observations in each individual season. Second, *S* is the sum of values over all seasons ($S = \sum_1^n S_i$), where S_i is the number of concordant pairs minus the number of discordant pairs in the i^{th} season and n is the number of seasons. The variance of *S* is calculated for each season and then summed over all seasons.

The sign (i.e., + or -) of the *S* statistic calculated from the sample represents the best estimate of the population trend direction but is uncertain (i.e., the direction of the population trend cannot be known with certainty). A continuous measure of confidence in the assessed trend direction can be determined based on the posterior probability distribution of S, the true (i.e., population) difference in concordant and discordant pairs (Snelder et al., 2022). The posterior probability distribution of *S* is given by a normal distribution with mean of *S* and variance of *var(S)*. The confidence in assessed trend direction can be evaluated as the proportion of the posterior probability distribution that has the same sign as S.

In practice. confidence can be calculated by first transforming the value of $S = 0$ on the posterior probability distribution into a standard normal deviate, Z (panel C2). *C* is then calculated as area under the standard normal distribution to the left (Z>0) or right (Z<0) of the value of Z, using the quantile function for the normal distribution.

The value C can be interpreted as the probability that the sign of the calculated value of *S* indicates the direction of the population trend (i.e., that the calculated trend direction is correct). The value C ranges between 0.5, indicating the sign of *S* is equally likely to be in the opposite direction to that indicated by the true trend, to 1, indicating complete confidence that the sign of *S* is the same as the true trend.

As the size of the sample (i.e., the number of observations) increases, confidence in the trend direction increases. When the sample size is very large, C can be high, even if the trend rate is very low. It is important therefore that C is interpreted correctly as the confidence in direction and not as the importance of the trend. As stated at the beginning of this section; both trend direction and the trend rate are relevant and important aspects of a trend assessment.

2.2 Assessment of trend rate

The method used to assess trend rate is based on non-parametric Sen slope regressions of water quality observations against time. The Sen slope estimator (SSE; Hirsch *et al.,* 1982) is the slope parameter of a non-parametric regression. SSE 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; *Figure 2*).

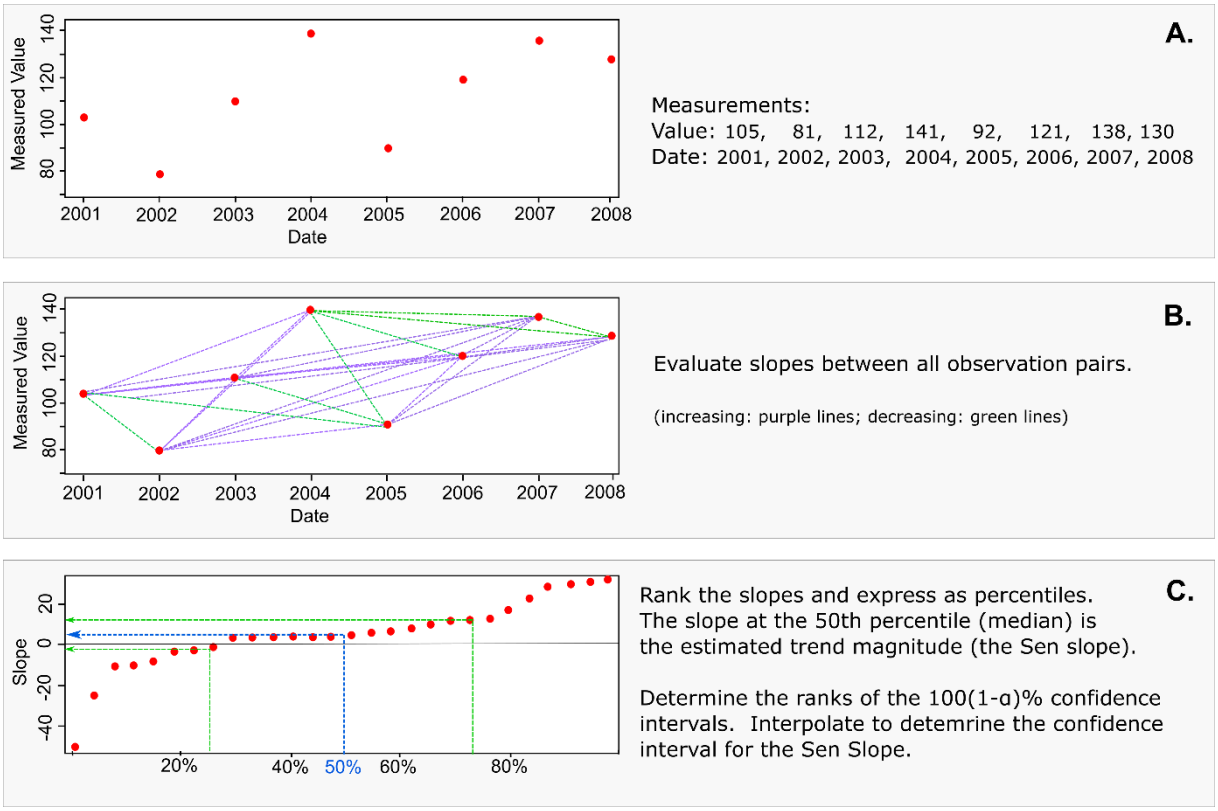


Figure 2. Pictogram of the calculation of the Sen slope, which is used to characterise trend rate.

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 but can also be defined (as of release version 2502) as whole multiples of the sampling interval (i.e., monthly sampling interval and quarterly season). The seasonal Sen slope estimator (SSSE) is calculated in two steps. First, for each season, the median of all possible inter-observation slopes is calculated in same manner as shown in *Figure 2* but for data pertaining to observations in each individual season. When the time increment is smaller than the Season time increment (i.e. there is generally more than one observation per Season by Year) the inter-observation slopes are excluded for pairs of observations that are in the same season-year, but all slopes in the season of one year are compared with all slopes in the season of the other years (Gilbert, 1987, p. 218). Second, SSSE is the median of the seasonal values.

Uncertainty in the assessed trend rate is evaluated following a methodology outlined in Helsel and Hirsch (2002). To calculate the 100(1-α)% two-sided symmetrical confidence interval about the fitted slope parameter, the ranks of the upper and lower confidence limits are determined, and the slopes associated with these observations are applied as the confidence intervals.

2.3 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 (Helsel, 2005) and Statistics for censored environmental data using MINITAB and R (Helsel, 2012).

Censored values in the data used to calculate Kendall's S and its p-value are robustly handled in the manner recommended by Helsel (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. This approach is implemented in LWPTrends using source code from the NADA package. The method is robust to changes in detection limit over time.

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

When calculating Sen slopes, 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 LWPTrends library, the censored data entries are substituted with 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 substituted values). 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.

In previous versions of LWPTrends, the influence of changes in detection limits over time on the Sen slope were handled through a high censor filter. Briefly, this filter replaced all observations below the highest detection limit (or some user specified value) to the highest detection limit and marked these observations as censored. We now implement a different approach that follows the logic used in the calculation of Kendall's S. Inter-observation slopes are considered to be ties and set to zero, regardless of their values, when: (1) both observations were either left or right censored, (2) when one observation is left censored and larger than the other non-censored observation; (3) when one observation is right censored and smaller than the other non-censored observation. This approach has the advantage that the user decisions around if and how to implement a high censor filter are removed.

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 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).

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, 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 "%Y-%m-%d"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 to the data 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 time increments that start in July and output a “custom year” (or water-year) that starts in July, the default is that the year starts in January). The function also provides additional columns (date metadata) describing the months, 2-months (BiMonth), quarters, twice annual (BiAnn), and

years – these are required for the analysis, and are common choices for the time increment or season of the dataset. The additional time increment options are all started at the start month and are labelled based on the start and end month of the increment (i.e., quarters might be: Jan.to.Mar, Apr.to.Jun; Jul.to.Sept; Oct.to.Dec). In the 2502 release, this function now generates a default estimate of the time increment of the data (TimeIncr), based on a user defined quantile for the spacing between observations. The 2502 version also includes a function that looks for months that have two or more observations in combination with a month before or after that month that is missing an observation. Where the previous month has a missing observation, the first observation is assigned metadata associated with the previous month, and where the following month has a missing observation, the last observation is assigned metadata associated with the following month. The observations retain the original date for use in the SenSlope evaluation. The date metadata columns in the output are factors (e.g. BiMonth, Qtr, BiAnn). 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)
```

```
  Month npid      sdate Value   finalQ     myDate Year CustomYear    BiMonth        Qtr       BiAnn TimeIncr
1   Apr   TP 1997-04-16 0.052 1.366466 1997-04-16 1997       1997 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr
2   Apr   TP 1999-04-22 0.046 1.331835 1999-04-22 1999       1999 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr
3   Apr   TP 2004-04-14  0.05 1.263050 2004-04-14 2004       2004 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr
4   Apr   TP 2014-04-09  0.06 1.006173 2014-04-09 2014       2014 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr
5   Apr   TP 2005-04-13 0.052 1.274132 2005-04-13 2005       2005 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr
6   Apr   TP 2002-04-17  0.03 1.049622 2002-04-17 2002       2002 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr
```

3.3 Time increments

The 2502 release has separated time increments from seasons. Now, seasons are only used as the factor to evaluate seasonality and to perform a seasonal trend assessment. Seasons can now be coarser (but a multiple of) the time increment. See 4.1.2 for details about selecting an appropriate Season. The default time increment is defined by the GetMoreDateInfo function, but the user can override this, e.g.,:

```
WQData_Ex1b$TimeIncr<-WQData_Ex1b$Month
```

Note, the time-increment must be a factor. Any user-defined time-increment can be analysed.

3.4 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:

```
WQData_Ex1b <- RemoveAlphaDetect(WQData_Ex1b,ColToUse="Value")
```

This output is a data frame with the original dataset plus three additional columns. The columns additional 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).

The modified WQData_Ex1b data frame is shown in

```
Month npid      sdate Value   finalQ     myDate Year Custo
mYear    BiMonth    Qtr      BiAnn  TimeIncr  RawValue Censored CenType
Apr   TP 1997-04-16 0.052 1.366466 1997-04-16 1997       1997 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr    0.052    FALSE     not
Apr   TP 1999-04-22 0.046 1.331835 1999-04-22 1999       1999 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr    0.046    FALSE     not
Apr   TP 2004-04-14  0.05 1.263050 2004-04-14 2004       2004 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr    0.050    FALSE     not
Apr   TP 2014-04-09  0.06 1.006173 2014-04-09 2014       2014 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr    0.060    FALSE     not
Apr   TP 2005-04-13 0.052 1.274132 2005-04-13 2005       2005 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr    0.052    FALSE     not
Apr   TP 2002-04-17  0.03 1.049622 2002-04-17 2002       2002 Mar.to.Apr Apr.to.Jun Jan.to.Jun      Apr    0.030    FALSE     not
```

Figure 4.

Month	npid	sdate	Value	finalQ	myDate	Year	CustomYear	BiMonth	Qtr	BiAnn	TimeIncr	RawValue	Censored	CenType
Apr	TP	1997-04-16	0.052	1.366466	1997-04-16	1997	1997	Mar.to.Apr	Apr.to.Jun	Jan.to.Jun	Apr	0.052	FALSE	not
Apr	TP	1999-04-22	0.046	1.331835	1999-04-22	1999	1999	Mar.to.Apr	Apr.to.Jun	Jan.to.Jun	Apr	0.046	FALSE	not
Apr	TP	2004-04-14	0.05	1.263050	2004-04-14	2004	2004	Mar.to.Apr	Apr.to.Jun	Jan.to.Jun	Apr	0.050	FALSE	not
Apr	TP	2014-04-09	0.06	1.006173	2014-04-09	2014	2014	Mar.to.Apr	Apr.to.Jun	Jan.to.Jun	Apr	0.060	FALSE	not
Apr	TP	2005-04-13	0.052	1.274132	2005-04-13	2005	2005	Mar.to.Apr	Apr.to.Jun	Jan.to.Jun	Apr	0.052	FALSE	not
Apr	TP	2002-04-17	0.03	1.049622	2002-04-17	2002	2002	Mar.to.Apr	Apr.to.Jun	Jan.to.Jun	Apr	0.030	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 trend data

The function InspectTrendData() assists with inspection of the data and the selection of an appropriate time increment, given some minimum data requirements. The function also evaluates whether the data are . It is assumed that the trend analysis is for a specific time-period that is defined by the arguments Year (specifies whether to use a calendar or custom year), TrendPeriod (number of years) and EndYear. The function outputs a dataframe that cuts the input data down to the specified time period and adds on a column TimeIncr, which is the highest observation frequency that meets the minimum data requirements (which are specified by propIncrTol and propYearTol, the proportion of time increment x years and proportion of years which must have observations).

```
> Out1a<-InspectTrendData(WQData_Ex1a, EndYear = 2017, FlowtoUse="finalQ",
+                         ReturnALLincr=TRUE,do.plot = TRUE,mymain="Example 1a")
```

```
> head(Out1a[[1]])
  Month npid      sdate  Value     finalQ     myDate Year    BiMonth        Qtr      BiAnn RawValue Censored CenType      Flow TimeIncr
1   Apr  DRP 1997-04-16  0.016 31.680979 1997-04-16 1997 Mar.to.Apr Apr.to.Jun Jan.to.Jun   0.0160    FALSE     not 31.680979      Apr
2   Apr  DRP 2010-04-12 0.0095 11.431822 2010-04-12 2010 Mar.to.Apr Apr.to.Jun Jan.to.Jun   0.0095    FALSE     not 11.431822      Apr
3   Apr  DRP 2009-04-06 0.0051  4.264485 2009-04-06 2009 Mar.to.Apr Apr.to.Jun Jan.to.Jun   0.0051    FALSE     not  4.264485      Apr
4   Apr  DRP 1998-04-08  0.015 35.104542 1998-04-08 1998 Mar.to.Apr Apr.to.Jun Jan.to.Jun   0.0150    FALSE     not 35.104542      Apr
5   Apr  DRP 2012-04-16 <0.004  2.631876 2012-04-16 2012 Mar.to.Apr Apr.to.Jun Jan.to.Jun   0.0040     TRUE      lt  2.631876      Apr
6   Apr  DRP 2008-04-14  0.006  2.402878 2008-04-14 2008 Mar.to.Apr Apr.to.Jun Jan.to.Jun   0.0060    FALSE     not  2.402878      Apr
```

The function also optionally outputs a data frame (when ReturnALLIncr=TRUE) that identifies the trend period length (TrendPeriodL), total number of observations (nobs), number of years with observations (nYear), proportion of years with observations (propYear), the proportion of observations that are censored (propCen), the number of unique left censored values (nCenLevelsLT), the number of unique right censored values (nCenLevelsGT) and the number of observations with flow data (nFlow). For each possible time increment (monthly, bimonthly, quarterly, biannual and annual) the total number of time increment x years with observations (nIncrYear) is provided, as well as the proportion of all possible time increment x years for the trend period with observations (propIncrYear). Based on the user defined minimum data requirements, each possible time increment is assigned a logical value to DataOK, to specify whether the time increment meets the minimum data requirements. The time increment used in the first dataframe output is designated as the highest frequency time increment where DataOK is TRUE.

```
> Out1a[[2]]
     Incr TrendPeriodL nobs nYear propYear nIncrYear propIncrYear   propCen nCenLevelsLT nCenLevelsGT nFlow DataOK
1   Month          21  245    21        1       244    0.9682540 0.1632653           2           0   245   TRUE
2 BiMonth          21  245    21        1       123    0.9761905 0.1632653           2           0   245   TRUE
3     Qtr          21  245    21        1        82    0.9761905 0.1632653           2           0   245   TRUE
4   BiAnn          21  245    21        1        42    1.0000000 0.1632653           2           0   245   TRUE
5    Year          21  245    21        1        21    1.0000000 0.1632653           2           0   245   TRUE
```

The function also optionally outputs four graphs: a time series plot; a matrix plot of the data; a matrix plot of the censoring; and a matrix plot of the number of observations per month. Matrix plots are always shown with months on the x-axis and calendar years of the y-axis. This can assist with visualising the sampling frequency and any changes in frequency over time.

All four plots can be shown together with the following command:

```
ggarrange(Out1a[[3]][[1]],ggarrange(Out1a[[3]][[2]],Out1a[[3]][[3]],Out1a[[3]][[4]],nr
ow=1,align="h"),nrow=2)
```

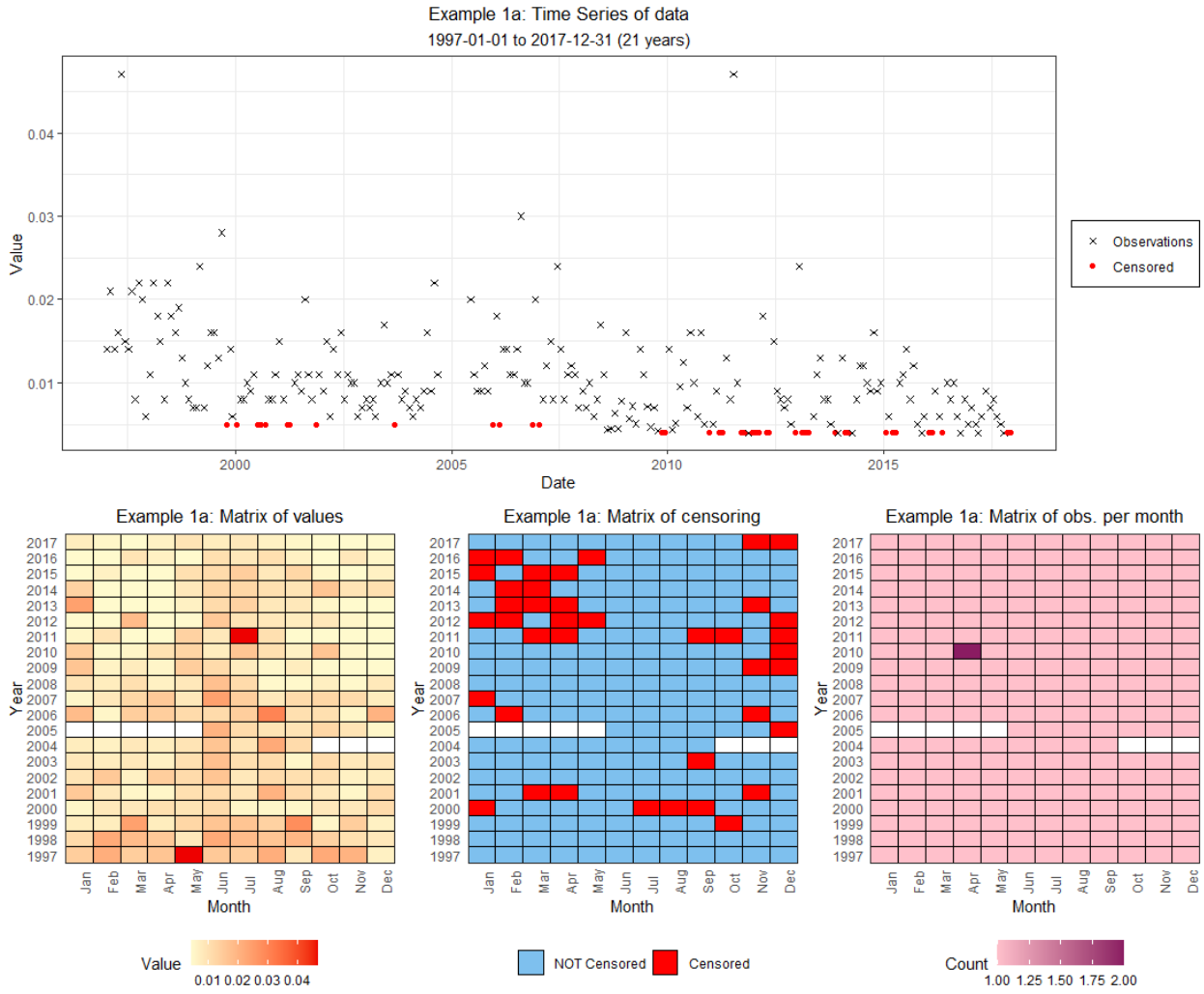


Figure 5. Plot outputs from InspectTrendData().

4.1.2 Season and Seasonality Test

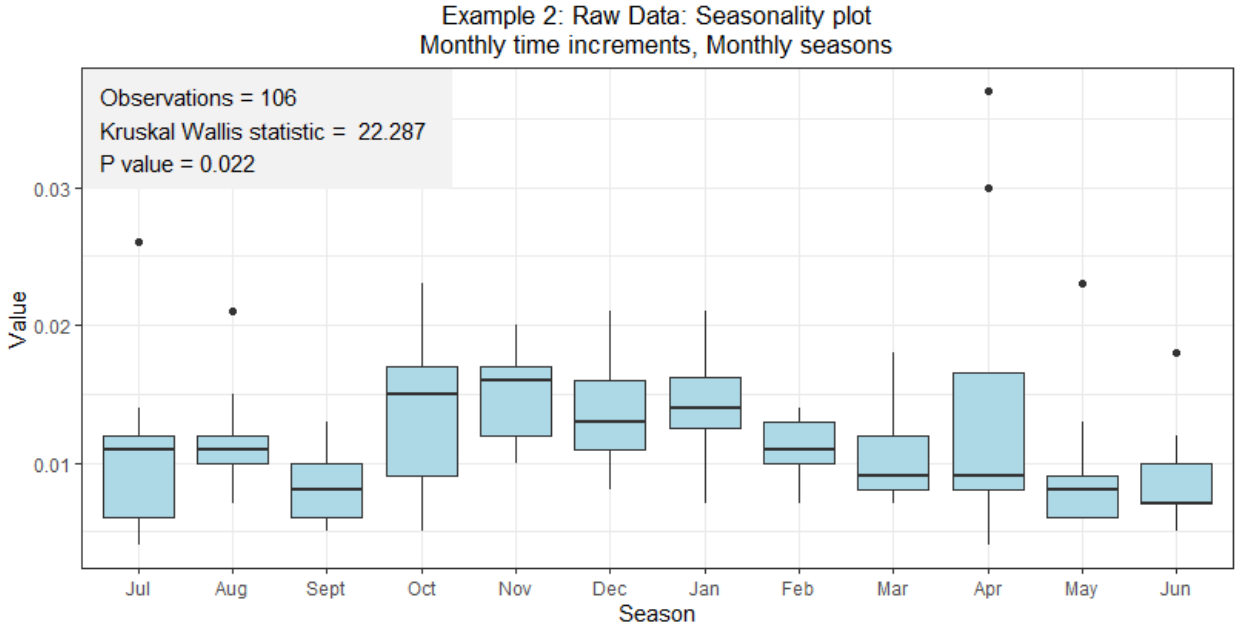
The function SeasonalityTest() tests if the data are seasonal. If the data are seasonal (*p*-Value<0.05), the trend analysis should be seasonal (i.e., the Kendall test should be the seasonal Kendall test and the Sen slope should be a seasonal Sen slope). The SeasonalityTest() function performs a Kruskal Wallis test (non-parametric ANOVA) on the observations using season as the explanatory (categorical) variable. A test is also performed to check whether the season would meet the minimum data requirements for the trend assessment (i.e., minimum number of samples

and unique values per season). If these requirements are not met, the data is assessed as non-seasonal, as a trend assessment with this season would return a “Not Analysed” result. The consequence of analysing as non-seasonal will generally be to have lower confidence in direction (and large Sen slope confidence intervals) than had the analysis been seasonal.

The SeasonalityTest() function has an option to return a plot summary of the seasonality test, in which case it outputs a list of:

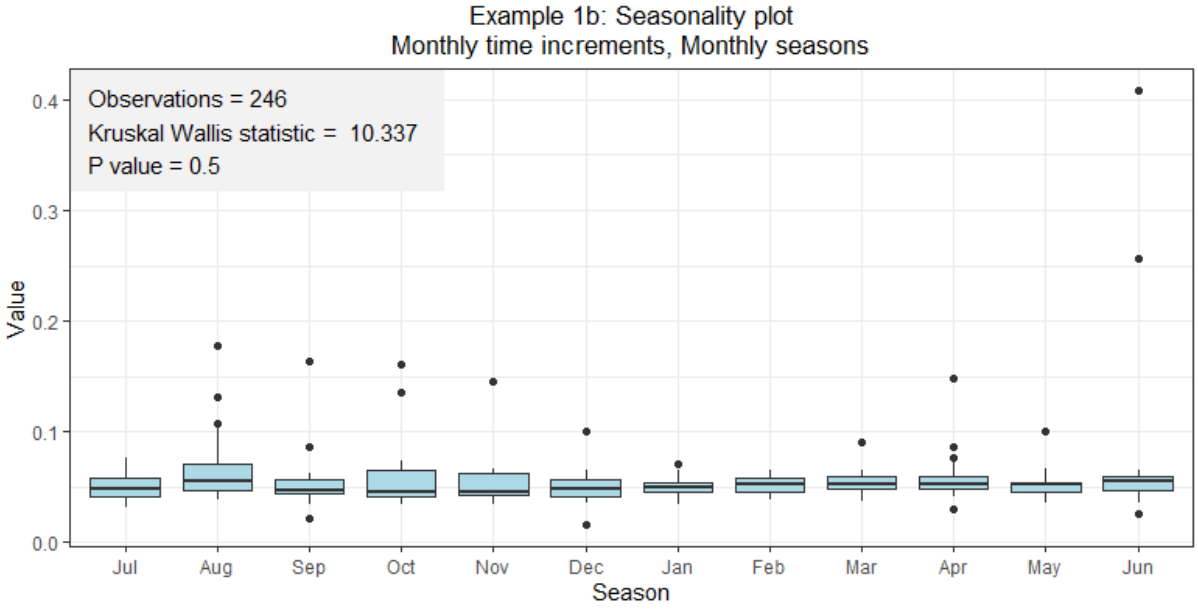
1. A summary of the KW test outputs for the selected season
2. A box plot of the data grouped by season (e.g., Figure 6 and Figure 7).

If do.plot is FALSE, SeasonalityTest() only returns a dataframe of the KW test outputs.



	Observations	KWstat	pvalue	SeasNote	TimeIncr	Season
Kruskal-Wallis chi-squared	106	22.28694	0.02224216	ok	Monthly	Monthly

Figure 6. Plot and data frame output from SeasonalityTest() for example seasonal data set.




```
 Observations   KWstat    pvalue          SeasNote TimeIncr  Season
  246           10.33715 0.5003391       ok       Monthly   Monthly
```

Figure 7. Plot and data frame output from SeasonalityTest() for example non-seasonal dataset.

As of version 2502, Season may be defined separately from the time increment (TimeIncr) for the analysis. A new function GetSeason() has been added (as of v2502) that selects an appropriate season by evaluating the Kruskall Wallis test to determine which potential season explains the most variability in the data. When performing this selection, the possible options for Season are the selected time increment, and any other increments (i.e., month, BiMonth, Qtr,BiAnn) that are whole multiples of the time increment (these are defined internally). The selected season is the time increment that has a p value <0.05 or, in the case that more than season increment meets this threshold, has the largest KW statistic. The function is a wrapper for SeasonalityTest(), and can return a dataframe of the data with a “Season” column added, as well as the outputs available from SeasonalityTest(). When data are not seasonal, “Season” is set to the time increment. While Season is not required for subsequent non-seasonal analysis, it ensures the same data structure across sites x variables. The function also has an option (ReturnALLincr=TRUE) to print out the KW test results for all Seasons tested:

```
      .id Observations    KWstat      pvalue SeasNote TimeIncr     Season
1   Month          106 22.286941 0.022242160       ok  Monthly    Monthly
2 BiMonth          106 15.890004 0.007165236       ok  Monthly Bi-monthly
3     Qtr          106 14.275771 0.002552829       ok  Monthly  Quarterly
4   BiAnn          106  1.597295 0.206286940       ok  Monthly  Bi-annual
```

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. P is the two-tail *p*-value from the Mann Kendall test, C is the confidence in the trend direction, and Cd is the confidence that the trend is decreasing.

```
Trend1b<-NonSeasonalTrendAnalysis(WQData_Ex1b,mymain="Ex 1b Raw Trend",
Year="CustomYear",do.plot=T);Trend1b[[1]]
```

```
nObs                  246
nTimeIncr             246
S                     9761
VarS                  1662221
D                     30135
tau                   0.3239091
Z                     7.570167
p                     3.72746e-14
C                     1
Cd                    1.86373e-14
prop.censored         0
prop.unique           0.2560976
no.censorlevels       0
TimeIncr              Monthly
SeasIncr              NonSeasonal
Median                0.05
AnnualSenSlope        0.0008910951
Sen_Lci               0.0007271339
Sen_Uci               0.001053087
AnalysisNote          ok
Percent.annual.change 1.78219
TrendDirection        Increasing
```

Figure 8. Data frame output from the NonSeasonalTrendAnalysis function. Note the data frame has been transposed for display purposes.

The NonSeasonalTrendAnalysis () 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 plotting.

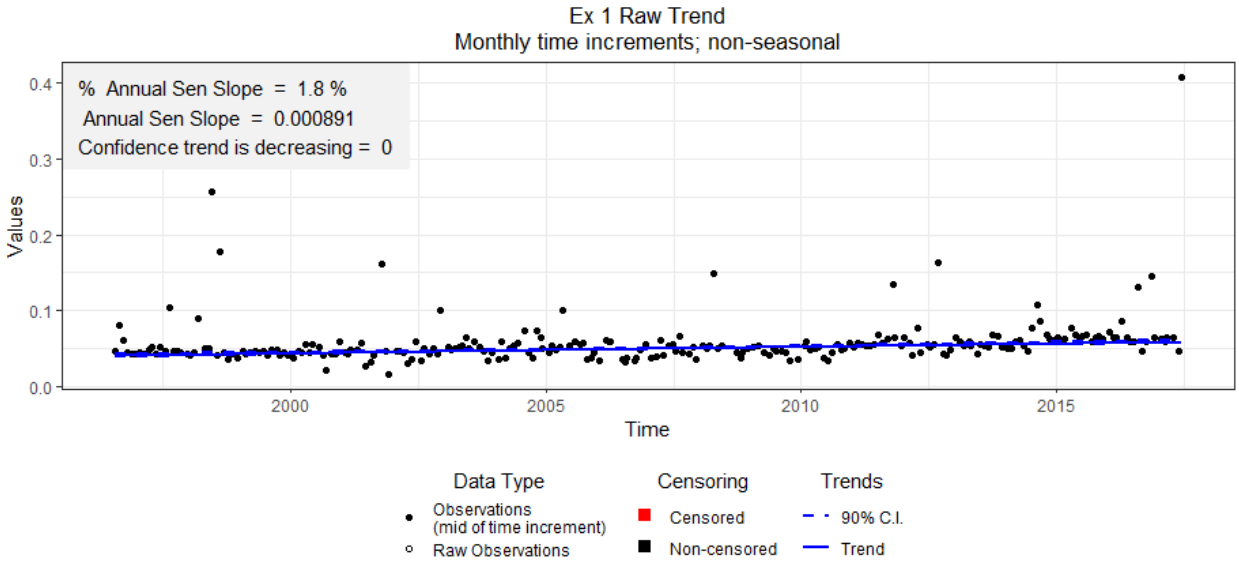


Figure 9. Plot output from SenSlope().

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

Analysis notes return some warnings when Sen slopes are evaluated based on tied values and/or censored values. When values are tied, the estimated Sen slope will be zero. These warnings generally arise for site/variable combinations with large proportions of censored values. They indicate that the 'true' Sen slope cannot be evaluated due to a lack of sufficient resolution of the measurement method (i.ee, the smallest detectable change that the testing method can accurately identify). Assuming the measurement resolution is fit for purpose, we generally would recommend that the results with any of these warnings are still used.

"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 censored data. The returned Sen slope will be zero, and this warning recognises that the 'true' Sen slope (i.e., if the water quality measurements were analysed and reported with higher resolution) 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 resolution of the measurement method. 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 resolution of the measurement method were 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 warning generally occurs when there are 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, (and this is generally due to low resolution of the measurement method). The returned Sen slope is not the 'true' Sen slope due to insufficient measurement resolution. The 'true' Sen slope would be a small value that cannot be evaluated due to the lack of sufficient resolution of the measurement method.

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 seasonal trend analysis has been updated to accommodate the possibility of differences in the assignment of the time increment and the season. When the Season increment is larger than the specified time increment, all observations within the same Season and Year are treated as ties in time when calculating S and the variance of the S statistic for the Mann Kendall test (following methods described in Gilbert, 1987, p. 243).

Further, when evaluating the seasonal Sen slope, slopes are not included between observations with the same Season and Year, but all slopes in the season of one year are compared with all slopes in the season of the other years (Gilbert, 1987, p. 218). This approach should increase the statistical power of the trend assessments by simultaneously reducing variability associated with seasonality, but retaining as many observations as possible.

The function handles the censored values as described in Handling Censored Values section. The output is a data frame of the test results.

```
Trend_ex2<-SeasonalTrendAnalysis(WQData_Ex2,mymain="Ex 1a Raw Trend",do.plot=T)
```

```
nObs                  106
nTimeIncr             106
S                     -43
VarS                  1020.333
D                     416
tau                   -0.1033654
Z                     -1.314856
p                     0.1885582
C                     0.9057209
Cd                    0.9057209
prop.censored         0.01886792
prop.unique           0.2075472
no.censorlevels       1
TimeIncr              Monthly
SeasIncr              Monthly
Median                0.011
AnnualSenSlope        -0.0001669719
Sen_Lci               -0.0005000926
Sen_Uci               0
AnalysisNote          ok
Percent.annual.change -1.517926
TrendDirection        Decreasing
```

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

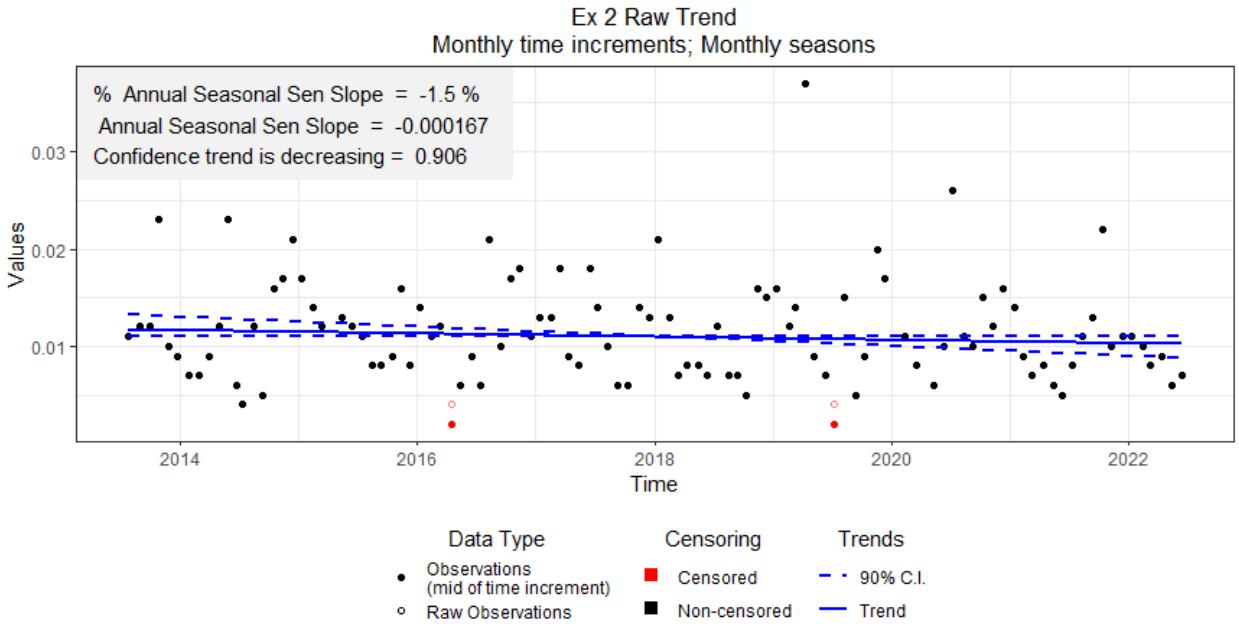


Figure 11. Plot and data frame output from SeasonalSenSlope().

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.

Note: in this example the dataset includes two observations in March 2003 (10th and 31st), and none in April 2003. The new date shift fix assigns the 31st of March observation to April, as such, there is one more "observation" in the trend assessment, compared to as analysed in versions prior to 2502.

4.4 Batch processing trend assessments

For many common applications of water quality trend assessments, there is a need to process multiple site x variable combinations at the same time. In this update we have provided a wrapper function that can be used in conjunction with ddply or dplyr to assist with this task: doMyTrends_v2502. It can also be used with a single site x variable timeseries.

Before using this function, it is still necessary to clean data appropriately, and to run the functions GetMoreDateInfo and RemoveAlphaDetect. The function then executes the following steps:

1. InspectTrendData
    a. Cuts dataset down to specified trend period
    b. Adds on a column TimeIncr to describe the highest frequency time increment that meets the minimum data requirements
2. GetSeason
    a. Evaluates whether data are seasonal and adds the column SeasIncr to the data
3. Run either NonSeasonlTrendAnalysis or SeasonalTrendAnalysis

```
My10yTrends <- ddply(RegionExData, c("siteID", "analyte"), function(x) {
```

```
doMyTrends_v2502(x, Year = "CustomYear", propIncrTol=0.9,
propYearTol=0.9,TrendPeriod=10, EndYear=2020, ValuesToUse="RawValue",
UseMidObs=TRUE, do.plot=FALSE)}, .progress = "win")
```

4.5 Covariate adjustment

While the functionality to undertake covariate adjustment is retained in LWPTrends, we do not recommend using flow adjusted trends for water quality data for what Snelder et al. (2021) refer to as regional applications (i.e., assessing and reporting trends across many sites and variables in the context of regional state of environment reporting). The purpose of flow-adjustment is to remove the confounding effect of flow so that the pattern of interest (the relationship between the observed water quality observations and time i.e., the trend) can be more confidently inferred. However, the definition of models describing observations - instantaneous flow is subjective and therefore there are unquantified uncertainties that arise due to procedural choices around flow adjustment that are likely to be made by individual analysts. Furthermore, there is evidence that trends are often associated with changes in the relationship between concentration and flow through the trend's time period (Snelder and Kerr, 2022). However, flow adjustment (based on defining a relationship between observations and instantaneous flow) assumes that the concentration - flow relationship is constant through time. Violations of this assumption will affect the robustness of 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.

Censored values are used in fitting the observation versus covariate relationship after values below detection limits are set at 0.5*face value and values above reporting limits are set to 1.1*face value. This follows the same conventions used in the Sen Slope assessment. In order that these adjustments are not repeated in the Sen Slope assessment, an argument is used to identify that the data are not based on raw values.

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 = "finalQ", Span = c(0.7), do.plot =T,
plotpval=T, mymain="Example 1a")
```

The adjusted output is shown on Figure 12 and the plots are shown in Figure 13.

```
head(FlowAdjusted)
```

	Gam	LogLog	LOESS0.7	Gam_R	LogLog_R	LOESS0.7_R	Gam_p	LogLog_p	LOESS0.7_p	myDate	OrginalRawValues
1	0.004626671	0.005771573	0.004186797	0.5189954	0.4518479	0.4991341	2.685867e-18	9.929775e-14	7.692382e-17	1997-01-15	0.014
2	0.008089662	0.011242863	0.008247457	0.5189954	0.4518479	0.4991341	2.685867e-18	9.929775e-14	7.692382e-17	1997-02-12	0.021
3	0.007102613	0.007409272	0.007379300	0.5189954	0.4518479	0.4991341	2.685867e-18	9.929775e-14	7.692382e-17	1997-03-19	0.014
4	0.002163419	0.005880275	0.002923741	0.5189954	0.4518479	0.4991341	2.685867e-18	9.929775e-14	7.692382e-17	1997-04-16	0.016
5	0.035617865	0.037887075	0.034930987	0.5189954	0.4518479	0.4991341	2.685867e-18	9.929775e-14	7.692382e-17	1997-05-14	0.047
6	0.004982641	0.006473284	0.004471920	0.5189954	0.4518479	0.4991341	2.685867e-18	9.929775e-14	7.692382e-17	1997-06-18	0.01

Figure 12. 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

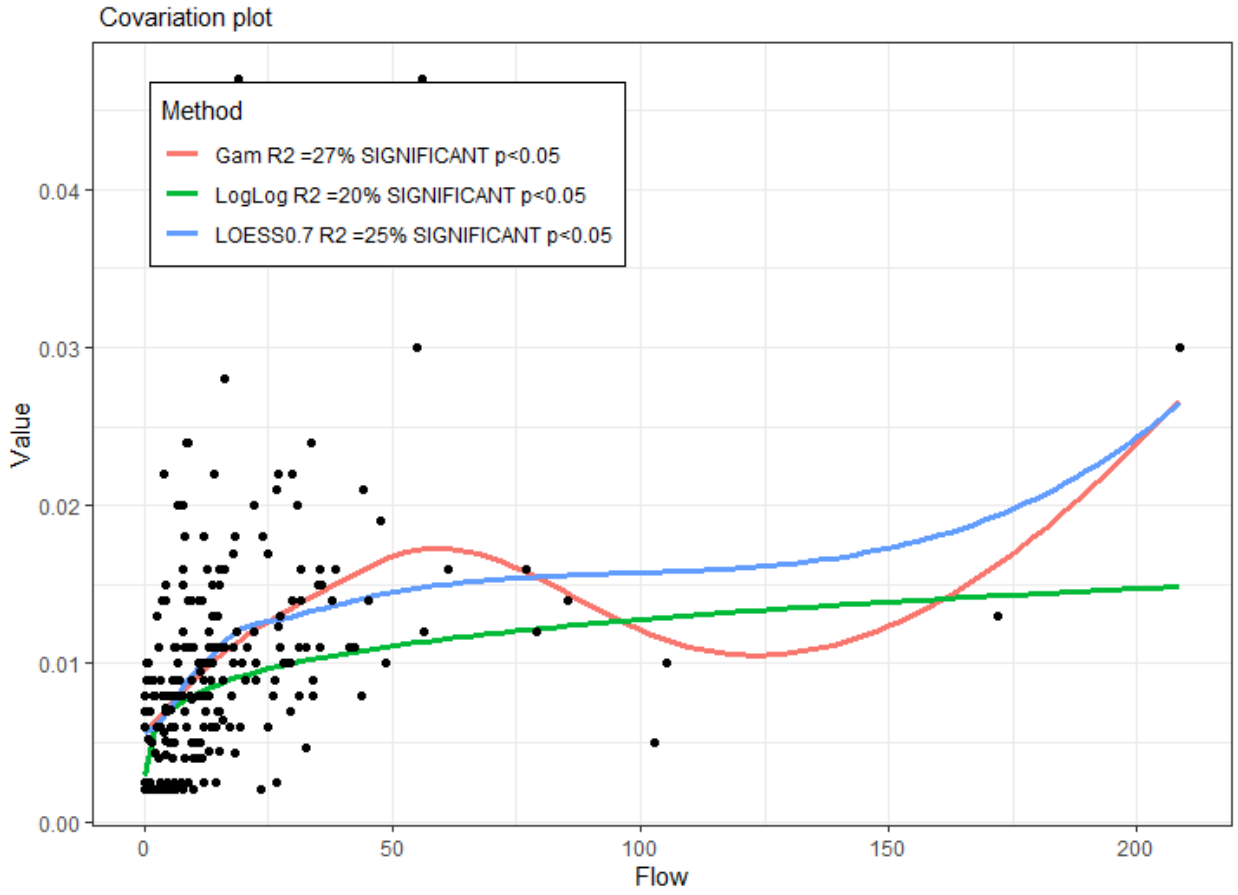
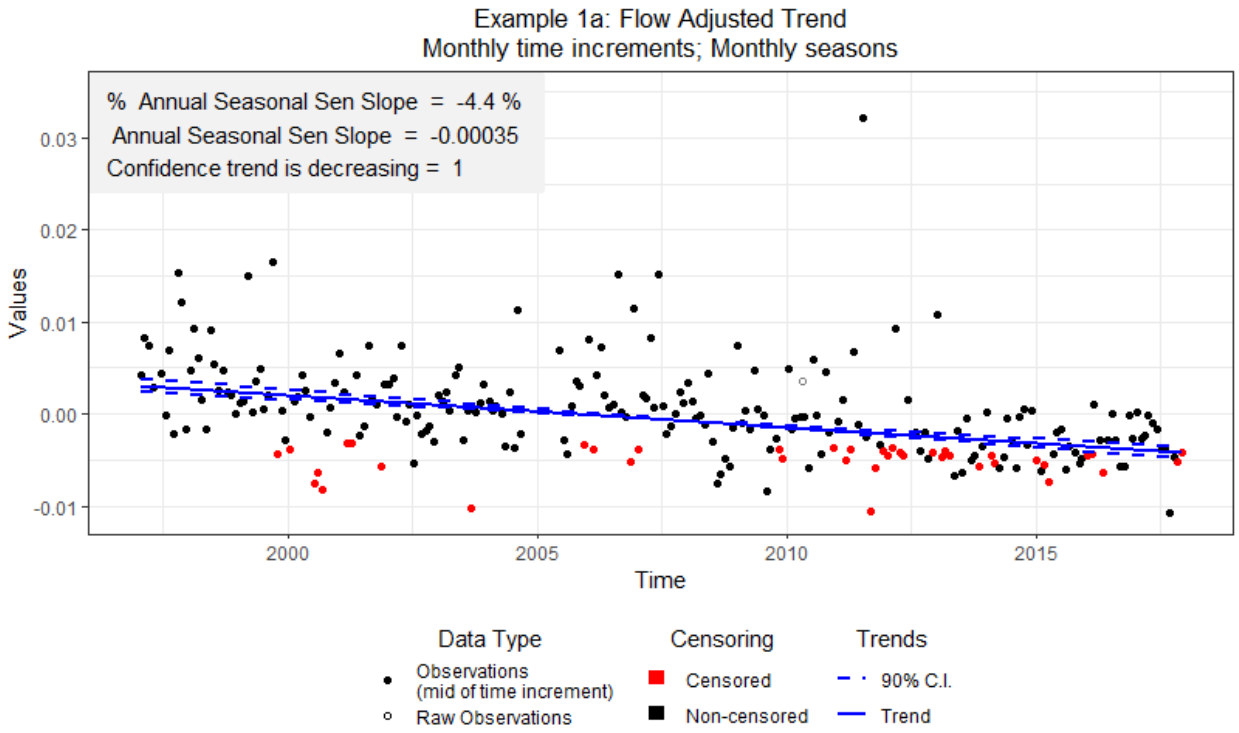


Figure 13. Output plots from the GetAdjustment function.

In the above example the LOESS model might be judged to be 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)])

Trend_ex1aFA<-SeasonalTrendAnalysis(WQData_Ex1a, ValuesToUse="LOESS0.7",
RawValues=FALSE, ValuesToUseforMedian="RawValue", mymain="Example 1a: Flow Adjusted
Trend", do.plot=T)
```



nObs	244
S	-817
VarS	11879.67
D	2360
tau	-0.3461864
Z	-7.486659
p	7.064914e-14
C	1
Cd	1
prop.censored	0.1632653
prop.unique	0.9836735
no.censorlevels	40
TimeIncr	Monthly
SeasIncr	Monthly
Median	0.008
AnnualSenSlope	-0.0003495519
Sen_Lci	-0.0004174958
Sen_Uci	-0.0002799674
AnalysisNote	ok
Percent.annual.change	-4.369398
TrendDirection	Decreasing

Figure 14. 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, or the entire country). The LWP Trends functions provide two approaches for aggregating trend directions. Inputs to these functions are a dataframe comprising the outputs of the previously

described trend analysis functions with many rows, each being the outputs for a given site and variable combination.

The first type of trend aggregation is a plot of aggregated confidence that the trend direction is decreasing. For each site and variable, the confidence that a trend is decreasing (or its complement; increasing) is expressed as a confidence category. The confidence categories (Table 1) can be assigned using the function AssignConfCat():

```
AllTrends10FA$DirectionConf <- AssignConfCat(x=AllTrends10FA,CatType="Decrease")
```

Table 1. Level of confidence categories used to convey the confidence that the trend (or step change) indicated improving water quality.The confidence categories are used by the Intergovernmental Panel on Climate Change (IPCC; Stocker et al., 2014).

Categorical level of confidence trend was decreasing	**Value of C_d (%)**
Virtually certain	0.99–1.00
Extremely likely	0.95–0.99
Very likely	0.90–0.95
Likely	0.67–0.90
About as likely as not	0.33–0.67
Unlikely	0.10–0.33
Very unlikely	0.05–0.10
Extremely unlikely	0.01–0.05
Exceptionally unlikely	0.0–0.01

Functionality also exists to use to categorise confidence that trend direction is improving. Each site trend is assigned to a category by firstly, converting C_d into a confidence that a trend was improving (C_i). Improvement is indicated by decreasing trends for most water quality variables ($C_i = C_d$), but for some variables, increasing trends indicated improvement (e.g., macroinvertebrate metrics, visual clarity, Secchi depth, dissolved oxygen etc). For these variables, C_i is the complement of C_d (i.e., C_i=1-C_d). For some variables, (e.g., pH, temperature etc), improvement is not clearly related to increasing or decreasing trends, and for these variables this approach should not be used. Each site/variable combination can then be assigned to a confidence in trend direction category according to its evaluated confidence of improvement. An option also exists to provide a simpler categorisation, following the categories shown in Table 1.

```
AllTrends10FA$ImproveConf <-
AssignConfCat(x=AllTrends10FA,CatType="Improve",Reverse="MCI",excl="pH",nCat="Sim
ple")
```

Table 2. Level of confidence categories used to convey trend confidence and direction.

Categorical level of trend confidence and direction	Value of C_i (%)
Very likely improving	0.90–1.00
Likely improving	0.67–0.90
Low confidence in direction	0.33–0.67
Likely degrading	0.10–0.33
Very likely degrading	0.0–0.10

The categorical levels of confidence for all sites and variables are then summarised using a colour coded bar chart (Figure 15). See the demonstration in the example file: RunLWPTrendsExample_v2502.R.

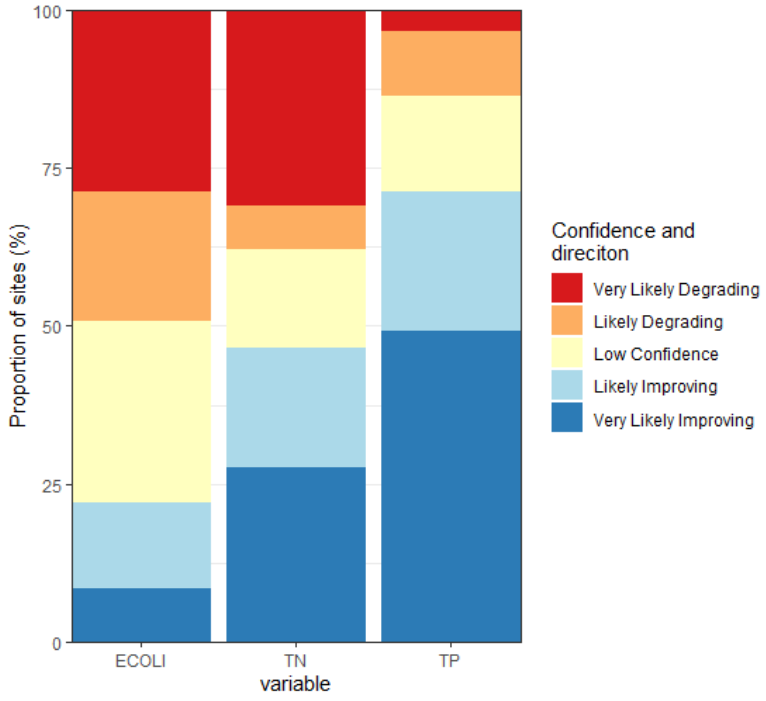


Figure 15. Example of aggregate confidence in trend direction plot showing the proportion of water-quality monitoring sites with improving trends at each categorical level of confidence.

The second approach uses a procedure for assessing aggregate trends developed by Snelder et al. (2022). The method uses the directions of the individual site trends derived using the Mann Kendall trend assessment method to calculate the aggregate trend direction (D^T) to be the modal (i.e., most frequently occurring) direction of the individual site trends and the aggregate trend strength ($\hat{\mathrm{T}}$)[2] as the proportion of site trends that are in the modal direction. The confidence in the

[2] Note that the Greek letter tau is used for trends corresponding to both individual sites and aggregated sites. When an individual site trend is being referred to the lower-case tau (τ) is used. When an aggregate trend is being referred to, the upper-case tau is used (T).

assessed aggregate trend direction (C^{T}) is calculated from the confidence in the direction of the individual site trends and adjusted for spatial autocorrelation between sites using the method described by Snelder et al. (2022). Briefly, the adjustment is necessary if there is spatial correlation among sites within a monitoring network. Spatial correlation means that the information representing the sites is not independent and this results in under-estimation of the variance and over-estimation of confidence in the assessed aggregate trend direction. The method of Snelder et al. (2022) uses the sample cross-correlation coefficient, which is computed from the observation time series for all sites, to "correct " the variance to account for the spatial correlation. C^{T} can be expressed as categorical levels of confidence in assessed trend direction by four categories shown in Table 3. The function also allows user to select whether the trend direction is expressed purely as direction (i.e., increasing/decreasing) or as improvement (i.e., improving/degrading).

Table 3: Level of confidence categories used to convey the confidence in the assessed trend direction.

Categorical level of confidence in assessed trend direction	**Value of C**
Highly likely	0.95 - 1.00
Very likely	0.90 – 0.95
Likely	0.67 – 0.90
Uncertain	0.50 – 0.67

```
TauRegional <- dlply(My10yTrends,.(analyte), function(y) getTAU(x=y,
              obs=RegionExData[RegionExData$analyte==y$analyte[1],],
              SiteNameCol="siteID", VarNameCol="analyte", ValuesToUse = "RawValue",
              Year = "CustomYear", UseMidObs=T))
```

Extract summary table:

```
TauRegionalSummary <- ldply(TauRegional, function(x) return(x$SummaryResults))
```

analyte	ECOLI	TN	TP
M	59	59	59
TAU	0.6440678	0.5762712	0.8135593
VarTAU	0.002205640	0.001860323	0.001662216
CorrectedVarTAU	0.008345952	0.015388517	0.006567288
DT	Increasing	Increasing	Decreasing
CT	0.9426011	0.7306698	0.9999454
ConfCat	Very likely	Likely	Highly likely
UncorrectedCT	0.9989211	0.9614981	1.0000000
UncorrectedConfCat	Highly likely	Highly likely	Highly likely

Histograms of the Pearson correlation coefficients between sites, by variable, can be obtained from PlotRegionalCor() (Figure 16) and aggregate trend direction, trend strength and confidence can be summarised graphically by PlotRegionalConfDir() (Figure 17)

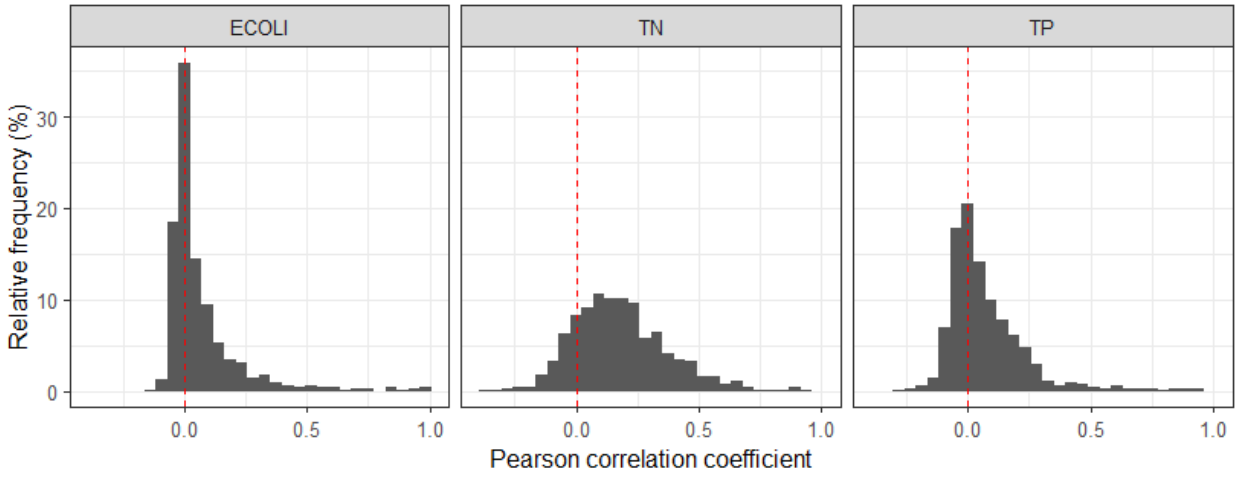


Figure 16. Distributions of cross-correlations between observations for all pairs of sites for each of the seven water quality variables for the 10-year trend period. The red dashed line indicates correlation of zero.

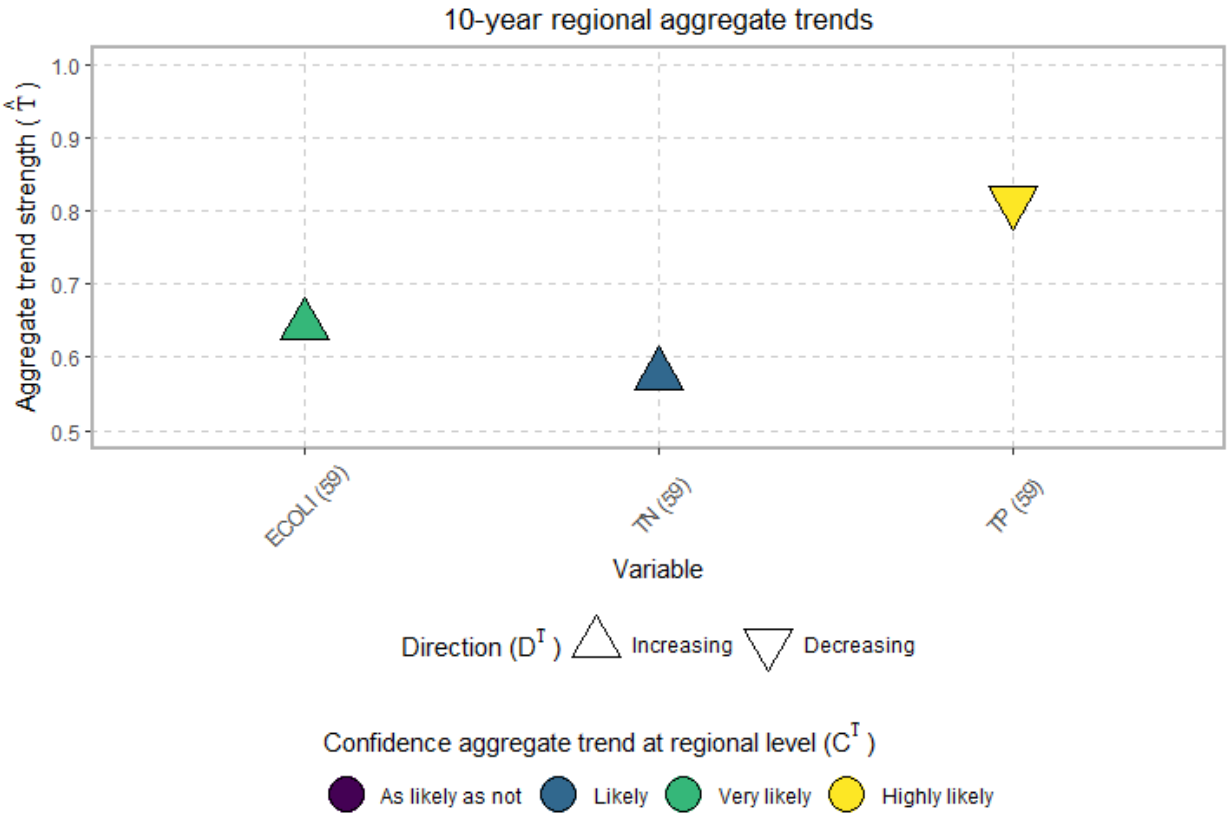


Figure 17. Aggregate trend direction (D^T) and strength ($\hat{T}$) for 10-year trends for three water quality variables over all sites. Confidence in the aggregate direction (C^T) is indicated by the four confidence categories. Confidence is shown for calculations that have been corrected for spatial correlation. The values in parentheses after the variable names on the x-axis labels indicate the number of sites that were included in the analysis.

6 References

Gilbert, R.O., 1987. Statistical Methods for Environmental Pollution Monitoring. John Wiley & Sons.

Helsel, D.R. (1990) Less than Obvious-Statistical Treatment of Data below the Detection Limit. *Environmental Science & Technology,* 24: 1766–1774.

Helsel, D.R. (2005) *Nondetects and Data Analysis. Statistics for Censored Environmental Data*. Wiley-Interscience, New Jersey.

Helsel, D.R. (2012) Statistics for Censored Environmental Data Using Minitab and R. John Wiley & Sons, Inc., Hoboken, New Jersey. **http://onlinelibrary.wiley.com/doi/10.1002/9781118162729.ch3/summary. Accessed 19 Aug 2016**.

Helsel, D.R., R.M. Hirsch, K.R. Ryberg, S.A. Archfield, and E.J. Gilroy, 2020. Statistical Methods in Water Resources. Report, Reston, VA. Available at: **http://pubs.er.usgs.gov/publication/tm4A3**

Hirsch, R.M., Slack, J.R., Smith, R.A. (1982) Techniques of Trend Analysis for Monthly Water Quality Data. *Water Resources Research,* 18: 107–121.

Mann, H.B. (1945) Nonparametric Tests against Trend. *Econometrica: Journal of the Econometric Society*: 245–259.

McBride, G.B. (2019) Has Water Quality Improved or Been Maintained? A Quantitative Assessment Procedure. *Journal of Environmental Quality.*

Snelder, T. and C. Fraser, 2019. Monitoring Change over Time: Interpreting Water Quality Trend Assessments. LWP Client Report, LWP Ltd, Christchurch, New Zealand. Available at: **https://www.mfe.govt.nz/publications/fresh-water/monitoring-change-over-time-interpreting-water-quality-trend-assessments**

Snelder, T., C. Fraser, S. Larned, and A. Whitehead, (2021). Guidance for the Analysis of Temporal Trends in Environmental Data. NIWA Client Report, NIWA, Christchurch, New Zealand. Available at: **https://landwaterpeople.co.nz/wp-content/uploads/2021/01/TrendAssessmentGuidance.zip**

Snelder, T., C. Fraser and Whitehead, A. L. (2022). Continuous measures of confidence in direction of environmental trends at site and other spatial scales. *Environmental Challenges*, *9*, 100601.

Snelder, T. and T. Kerr, 2022. Relationships between Flow and River Water Quality Monitoring Data and Recommendations for Assessing NPS-FM Attribute States and Trends. Prepared for Auckland Council, LWP Ltd.

Stocker, T., Qin, D.Q., Plattner, G.K. (2014) *Climate Change 2013: The Physical Science Basis: Working Group I Contribution to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge University Press.

7 Function descriptions

`GetMoreDateInfo`

Description

Takes dates and produces additional columns summarising, months, bimonths (samples ever 2 months), quarters, biannual (samples twice a year) and years. 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 function also as an option to search for observations that should be representative of a sample from the following (or previous) month, and assign the additional date information associated with the following (or previous) month, rather than the month of sampling.

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, FindDateShifts=TRUE,dateshiftMINdt=20)
```

Arguments

`x`	Dataframe or vector containing myDate
`firstMonth`	Specify the first month (numeric (1:12)) for the analysis year
`FindDateShifts`	Logical. Indicates whether to look for observations that should be representative of previous or following months.
`dateshiftMINdt`	Minimum time difference (days) between two observations within a single month for them to be considered independent and for one to potentially be assigned to the previous or following month. (default is 20 days)

Value

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

`Year`	Calendar Year (numeric)
`CustomYear`	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 (pairs of months) String (factor)
`Qtr`	Quarter string (factor)
`BiAnn`	Bi-annual (6-monthly increments) String (factor)

`RemoveAlphaDetect`

Description

Takes a dataframe including a column observed values that includes censored values (specified as the prefix > or <,column is type character) and returns face values and information about the nature of the censoring.

Usage

```
RemoveAlphaDetect(Data,ColToUse="Value")
```

Arguments

`x`	Dataframe containing column ColToUse
`ColToUse`	Name of column in dataframe with censored observations

Value

A data frame with original Data plus additional fields as follows

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

`InspectTrendData`

Description

InspectTrendData prepares the data for subsequent trend analysis. This involves cutting down the dataset to a specified trend period (specified by an end year and a trend period length in years) and adds on the time increment column for the trend analysis, which is selected as the lowest frequency increment that that meets the minimum data requirements specified (see Snelder et al. 2022 for guidance on minimum data requirements). The function can also provide optional summary statistics and graphs.

Usage

```
InspectTrendData (x, TimeIncrOpts=NA,Year = "Year", TrendPeriod=NA,
EndYear = NA, AnnualOK=”MCI”,
propIncrTol=0.9,propYearTol=0.9,ReturnALLincr=FALSE, do.plot = FALSE,
 ...)
```

Arguments

`x`	Input data frame must contain: myDate, RawData, Censored
`TimeIncrOpts`	Vector of acceptable (sub-annual) time increments. Default is NA, which uses all coded time increments c("Monthly","Bi-monthly","Quarterly","Bi-annual") – but a subset of these can alternatively be specified.
`Year`	Column name for Year data to be used
`TrendPeriod`	Sets the trend period to be analysed (If NA, will be selected from as longest possible trend period in whole years from the data)
`EndYear`	Sets the end year of the time series. (If NA, will be selected from data)
`AnnualOK`	A vector of variables which are allowed to have the time increment set as annual.
`propIncrTol`	The proportion of time increments x years that must have observations
`propYearTol`	The proportion of year that must have observations
`do.plot`	Produce a plot if TRUE and ReturnAllIncr is TRUE
`FlowtoUse`	Name of column of flow data (only required if relevant)
`UseMidObs`	Default is TRUE and uses value closest to the mid-point of the season. Set to FALSE, to use median of time increments.
`ReturnALLincr`	Logical. If true, only returns x with an additional column indicating the time increment.
`UseMidObs`	Defines how multiple observations in a month are dealt with. Default is TRUE, which takes the observation that is closest to the middle of the season. Set to FALSE, to take the median across the season.

...	Passed to plot function

Value

A data frame of x with additional columns appended. Optionally a list with an extra dataframe and optionally a plot. Data frame appended fields as follows

`Incr`	The time increment name (one row of data per time increment)
`TrendPeriodL`	Length of trend period
`Nobs`	Total number of observations
`nYear`	The number of years with data in the sampling period
`propYear`	Proportion of trend period years with observations
`nIcnrYear`	Number of time increment x year with observations
`propIncrYear`	Proportion of total time increment x year for the trend period with observations
`propCen`	Proportion of observations that are censored
`nCenLevelsLT`	Total number of unique non-detect censor levels (data with <)
`nCenLevelsT`	Total number of unique detection limits (data with >)
`nFlow`	Proportion of observations with simultaneous flow observations

GetSeason

Description

The function GetSeason() evaluates an appropriate season for the analysis by performing a Kruskal Wallis test (non-parametric ANOVA, via the SeasonalityTest() function) on the observations using possible time increment as the explanatory (categorical) variable. Possible season time increments are (1) the time increment specified by TimeIncr, and (2) any other time increment that is a whole multiple of TimeIncr. This function also then returns the SeaonalityTest() outputs for the selected Season (i.e., user does not need to run SeasonalityTest() as well).

Usage

```
GetSeason(x, ValuesToUse="RawValue", printKW, do.plot=TRUE,…)
```

Arguments

x	Input data frame must contain, TimeIcnr, ValuesToUse,Centype, Censored
ValuesToUse	Select Column with data to use in the test
printKW	If TRUE the function prints a table of the KW test outputs for all tested seasons to the console.
do.plot	Option to return a ggplot object of the boxplots for the chosen Season
…	Passed to plot function

Value

A list containing (1) A data frame with fields as follows (in addition to those columns in x), (2) the outputs from SeasonalityTest() and optionally (3) a plot object.

Season	Season string (factor)

`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 optionally produces a box plot of the data grouped by season.

Usage

```
SeasonalityTest(x, ValuesToUse="RawValue", do.plot=TRUE)
```

Arguments

`x`	Input data frame must contain, Season, ValuesToUse,Centype, Censored
`ValuesToUse`	Select Column with data to use in the test
`do.plot`	Produces a ggplot object in a list if TRUE
…	Passed to plot function

Value

A data frame and (optionally) a plot. If do.plot set to TRUE 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)
`TimeIncr`	The time increment of the dataset
`Season`	The season (time increment) used for the seasonality test

`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", do.plot=F)
```

Arguments

`x`	Input data frame must contain: ValuesToUse, myDate, CenType
`forwardT`	See NADA function ros()
`reverseT`	See NADA function ros()
`do.plot`	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 do.plot="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	Total number of observations
nTimeIncr	Number of time increments with observations
S	S-statistic
VarS	Variance
D	n * (n - 1)/2
tau	Kendall's tau
Z	Z-statistic
p	p-value
C	Confidence in trend direction
Cd	Confidence that trend is decreasing
prop.censored	Proportion of observations that are censored
prop.unique	Proportion of unique observations
no.censorlevels	Number of unique left censor levels
TimeIncr	Time increment of the analysis
SeasIncr	Season time increment of the analysis (or "non-seasonal")

Median	Data Median (based on data from ValuestoUseforMedian)
AnnualSenSlope	Annual Sen slope (attribute units/year)
Sen_Lci	Lower confidence interval for annual Sen slope
Sen_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
Percent.annual.change	Percent annual change in Sen slope (value between 0 and 1)
TrendDirection	Face value trend direction

doMyTrends_v2502

Description

New wrapper function that reads in data, calls InspectTrendData to cut down the dataset to the appropriate trend period and assign the highest acceptable frequency as the time increment. The wrapper then performs a seasonality test to determine if the data is seasonal (and which season explains the most observed variation), and to add a SeasIncr column to the dataset. Based on the seasonality test results, the function chooses to either call the seasonal or non-seasonal tend analysis wrappers. The function must be run on data that has already been passed through RemoveAlphaDetect

Usage

```
doMyTrends(x, TimeIncrOpts=NA, do.plot=F, propIncrTol=0.9,
propYearTol=0.9, AnnualOK=c("MCI"), TrendPeriod=NA, EndYear=NA,
Year="Year",)
```

Arguments

x	Input data frame must contain: myDate, RawValue, analyte, Censored, CenType and data metadata columns from GetMoreDateInfo
TimeIncrOpts	Vector of acceptable (sub-annual) time increments. Default is NA, which uses all coded time increments c("Monthly","Bi-monthly","Quarterly","Bi-annual") – but a subset of these can alternatively be specified.
do.plot	Set to TRUE to return a timeseries plot of the trend. Does not apply to the InspectData function (hardwired as do.plot=F)
Year	Column name for Year data to be used
TrendPeriod	Sets the trend period to be analysed (If NA, will be selected from as longest possible trend period in whole years from the data)
EndYear	Sets the end year of the time series. (If NA, will be selected from data)
AnnualOK	A vector of variables that are allowed to have the time increment set as annual.
propIncrTol	The proportion of time increments x years that must have observations
propYearTol	The proportion of year that must have observations
…	Variables to be passed to MannKendall() and SenSlope() functions

Value

If do.plot=”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	Total number of observations
nTimeIncr	Number of time increments with observations
S	S-statistic
VarS	Variance

D	n * (n - 1)/2
tau	Kendall's tau
Z	Z-statistic
p	p-value
C	Confidence in trend direction
Cd	Confidence that trend is decreasing
prop.censored	Proportion of observations that are censored
prop.unique	Proportion of unique observations
no.censorlevels	Number of unique left censor levels
TimeIncr	Time increment of the analysis
SeasIncr	Season time increment of the analysis (or "non-seasonal")
Median	Data Median (based on data from ValuestoUseforMedian)
AnnualSenSlope	Annual Sen slope (attribute units/year)
Sen_Lci	Lower confidence interval for annual Sen slope
Sen_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
Percent.annual.change	Percent annual change in Sen slope (value between 0 and 1)
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",
RawValues=TRUE, UseMidObs=FALSE)

OR

SeasonalKendall(x, ValuesToUse = "RawValue", Year = "Year",
RawValues=TRUE, UseMidObs=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
RawValues	Set to FALSE if using flow adjusted data. Default is treatment of the data as raw
UseMidObs	Defines how multiple observations in a time increment are dealt with. Default is TRUE, which takes the observation that is closest to the middle of the season. Set to FALSE, to take the median across the season.

Value

A data frame with fields as follows

nObs	Total number of observations
nTimeIncr	Number of time increments with observations
S	S-statistic
VarS	Variance
D	n * (n - 1)/2
tau	Kendall's tau
Z	Z-statistic
p	p-value
C	Confidence in trend direction
Cd	Confidence 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
TimeIncr	Time increment of the analysis
SeasIncr	Season time increment of the analysis (or “non-seasonal”)

`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", mymain=”My trend plot”,…)

OR

SeaonalSenSlope(x, ValuesToUse = "RawValue", ValuesToUseforMedian= "RawValue", Year = "Year", 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
`do.plot`	Returns a ggplot object if TRUE
`mymain`	Optional title for plot.
`RawValues`	Set to FALSE if using flow adjusted data. Default is treatment of the data as raw
`UseMidObs`	Defines how multiple observations in a time increment are dealt with. Set to TRUE to use to observation that is closest to the middle of the time increment. Set to FALSE to take the median across the time increment. Default is TRUE.
…	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 do.plot=TRUE 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]
Sen_Lci	Lower confidence interval for annual Sen slope
Sen_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)

Notes: * These variables are only provided in the interest of back compatibility with LWPTrends versions prior to V2001. They are not included as outputs in the wrapper functions

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), do.plot = 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
do.plot	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
mymain	Title to be passed to the GGplot objects
…	Additional plotting variables to pass to the base plot

Value

A data frame and (optionally) a plot. If do.plot=”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

`AssignConfCat`

Description

Assigns a categorical description of the confidence in one of: (1) trend direction; (2) the trend is decreasing; or (3) the trend is improving. Can specify whether to use simple categories (Table 2), or full categories (Table 1).

Usage

```
AssignConfCat(x,CatType="Improve",Reverse=c("CLAR","MCI),
NoImproveDir=c("pH),nCat="Simple")
```

Arguments

`x`	If CatType is "Direction", can be a dataframe or vectors containing C. If CatType is "Decrease" a dataframe containing Cd. If CatType is "Improve", a dataframe containing Cd and analyte. Dataframe or vector containing: npID and Cd
`CatType`	Category type sought. "Direction" (confidence in trend direction). "Improve" (confidence trend is improving). "Decrease" (confidence trend is decreasing).
`Reverse`	Vector of analyte names for which an increasing trend indicates improvement (only used when CatType = "Improve").
`NoImproveDir`	Vector of analyte names for which an increasing trend indicates neither improvement or degradation (only used when CatType = "Improve"). Categories for these analytes will be returned as NA values.
`nCat`	Number of categories. "Simple" used categories in Table 2. "Full" uses categories in Table 1.

Value

A vector of confidence categories (as per table1 or table 2) as factors. .

getTAU

Description

Method to assess the modal direction of individual trends at multiple sites (i.e., the aggregate trend direction) and assesses the confidence associated with this determination while taking into account the spatial autocorrelation between sites.

Usage

```
getTAU(x, obs, SiteNameCol="sID", VarNameCol="analyte", ValuesToUse="RawValue", Year="Year", UseMidObs=T, DirType="Decrease", Reverse=c("CLAR","MCI"), excl=c("pH"))
```

Arguments

x	Dataframe of outputs from TrendAnalysis function (must include a column with the analyte name and site name)
sobs	Data frame of timeseries data used for trend assessment (must include a column with the analyte name and site name)
SiteNameCol	Column name for site name
VarNameCol	Column name for variable name
ValuesToUse	Select Column with data to use in the test
Year	Column name for Year data to be used
UseMidObs	Defines how multiple observations in a time increment are dealt with. Default is TRUE, which takes the observation that is closest to the middle of the season. Set to FALSE, to take the median across the season.
DirType	Defines whether you want direction to be expressed as decreasing/increasing ("Decrease") or improving/degrading ("Improve")
Reverse	String of variables (analyte) for which increasing trends indicate improvement (only used when DirType = "Improve")
Excl	String of variables (analyte) for which trend direction is not related to improvement or degradiation (only used when DirType = "Improve").

Value

A list containing a vector of correlation coefficients between each pair of sites (ObservationCorrelations) and a dat A data frame with fields as follows

M	Total number of sites
TAU	Aggregate trend strength
VarTAU	Variance in aggregate trend strength

CorrectedVarTAU	Variance in aggregate trend strength adjusted for spatial autocorrelation between sites
DT	Aggregate trend direction
CT	Aggregate confidence in trend direction
ConfCat	Confidence categories (as per Table 3)
UncorrectedCT	Uncorreacted confidence in trend direction
UncorrectedConfCat	Confidence category based on UncorrectedCT

.

`PlotRegionalConfDir`

Description

Generates summary plot of aggregate trend assessment.

Usage

```
PlotRegionalConfDir(x, mymain=NA, DirType="Decrease")
```

Arguments

`x`	A dataframe of the $SummaryResults outputs from the getTAU function
`mymain`	Optional plot title.
`DirType`	Defines whether direction is expressed as decreasing/increasing ("Decrease") or improving/degrading ("Improve") (must match specification used in getTAU)

Value

A plot summarising aggregate trends.

`PlotRegionalCor`

Description

Generates histograms of the Pearson correlation coefficients between sites used in an aggregate trend assessment, by variable.

Usage

```
PlotRegionalCor(x)
```

Arguments

`x`	A list of outputs from the getTAU function

Value

A plot showing distributions of cross-correlations between observations for all pairs of sites for each of water quality variable.