TECHNICAL PUBLICATION SJ2004-4

A USER-WRITTEN SAS PROGRAM FOR ESTIMATING TEMPORAL TRENDS AND THEIR MAGNITUDE

Technical Publication SJ2004-4

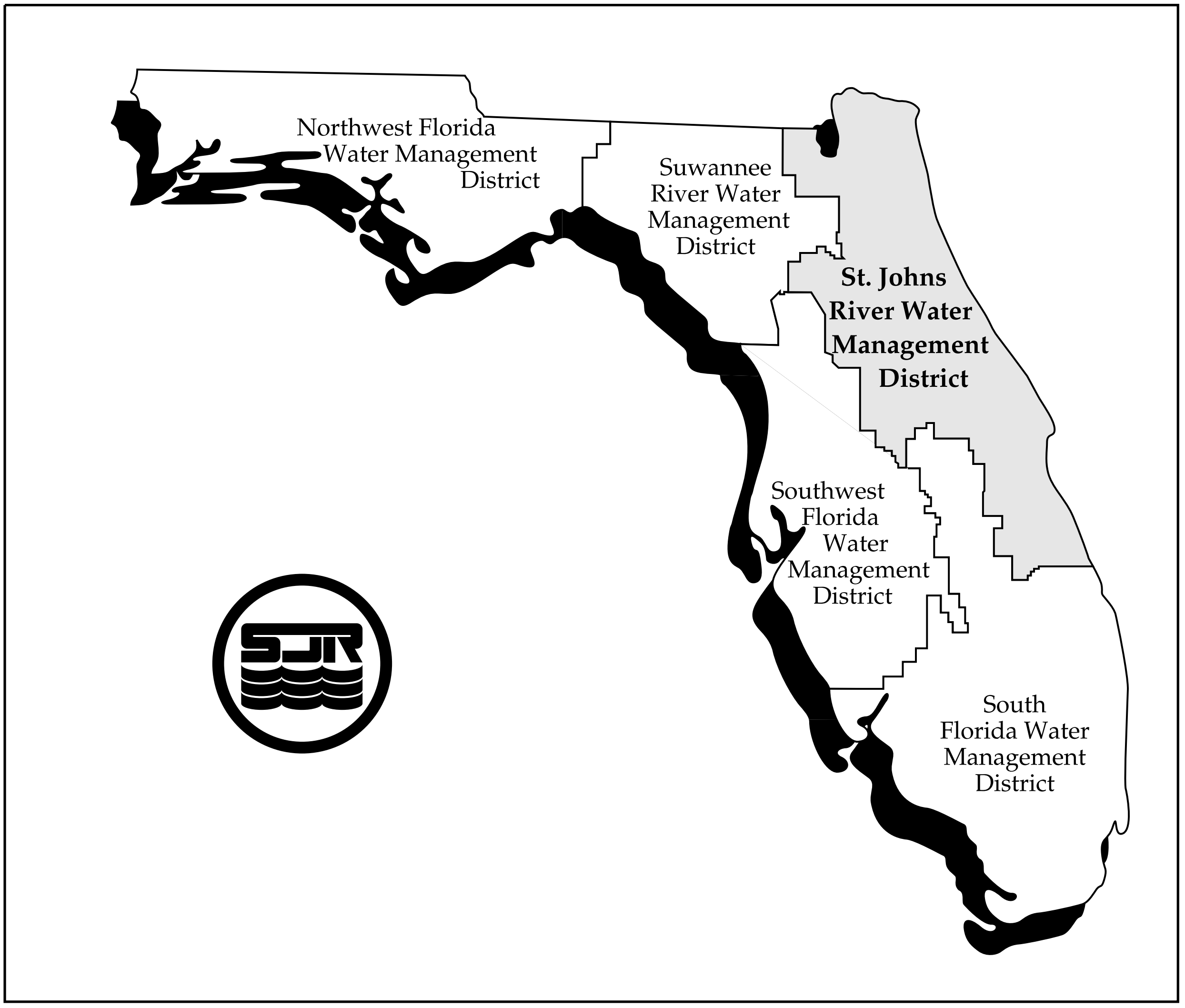
A USER-WRITTEN SAS PROGRAM FOR ESTIMATING TEMPORAL TRENDS AND THEIR MAGNITUDE

by

Steve Winkler

St. Johns River Water Management District
Palatka, Florida

2004



The St. Johns River Water Management District (SJRWMD) was created by the Florida Legislature in 1972 to be one of five water management districts in Florida. It includes all or part of 18 counties in northeast Florida. The mission of SJRWMD is to ensure the sustainable use and protection of water resources for the benefit of the people of the District and the state of Florida. SJRWMD accomplishes its mission through regulation; applied research; assistance to federal, state, and local governments; operation and maintenance of water control works; and land acquisition and management.

This document is published to disseminate information collected by SJRWMD in pursuit of its mission. Copies of this document can be obtained from:

Library
St. Johns River Water Management District
4049 Reid Street • P.O. Box 1429
Palatka, FL 32178-1429

Phone: (386) 329-4132

EXECUTIVE SUMMARY

This report provides documentation and instruction for a user-written computer program designed to determine temporal trends in water quality data and their magnitudes. The computer program was written using SAS® software on a UNIX SunOS 5.8 platform. The program incorporates both a non-seasonal and a seasonal version of the non-parametric Mann-Kendall trend test. In addition, the program provides an estimate of the trend magnitude using a Sen slope estimation. Confidence limits on the slope are provided as well. Users can modify the program as needed to suit their requirements for data analysis. Examples of program output are provided in the report. The program is available on the St. Johns River Water Management District's UNIX network.

CONTENTS

TABLES

INTRODUCTION

Temporal trend determination is an important part of data analysis. Trend analysis will indicate whether analyte concentrations are increasing or decreasing over time. In addition, an estimate of the trend's magnitude can help to determine whether a statistically significant trend is of practical concern.

This report presents background information and instruction for a user-written SAS® program that performs both a trend test and an associated slope estimation. Although the program was written primarily for use with surface water quality data, it could be used for other types of data as well, such as air quality data, groundwater data, or other types of environmental measurements.

Environmental data tend to be either normally (parametric) or non-normally (non-parametric) distributed. There are different trend tests available for either of these cases. For example, linear regression is often used for parametric data. A linear regression can be appropriate as long as all assumptions of the method are met. However, non-normally distributed data are best analyzed using non-parametric tests, such as the Mann-Kendall test (Helsel and Hirsch 1995).

While the Mann-Kendall test is a good test to use for non-parametric water quality data, it hasn't been available as an option in many software products. Although the test is available in a few commercially available software packages such as WQSTAT, these packages are generally designed for individual station analysis, and batch processing using the test is not available. One of the goals of the ambient water quality monitoring program at the St. Johns River Water Management District (SJRWMD) is to evaluate trends at over 150 water quality sampling stations. Evaluating each station individually using WQSTAT was a cumbersome and inefficient technique. A much more practical solution was to program the test using SAS software.

SAS is a powerful data management and analysis software package produced by the SAS Institute based in Cary, North Carolina. Although it doesn't come with a Kendall test, SAS software is sufficiently versatile that such a test can be programmed using it. The user-written version of the test described in this report allows the user to batch-process multiple stations. In addition, there is a Sen slope estimation of the trend's magnitude.

PROGRAM OVERVIEW

The program was written using SAS Version 8.2 on a UNIX SunOS 5.8 platform. The program contains a non-seasonal and a seasonal version of the Mann-Kendall trend test. The seasonal version of the test is invoked when the program determines that there are statistically significant differences between the data from different user-defined seasons. The program doesn't evaluate what months or combinations of months constitute a "season" in the dataset. Rather, the user must assign seasons they consider most appropriate for the locale and type of data analysis they are doing. In Florida, for example, it is often useful to assign a wet and a dry season to reduce variability in the dataset. For areas further north, the typical four seasons of fall, winter, spring, and summer might be most appropriate. The decision rests with the user, who must decide based on their assessment of climatic conditions, data sufficiency, and other variables. If the user assigns no seasons to the entire dataset, only the non-seasonal test will be applied. If the user believes that there is or should be seasonality in the dataset, then they can assign seasons. For each station, the program uses a Wilcoxon test to determine if the datasets for each assigned season are significantly different from each other. The default significance level of the seasonality test is 0.1, but it can be altered in the program. Those stations that have significantly different datasets will be analyzed using the seasonal version of the Mann-Kendall test. The program also calculates a Sen slope estimation and associated confidence limits, if possible. Seasons are assigned in section 4 of the program.

NON-SEASONAL MANN-KENDALL TEST

The Mann-Kendall test is a nonparametric trend test. The test does not require normally distributed data and is well suited for analyzing datasets that have missing or tied data (Gilbert 1987). The null hypothesis (H_o) is that there is no trend. The alternative hypothesis (H_a) is that there is a trend. The test first ranks all observations by date order. Then the difference between each successive value is calculated, and the sum of the signs of those differences is evaluated as the Kendall sum statistic, or K.

$$\text{Kendall K} = \sum_{k=1}^{n-1} \sum_{j=k+1}^{n} \operatorname{sgn}(x_j - x_k)$$

This process is repeated in an iterative fashion until all successive differences have been evaluated. The Kendall K is then the number of positive differences minus the number of negative differences. The Kendall K is then compared to a critical value, and if it exceeds that value, then there is a significant trend. The number of data values is important in determining the critical value for the Kendall K. For a dataset with less than four data values, no critical values are available. For datasets with 40 or fewer values, the probability associated with the Kendall K can be found in "Upper-Tail Probabilities for the Null Distribution of the Kendall K Statistic" (Table A30, Hollander and Wolfe 1999). Critical values approximations for a two-tailed test at a significance level of $\alpha \leq 0.1$ are currently coded into the program. Should the user want to have a two-tailed test at an α value other than 0.1, reprogramming will be required at the appropriate point in the program. Table 1 gives some approximations of the critical values for selected *n*.

Table 1. Some approximate critical values for a two-tailed test

alpha	n=5	n=10	n=15	n=20	n=25	n=30	n=35	n=40
α=0.05	11	30	40	62	85	111	139	169
α=0.1	7	29	34	52	71	94	117	142
α=0.2	7	16	28	40	57	73	92	111

Source: Hollander and Wolfe, Table A30

If the dataset has more than 40 observations, a z-score is calculated based on the K statistic and the variance. The variance for the non-seasonal Kendall statistic is

$$\text{Var} = \frac{1}{18}\left[n_i(n_i - 1)(2n_i + 5) - \sum_{p=1}^{g_i} t_{ip}(t_{ip} - 1)(2t_{ip} + 5)\right]$$

In this equation, *n* represents the total number of values, *g* represents the number of tied groups, and *t* is the number of ties in each group (Gilbert 1987). The z-score is calculated according to one of the following equations:

$Z = K\text{-}1/\sqrt{(\text{variance } K)}$ for $K > 0$
$Z = 0$ for $K = 0$
$Z = K+1/\sqrt{(\text{variance } K)}$ for $K < 0$

In the program, if the probability associated with the z-score is less than or equal to 0.1, then the trend is considered to be significant. If the Kendall K is positive, that means that values are in general increasing over time. If the Kendall K is negative, then values are decreasing over time.

The program can evaluate values sampled at any frequency, as long as there is no more than one value per day. If there is more than one value per day, the user should take a mean or median, or somehow reduce the data to one value per day. The program is not currently set up to handle the variance calculation for multiple observations per day, and may give unpredictable results for these types of data.

SEASONAL MANN-KENDALL TEST

In the program, seasons are currently based on the wet and dry seasons in SJRWMD; however, other seasons can be assigned as needed. Wet season runs from June to October, while dry season runs from November to May (Rao et al. 1989). The seasonal Kendall test works on the same principle as the Mann-Kendall test, the main difference being that the Kendall K is calculated for each season and then summed, and the variance calculation is more complex.

The seasonal test accounts for the fact that there may be significant differences between data obtained in one season vs. another. For example, the data values for a given analyte at a particular station may generally be lower in the dry season than in the wet. The seasonal test accounts for this by taking differences only among values in the same season. The user must define the seasons, and the program will automatically test for differences among values from each season. The user has the option of modifying the significance level of the test for seasonality. The seasonality is evaluated using the Wilcoxon test, which is a non-parametric test available in the SAS procedure NPAR1WAY (SAS 1989). If there is seasonality, the program automatically assigns the dataset for that station to the seasonal trend test. However, there may not be seasonality, which means that even though seasons are assigned, a seasonal trend calculation will not necessarily be performed. Different seasons can be assigned at the appropriate place in the program. The Kendall K is calculated as before, but for each season. So if there are four seasons defined, there will be four separate K values. These K values are then summed, and a variance is calculated. The variance equation for the seasonal test is

$$\mathrm{var}(K)=\frac{1}{18}\left[n_i(n_i-1)(2n_i+5)-\sum_{p=1}^{g_i}t_{ip}(t_{ip}-1)(2t_{ip}+5)-\sum_{q=1}^{h_i}u_{iq}(u_{iq}-1)(2u_{iq}+5)\right]$$
$$+\frac{\sum_{p=1}^{g_i}t_{ip}(t_{ip}-1)(t_{ip}-2)\sum_{q=1}^{h_i}u_{iq}(u_{iq}-1)(u_{iq}-2)}{9n_i(n_i-1)(n_i-2)}$$
$$+\frac{\sum_{p=1}^{g_i}t_{ip}(t_{ip}-1)\sum_{q=1}^{h_i}(u_{iq}-1)}{2n_i(n_i-1)}$$

The variance equation accounts for the number of tied values (t) and the number of groups of tied values (g) across all years in each season. Additionally, the variance equation accounts for the number of years in each season that have multiple data (h), and the number of multiple data in each year (u) (Gilbert 1987). The variance is calculated for each season and the variances are summed. A z-score is calculated as before. For the seasonal test, all z-scores are evaluated using the normal probability distribution ($\alpha \le 0.1$). One major difference between the seasonal and non-seasonal test is that a z-score will be calculated even when n is less than 40.

SEN SLOPE

The Sen slope (Gilbert 1987) calculation is a recent addition to the program. It is determined along with the Kendall trend, and is the median of all differences between successive data values. A 90% confidence interval around the slope is obtained using the simplified procedure found in Gilbert (1987). The user can modify the confidence interval at the appropriate place in the program. Appropriate adjustments are applied to result in an estimate of the annual change in units for each station evaluated. Currently, the Sen slope will not produce confidence limits if there are tied values in the dataset.

USING THE PROGRAM

DISTRICT NETWORK

Since the program was developed at SJRWMD, there are things about the program that are specific for SJRWMD. At SJRWMD, there are really two computer networks. One is the personal computer (PC) network with which most SJRWMD employees are familiar. The PC network consists of hundreds of desktop PCs all linked through fiber optic cable to servers, which essentially act as huge disk drives. Locally stored Microsoft Office products are usually run on these PCs, and work files are stored on the network drives. The other network at SJRWMD is the UNIX network, which has the opposite setup. On the UNIX network, programs, data, and work files are stored on the Unix servers, and the desktop PC essentially acts as a dumb terminal. Although SAS Institute produces both a UNIX version of SAS and a PC version of SAS, SJRWMD currently uses the UNIX version. Thus, all analyses are done on the UNIX server, and the programs described in this report are designed to run on the Unix server. Should SJRWMD ever completely convert to a PC network, users would use PC SAS, and filenames in the program would have to be modified. However, a knowledgeable user should have no problem modifying the program to run on PC SAS. Sophisticated computer users know that there are shortcuts and other ways of setting up files on both networks. The author does not insist on any particular way of using the computer networks at SJRWMD, but only provides the following suggestions to help the average user gain maximum benefit from the trend test program.

LOGGING ON

In order to use SAS at SJRWMD, the user must log on to the UNIX network. For most people, that means starting up the EXCEED software on the PC and typing in your username and password at the prompt. If you don't have a username, see one of the IR administrators, or call the help desk.

OBTAINING THE PROGRAM

The program (*mann_kendall_sen.sas*), a test dataset (*testset.txt*), and a result dataset (*kendall_results.txt*) can be found in the */home/swinkler/kendall_test* directory. The user can copy these files from the directory or load them from the CD-ROM associated with this report.

TRANSFERRING DATA

As a result of having two networks, any data that is on the PC network has to be transferred to the UNIX network in order for the program to be used. There are a number of ways to accomplish this. Text files can be transferred using ftp programs. Excel files can be saved to the user's UNIX directory using the *save as* command in the *file* pull-down menu. Files can also be moved using the Microsoft Windows Explorer program. I recommend that the user create a folder in their work directory on the UNIX network to store the programs as well as their datasets for analysis. This can be done using pull-down menus or UNIX commands.

I highly recommend that users apply the *dos2unix* command in UNIX to their datasets to get rid of any end-of-record characters (especially ^M) that PC-DOS might add on the end of each record. The command is easy to use. Open a terminal window in UNIX and change directories (if needed) using the *cd* command to be sure you are in the directory where your datasets are located. Once in the proper directory, run the *dos2unix* command. The command is:

dos2unix filename1 filename2

For example, to run the command on a dataset called testset.txt type:

dos2unix testset.txt testset.txt

In the previous example, the *dos2unix* command will write a new version of testset.txt that overwrites the old version. Using the *dos2unix* command will help reduce unpredictable results when using datasets that have been stored on the PC network.

FILE FORMAT

The program will work with just one station's data, or many. However, the file format has to be structured as a comma-delimited format that contains the following items on every line: **station, date, value**. The date must be in **yyyymmdd** format. SAS does have a facility for importing datasets, and knowledgeable users can take advantage of it. The following example dataset shows how the input file needs to be set up:

station 1, date1, value
station 1, date2, value
station 1, date3, value
station 1, date4, value
station 2, date1, value
station 2, date2, value
station 2, date3, value
etc.

For an actual test dataset used as input, see Appendix 1, or see the file *testset.txt* on the */home/swinkler/kendall_test* directory in UNIX. If there are more than 400 values for a station, the program will have to be modified as indicated in Appendix 2. Blank spaces are allowed between fields, but not required.

DATA REQUIREMENTS

The program has no requirements on the data other than that there be no more than one value per day for each station. Also, the program has no provisions for data screening or cleanup. The user must modify the input dataset as needed to account for sample codes, outliers, and any other data issues. For example, the program will calculate trends even if there are missing years of data. Also, uneven sampling frequencies are not accounted for in the program. The user may want to take monthly means, seasonal means, or medians in these cases. In any event, the user must verify whether the datasets used are sufficient for their purpose. The author recommends that users consult Gilbert's (1987) text and Helsel and Hirsch (1995) for more information on data preparation and usage of the Mann-Kendall test.

USING SAS

Once on UNIX, the user needs to select the common desktop environment mode and open up a terminal window. In that window, the user can type the following to bring up the SAS operating environment:

sas

Several SAS windows will pop up, but the program editor and log window are usually the most important. In the SAS program editor, there is a pull-down menu under *file*, and by choosing *open*, the SAS program *mann_kendall_sen.sas* can be selected from the user's directory. The program editor will import the program, and then a click on the *run* pull-down menu will allow the user to click on *submit* to run the program. Results will be written to the output file specified in the *filename data_out* statement of the program. The user should keep an eye on the log file while the program runs, and look for any errors. Any errors will be in red, and will usually require a fix of some sort.

PROGRAM RESULTS

The program outputs a comma-delimited file (Table 2).

Table 2. Output from program

Output Field	Description
station	Site that was evaluated
trend_n	Total number of raw data values used for the trend test
trend_years	Number of years of data used for the trend test
kendall_sum	The Kendall K statistic, or the sum of all differences between values
zscore	The critical value when the normal approximation of the test is used
zprob	The probability of getting the z-score if there was no trend
trend	Qualitative description of the trend based on $\alpha \leq 0.1$
season	Specifies whether the seasonal or non-seasonal trend test was used
sen_slope	Estimated annual change in the data
lo_confidence	Low 90% confidence limit for the Sen slope
high_confidence	High 90% confidence limit for the Sen slope

It is important to note that whenever the seasonal test is used, a z-score and z-prob will be output. If the non-seasonal test is used, there will be a z-score if trend_n is greater than 40. If trend_n is less than or equal to 40, there will not be a z-score or z-trend, because the K value itself will be evaluated. In addition, a Sen slope estimate is always calculated, but confidence limits are not. No confidence limits are reported for a non-seasonal test with $n \leq 40$. Also, the program currently will not produce confidence limits if there are ties in the dataset. Finally, it is best not to have multiple measurements on the same day. While it wouldn’t affect the seasonal calculations, the program is not set up to automatically calculate the ‘multiple observations per time period’ variance when calculating non-seasonal Kendall sums. Please consult Appendix 2 and the ‘user modification’ section for more detailed notes and for instructions on how to make program modifications.

SAMPLE OUTPUT

The sample output in Table 3 was generated from the input dataset in Appendix 1. In the output, four water quality sites were evaluated for trend in a water quality index for streams and a trophic state index for two lakes. These were Lake Ashby (ASH), Crescent Lake (CRESLM), Durbin Creek at Racetrack Road (LSJ087), and the St. Johns River at Popo Point (PP62). These stations were chosen for illustrative purposes only and don't necessarily reflect current water quality at the sites.

Table 3. Sample program output

station	trend_n	trend_ years	kendall_ sum	zscore	zprob	trend	season	sen_ slope	lo_ confidence	hi_ confidence
ASH	59	10	446	2.910	0.004	increasing	non-seasonal	0.840	0.449	1.259
CRESLM	23	4	3			insignificant	non-seasonal	0.081		
LSJ087	51	10	–108	–1.707	0.088	decreasing	seasonal	–0.880	–1.900	–0.014
PP62	25	4	34	1.506	0.132	insignificant	seasonal	1.833	0.201	3.601

ASH

The output shows that Lake Ashby had 59 trophic state index values upon which trend analysis was performed. These 59 values represented 10 years of data. The K statistic (Kendall Sum) is 446, which means that there were 446 more positive differences than negative differences. Since there were more than 40 measurements, a z-score of 2.910 was calculated, and the probability of getting such a z-score, if in fact there was no trend, was 0.004. In other words, there is a 0.4% or 1 in 250 chance that there really is no trend at Lake Ashby, given these data. Since the program is set up for a significance level of 0.1, the trend was considered significant, and since the K statistic is positive, numbers in general are increasing, so there is an increasing trend. The Sen slope estimate is 0.840 units / year, and there is a 90% probability that the true slope is between 0.449 units / year and 1.259 units / year. Although wet and dry seasons were assigned, the Wilcoxon seasonal test determined no seasonality existed in the dataset, so ASH was evaluated using the non-seasonal test.

CRESLM

Crescent Lake had only 23 trophic state index values, and since there was no significant difference between the wet and dry season values, the station was evaluated using the non-seasonal test. The 23 values were spread out over 4 years, and there were three more positive differences than negative. This would seem to indicate an increasing trend. However, tabled critical values at an $\alpha \leq 0.1$ reveal that the trend is insignificant, and in fact, from Table A30, the Kendall K would have to exceed 64 in order for the trend to be significant. Since $n = 23$ and thus $n < 40$, no z-score was calculated, so there are no confidence limits associated with the Sen slope estimate of 0.081 units per year.

LSJ087

There were 51 water quality index values taken over a 10-year period at Durbin Creek, and the net sum of differences is –108. Since the K is negative, values are decreasing over time, and in this case, there was a seasonal difference between wet and dry season values. The resulting z-score was –1.707, and the probability of getting such a z-score if in fact the trend was 0 was less than 1 in 10 ($p = 0.088$). Thus, this is a significant, decreasing trend with an estimated –0.880 units / year decrease, and 90% of the time, the decrease will be in the range of –1.900 to –0.014.

PP62

Finally, the St. Johns River at Popo Point had an insignificant trend in water quality index values, even though the Sen slope estimate is fairly large at 1.833 units per year. Even though there were fewer than 40 measurements, a z-score was calculated because the seasonal test option was applied. Had there been no difference between the wet and dry season data, the non-seasonal test option would have applied, and no variance, z-score, or confidence limits on the Sen slope would have been calculated.

USER MODIFICATIONS

As mentioned throughout this report, there are several modifications that a user can make to the program. While a working knowledge of SAS would be helpful, no special techniques are required to change the program. The user is advised to open the program in a text editor, make the appropriate changes, save the program, and import it to the SAS program editor window to run it. Specific details on modifications are in the commented sections of the program in Appendix 2. Of course, a SAS-savvy user could make many more modifications than are mentioned here.

DATASET-SPECIFIC MODIFICATIONS

Modifying the Input and Output Filenames in the Program

The program creates a SAS dataset from the specified input file. The input filename needs to be modified to reflect the user's directory and filename. For example, for an input file called samples1.dat, the user would type:

filename data_in '/home/user directory/samples1.dat';

Note that the goal here is to tell SAS where your input file is. The output file also needs to be specified. SAS will not create an output file until the program runs, but it must be named so that SAS will know what to call it and where to put it. For example, if the user wants an output file called *trend_result.dat*, the user could type the following at the output statement line in the program:

filename data_out '/home/user directory /trend_result.dat';

See sections 1 and 2 of the program in Appendix 2 for further instructions.

Array Size

If there are more than 400 measurements for any given stations, the arrays which hold the data will need to be modified. Make the modification as indicated in section 5B and section 6C of the program in Appendix 2.

OTHER MODIFICATIONS

Seasons

Seasons must be assigned to the values in the dataset, since the program doesn't know what seasons are appropriate for the user's data. The default programming is for two seasons, wet and dry. The user can select 12 seasons (monthly), four seasons, or no seasons, as needed. See section 4B in Appendix 2, where seasons can be uncommented if change is wanted. The variable 'mo' stands for month.

Significance Level of the Seasonal Test

The program evaluates whether there are any differences between data from different seasons. For example, for a wet and dry season option, the program evaluates, for every station, whether the data associated with the dry season are statistically different than those associated with the wet season. The Wilcoxon test is used for the comparison, and it ranks the data by magnitude. The sums of the ranked data for each season are then used to calculate a Wilcoxon statistic, and if the statistic exceeds the critical value at $\alpha \leq 0.1$, then seasonality is present in the data. See section 4C in the program for details on changing the significance level.

Significance Level of the Non-seasonal Kendall Test for $n \leq 40$

The user is referred to section 5C. The program is currently programmed for a significance level of 0.1. The critical values in the array 'crits' are from Table A30 and correspond directly with the *n* value found in array 'nsize'. The user can consult table A30 and put in a different set of critical values if a different significance level is needed. Just remember to select values that are $\alpha/2$ if a two-tailed test is required. For example, if $\alpha = 0.2$ is desired, select values from table A30 where the $p = 0.1$. This is because, for a given critical value, probabilities in Table A30 are doubled for a two-tailed test.

Changing the Significance of the Non-seasonal Kendall Test When $n > 40$

If the user wants to change the significance level of the non-seasonal test, they should consult section 5D. The default is 0.1, but any value can be entered. The value being compared is the *p* value (or probability) associated with the z-score. The probability is the chance of getting the calculated z-score if in fact there was no trend. For example, changing the alpha value to 0.05 would mean a 95% significance level. The user would be saying that they would accept a trend result only if there were a less than or equal to 1 in 20 chance of getting such a result if in fact there was no trend.

Changing the Significance Level of the Seasonal Test

The significance level of the seasonal test can be changed in section 6D. The default is 0.1. Refer to the previous section for an explanation of what the *p* value and the significance level mean.

Changing the Confidence Intervals Around the Sen Slope Estimate

In section 7B, the user can change the confidence intervals around the Sen slope. For the default 90% interval, the value is 1.645. Should the user desire some other interval, the number can be changed. The number is the critical value associated with the normal distribution for a two-tailed test. For example, for a significance level of 0.2, the critical value is 1.816. For $\alpha = 0.05$, the value would be 1.96, and for $\alpha = 0.01$, the value is 2.5758. Again, the confidence interval for $\alpha = 0.1$ simply means that interval which has a 9 out of 10 chance of containing the true slope.

REFERENCES

Gilbert, R.O. 1987. *Statistical methods for environmental pollution monitoring*. New York: Van Nostrand Reinhold.

Helsel, D.R., and R.M. Hirsch. 1995. *Statistical methods in water resources. Studies in Environmental Science 49*. Netherlands: Elsevier Science B.V.

Hollander, M., and D.A. Wolfe. 1999. *Nonparametric statistical methods*. New York: John Wiley and Sons.

Rao, D.V., S.A. Jenab, and D.A. Clapp. 1989. *Rainfall analysis for northeast Florida, Part 3: Seasonal rainfall data*. Technical Publication SJ89-1. Palatka, Fla.: St. Johns River Water Management District.

SAS Institute Inc. 1989. *SAS/STAT user's guide*. Version 6, fourth edition. Vol. 2. Cary, N.C.

APPENDIX 1 — SAMPLE INPUT DATASET

station,date,value

ASH,19870409,35.4
ASH,19870604,58.1
ASH,19870806,48.1
ASH,19871007,45
ASH,19871203,41.9
ASH,19880204,43.4
ASH,19880607,46
ASH,19880802,39.9
ASH,19881006,51.3
ASH,19881205,50.9
ASH,19890131,55.4
ASH,19890330,41.7
ASH,19890720,46.4
ASH,19950508,64.7
ASH,19950724,39.5
ASH,19950912,6.4
ASH,19951107,55.6
ASH,19960130,46.8
ASH,19960313,40.8
ASH,19960528,4.6
ASH,19960717,-12.7
ASH,19960925,55.7
ASH,19961119,42.5
ASH,19970128,59.5
ASH,19970303,57.2
ASH,19970505,45.8
ASH,19970804,50.2
ASH,19970911,56.7
ASH,19971103,48.1
ASH,19971216,52.7
ASH,19980218,40.8
ASH,19980330,41.4
ASH,19980518,44.8
ASH,19980707,45.9
ASH,19981005,61.4
ASH,19981103,61.6
ASH,19981207,59.8
ASH,19990126,86.2
ASH,19990218,58.7
ASH,19990317,56.9
ASH,19990405,60
ASH,19990505,54.6
ASH,19990602,55.2

station,date,value

ASH,19990701,55.3
ASH,19991215,62.6
ASH,20000110,62.2
ASH,20000207,58.7
ASH,20000309,61.6
ASH,20000410,63.2
ASH,20000606,54.8
ASH,20000719,55.1
ASH,20000817,52.1
ASH,20000925,51.7
ASH,20001017,52.4
ASH,20001115,51.6
ASH,20001214,51.9
ASH,20010110,50.4
ASH,20010206,52
ASH,20010308,51.9
CRESLM,19970430,59.8
CRESLM,19970625,61.2
CRESLM,19971222,52.3
CRESLM,19991221,60.4
CRESLM,20000329,65.2
CRESLM,20000426,74.1
CRESLM,20000524,57.3
CRESLM,20000629,55.9
CRESLM,20000727,68.1
CRESLM,20000824,75.1
CRESLM,20000928,68.5
CRESLM,20001026,63.4
CRESLM,20001127,60.6
CRESLM,20001221,57.2
CRESLM,20010123,62.2
CRESLM,20010227,59.7
CRESLM,20010329,71.1
CRESLM,20010501,61.6
CRESLM,20010530,64.3
CRESLM,20010628,67.9
CRESLM,20010723,65.3
CRESLM,20010925,56
CRESLM,20011024,43.8
LSJ087,19920407,58.6
LSJ087,19920713,65.8
LSJ087,19921012,59.3
LSJ087,19930104,63.9

station,date,value

LSJ087,19940110,38.9
LSJ087,19940321,68.8
LSJ087,19940523,64.6
LSJ087,19940728,53.9
LSJ087,19941003,64.1
LSJ087,19941107,59.5
LSJ087,19950110,45.7
LSJ087,19950301,70.8
LSJ087,19950607,77.2
LSJ087,19950706,67.6
LSJ087,19950907,58
LSJ087,19951102,66.9
LSJ087,19960123,46.9
LSJ087,19960305,50.5
LSJ087,19960522,50.6
LSJ087,19960731,50
LSJ087,19960912,58.6
LSJ087,19961121,59.4
LSJ087,19970317,59.8
LSJ087,19970520,65.8
LSJ087,19970724,56.8
LSJ087,19970904,65.5
LSJ087,19971117,45.1
LSJ087,19980105,46.1
LSJ087,19980318,54.8
LSJ087,19980507,38.7
LSJ087,19980729,49
LSJ087,19980909,66.8
LSJ087,19981022,54.8
LSJ087,19981231,36.7
LSJ087,19990222,47.2
LSJ087,19990429,49
LSJ087,19990629,56.5
LSJ087,19990819,45.1
LSJ087,19991013,65.5
LSJ087,19991202,60
LSJ087,20000203,37.1
LSJ087,20000406,48.1
LSJ087,20000620,51.1
LSJ087,20000822,50.6
LSJ087,20001003,65.7
LSJ087,20001206,55.3
LSJ087,20010201,48.6
LSJ087,20010404,64.5
LSJ087,20010625,54.2
LSJ087,20010822,65.2

station,date,value

LSJ087,20011017,70.1
PP62,19930922,61
PP62,19931020,57.9
PP62,19931130,51.9
PP62,19931228,54.9
PP62,19940126,60.7
PP62,19940223,58.9
PP62,19940330,56.8
PP62,19940420,55.5
PP62,19940929,59
PP62,19941024,52.4
PP62,19941130,50.5
PP62,19941228,48.7
PP62,19950215,53.5
PP62,19950329,52.9
PP62,19950502,57.6
PP62,19950629,68
PP62,19950727,64.4
PP62,19950831,61.3
PP62,19951024,57.8
PP62,19951121,55.3
PP62,19960130,46.5
PP62,19960327,62.9
PP62,19960528,60.2
PP62,19960702,69.7
PP62,19961001,63.9

APPENDIX 2 — THE SAS PROGRAM

```
/****************************************************************************************************/
/* A SAS VERSION OF THE MANN-KENDALL TEST, PROGRAMMED BY STEVE WINKLER*/
/* SUMMER 2002. SEN SLOPE MODIFICATION ADDED OCTOBER 2003                    */
/*                                                                           */
/* The following program evaluates the input dataset values for trends       */
/* at each station, and also calculates a Sen slope estimate. The            */
/* Wilcoxon test is used to determine whether seasonality is present in      */
/* the input dataset at a<=0.1. The significance level can be changed        */
/* if needed. The user must define the seasons (see below) which can         */
/* vary from the wet and dry seasons currently used to monthly seasons.      */
/* Stations with seasonality are evaluated for trend using the seasonal      */
/* Kendall test, and those without seasonality are evaluated with the        */
/* Kendall test. The annual Sen slope is calculated for all stations.        */
/* 90% confidence limits are also calculated for those stations where a      */
/* normal approximation of the variance is performed. Practically            */
/* speaking, this means that stations with 40 or fewer values and            */
/* non-seasonality will only have a Sen slope, with no confidence            */
/* limits. All other stations will have confidence limits. The output        */
/* file contains the following values:                                       */
/*                                                                           */
/* station: the station                                                      */
/* trend_n: number of raw data values input                                  */
/* trend_yrs: the number of years of raw data                                */
/* kendall_sum: kendall K statistic                                          */
/* z-score: the critical value for a normal approximation                    */
/* z-prob: probability of the zscore                                         */
/* trend: type of trend                                                      */
/* season: states whether seasonality is present                             */
/* sen_slope: annual change in whatever units were input                     */
/* lo_confidence: low 90% confidence limit for the sen slope                 */
/* high_confidence:  high 90% confidence limit for the sen slope             */
/*                                                                           */
/* If there are more than 400 values for a station, the arrays will          */
/* have to be modified to hold them (see section 5b and 6c). Finally, it     */
/* is best not to have multiple measurements on the same day. While it       */
/* wouldn't affect the seasonal calculations, the program is not set up      */
/* to automatically calculate the 'multiple observations per time period'    */
/* variance when calculating non-seasonal kendall sums.                      */
/****************************************************************************************************/
```

```
/******************************************************************************************************/
/* SECTION 1.                                                                                        */
/* modify the following 'filename data_in' line to identify your inpu                               */
/* file. the input file needs to be in the order of station, date, value                            */
/* with all fields separated by commas. in addition, leave a header line                            */
/* in the file so that the first data record is on line 2                                           */
/******************************************************************************************************/

filename data_in   '/home/swinkler/kendall_report/testset.txt';

/******************************************************************************************************/
/* SECTION 2.                                                                                        */
/* modify the following 'filename data_out' line to identify your output                            */
/* file. the result file will be comma delimited with a header line                                 */
/******************************************************************************************************/

filename data_out '/home/swinkler/kendall_report/kendall_results.txt';

/******************************************************************************************************/
/* SECTION 3. READ THE DATA FILE                                                                     */
/* the following section of code reads in your datafile and puts it into a                          */
/* sas dataset.                                                                                      */
/******************************************************************************************************/

data test ;
 length station $ 16 ;
 infile data_in  truncover dlm=',' dsd firstobs=2;
 input station $ date value;
run;

/******************************************************************************************************/
/* SECTION 4A. DETERMINE SEASONALITY                                                                 */
/* determine whether seasonality exists for the stations by using the                               */
/* wilcoxon function in the npar1way procedure. the program currently                               */
/* defines a dry season as running from november 1 to may 31 and a wet                              */
/* season from june 1 to october 31                                                                  */
/******************************************************************************************************/

data test yrtrend (keep= station yr); set test;

 chardate= put (date, z8.);
 sasdate= input (chardate , yymmdd8.);
 mo= month(sasdate);
 yr=year(sasdate);
```

```
/*************************************************************************************************/
/* SECTION 4B.                                                                                  */
/* you can add additional seasons in this section, but if there are more                         */
/* than 2 seasons the proc npar1way will calculate a kruskal-wallis                              */
/* statistic. however it will not tell you which of the season(s) is                             */
/* different from the others. Uncomment the lines you want, but remember                         */
/* to comment the ones you don't want. Comment means to add asterisks and                        */
/* slashes to the beginning and end of each line, while uncomment means to                       */
/* remove these characters from the beginning and end of each line.                              */
/* SAS will not read commented lines as program code.                                            */
/*************************************************************************************************/

/* if you want monthly seasons, uncomment the following lines */

/* if mo=1 then season='jan'; */
/* if mo=2 then season='feb'; */
/* if mo=3 then season='mar'; */
/* if mo=4 then season='apr'; */
/* if mo=5 then season='may'; */
/* if mo=6 then season='jun'; */
/* if mo=7 then season='jul'; */
/* if mo=8 then season='aug'; */
/* if mo=9 then season='sep'; */
/* if mo=10 then season='oct'; */
/* if mo=11 then season='nov'; */
/* if mo=12 then season='dec'; */

/* if you want four seasons, uncomment the following lines */

/* if mo>=1 and mo<=3 then season='win'; */
/* if mo>=4 and mo<=6 then season='spr'; */
/* if mo>=7 and mo<=9 then season='sum'; */
/* if mo>=10 and mo<=12 then season='fal'; */


/* the program is currently set up for 2 seasons */

if mo>=1 and mo<=5 then season='dry';
if mo>=6 and mo<=10 then season='wet';
if mo>=11 and mo<=12 then season='dry';

/* uncomment the following if you don't want seasons */

/* season='non';  */

 drop chardate;
run;

proc sort data=yrtrend noduplicates; by station yr; run;
```

```
proc univariate data=yrtrend noprint;
 var yr;
 by station;
 output out=numyrs n=nyrs;
run;

proc sort noduplicates data=test; by station; run;

data test ; merge test numyrs; by station ; run;

proc npar1way data=test noprint;
 by station;
 class season;
 var value;
 output out=wiltest wilcoxon;
run;

data wiltest; set wiltest;
 length climate $ 15;

/*****************************************************************************************************/
/* SECTION 4C.                                                                                      */
/* modify the following statement if you want to change the significance                            */
/* level for the wilcoxon test. the test is a two-sided test and the                                */
/* original program uses a significance level of 0.10, or 90%. For                                  */
/* example, if 0.1 is changed to 0.2, then you have an 80% confidence                               */
/* level. Note that any changes here only affect the significance level of                          */
/* the seasonality test. For changes to the significance level of the                               */
/* kendall trend test, see section 5c, 5d, and 6d.                                                  */
/*****************************************************************************************************/

 if p2_wil <= 0.10 then climate='seasonal';
  else climate='nonseasonal';
 keep station climate;
run;

proc sort data=wiltest; by station; run;

data test; merge test wiltest; by station; run;

data seas nonseas; set test;
 drop climate;

 if climate='seasonal' then output seas;
 else output nonseas;

run;
```

```
/*********************************************************************************************************/
/* SECTION 5A. NON-SEASONAL KENDALL TEST                                                                */
/* the following section of code calculates the mann kendall test on                                    */
/* the nonseasonal dataset.                                                                              */
/*********************************************************************************************************/

data test; set nonseas; run;

proc sort data=test noduplicates; by station date; run;

proc means data=test noprint;
 by station;
 var value;
 output out=enns n=n;
run;

proc sort data=enns; by station; run;

data test; merge test enns; by station; run;

proc sort data=test; by station date; run;

data calc sen (keep = station q count) ; set test; by station;

 if first.station=1 then do;

 place = _n_ - 1;

/*********************************************************************************************************/
/* SECTION 5B. If you have more than 400 values for a station, the                                      */
/* following arrays will need to be expanded. For example, if you                                       */
/* have 500 values, change the line 'array val (400) x1 - x400 ' to                                     */
/* 'array val (500) x1 - x500'. Then, change the do loop from                                           */
/* '1 to 400' to '1 to 500'. Do the same thing for the days array.                                      */
/*********************************************************************************************************/

 array val (400) x1-x400;
 do i = 1 to 400;
  val (i)= .;
 end;

 array days (400) d1-d400;
  do i= 1 to 400;
   days (i) =.;
  end;

   do i= 1 to n;
    dex=i+place;
    set test point= dex;
```

```
    val (i)=value;
    days (i)= sasdate;
   end;

sum=0;
count=0;
 do c=2 to n;
  do i=1 to c-1;
   diff= val(c) - val(i);
   daydiff= days(c) - days(i);
   if daydiff=0 then sign=0;
   else do;
    if diff<0 then sign= -1;
    if diff=0 then sign= 0;
    if diff>0 then sign= 1;
    q= (diff/  daydiff)*365;
    count=count+1;
    output sen;
   end;
   sum=sum+sign;
  end;
 end;

end;
retain place;
output calc;
run;

/* determine the median slope value */

proc sort data=sen; by station; run;

proc univariate noprint data= sen;
 by station;
 var q count;
 output out=med_q median=median_q medcount max= max_q max_count;
run;

/* rank the sen slopes so that confidence intervals can be calculated */

proc rank data=sen out=sen_rank;
 by station;
 var q;
 ranks q1;
run;

proc sort data=sen_rank;
 by station q1; run;
```

```
/* back to trend calculations */

data calc; set calc (where=(sum^=.));
 keep station n sum nyrs;
run;

/*****************************************************************************************************/
/* SECTION 5C.                                                                                       */
/* The array labeled 'crits' contains the approximate critical values                               */
/* (from table A30, Hollander and Wolfe) for the Kendall K statistic                                */
/* at a<=0.1. Since the program tests for a positive or negative trend,                             */
/* the test is 2-tailed, and thus values are chosen from the table                                  */
/* where the probability is a/2 or p=0.05.                                                          */
/* Each value in the array 'crits' corresponds to the n value in the                                */
/* array 'nsize'. For example, the critical value for an n of 15 is 34.                             */
/* All these critical values can be changed should the user desire a                                */
/* different confidence level. The user should consult table A30 for                                */
/* the critical values to use at a different confidence level. Also see                             */
/* Gilbert 1987 for more information.                                                               */
/*****************************************************************************************************/

data calc2; set calc;
 length qtrend $ 15;

 array crits (37) _temporary_ (5 7 10 12 15 17 20 22 25 28
           31 34 37 41 44 48 51 55 60 64 67 71 76 80 85
           89 94 98 103 107 112 117 122 127 132 137 142);

 array nsize (37) _temporary_  (4 5 6 7 8 9 10 11 12 13 14
          15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
          31 32 33 34 35 36 37 38 39 40 );

   do i=1 to 37;
    if n=nsize(i) then do;
     if abs(sum)>=crits(i) then do;
      if sum>0 then qtrend='increasing';
      if sum<0 then qtrend='decreasing';
     end;
     else qtrend='insignificant';
    end;
   end;

 if n<4 then qtrend='insufficient';

run;

/* rank the values in the dataset so that ties can be evaluated   */
/* in order to calculate the variance. read from the test dataset */
```

```
proc rank data=test out=rtest ;
 by station;
 var value;
 ranks rvalue;
run;

proc sort data=rtest; by station rvalue; run;

proc univariate data=rtest noprint;
 by station rvalue;
 var rvalue;
 output out=mranks n=rank_n;
run;

data mranks; set mranks;
 if rank_n>1;
run;

proc univariate data=mranks noprint;
 by station;
 var rank_n;
 output out=mstat n=stat_n;
run;

data combrank; merge calc2 mranks mstat; by station;
 keep station n rank_n stat_n sum qtrend nyrs;
run;

data over40 (keep= station n rank_n stat_n sum nyrs)
     under40 (keep= station n sum qtrend nyrs);
 set combrank;
 if n<=40 then output under40;
 else output over40;
run;

data over40; set over40;
 if stat_n=. then stat_n=0;
 if rank_n=. then rank_n=0;
run;

proc sort data=over40 ; by station ;run;

/* calculate the variance */

data over40; set over40; by station;
  length qtrend $ 15;

 if first.station=1 then do;
```

```
 place = _n_ - 1;

  subsum=0;
  do i= 1 to stat_n;
   dex=i+place;
   set over40 point= dex;
   subvar= rank_n*(rank_n - 1)*(2*rank_n + 5);
   subsum=subsum+subvar;
  end;

 var_ns= (n*(n-1)*(2*n + 5) -subsum) / 18;

 if sum>0 then do;
  zscore= (sum-1) / (var_ns**0.5);
 end;

 if sum = 0 then zscore=0;

 if sum<0 then do;
  zscore= (sum+1) / (var_ns**0.5);
 end;

 p= (1-probnorm(abs(zscore)))*2;

/*****************************************************************************************************/
/* SECTION 5D.  modify the following statement for a different                                      */
/* probability to test for non-seasonal trend. The program is originally                            */
/* set for 0.10. For example, changing the 0.1 to 0.2 means an 80%                                  */
/* significance level.                                                                              */
/*****************************************************************************************************/

 if p<=0.10 then do;
  if sum>0 then qtrend='increasing';
  if sum<0 then qtrend='decreasing';
 end;
 else qtrend='insignificant';

end;

retain place;

run;

data result; set over40 under40;
 if qtrend^=' ';
 calcmeth='non-seasonal';
 keep station n sum qtrend zscore p calcmeth nyrs var_ns;
run;
```

```
proc sort data=result noduplicates; by station; run;

/*****************************************************************************************************/
/* SECTION 6A. SEASONAL KENDALL TEST                                                                 */
/* the following code calculates the kendall statistic on seasonal data                             */
/* as previously defined. the first step is to get the values needed to                             */
/* calculate the variance. first start by ranking the values in the                                 */
/* dataset so that ties can be evaluated                                                             */
/*****************************************************************************************************/

proc sort data=seas; by station season; run;

/* determine the number of data values per season, this is n */

proc means data=seas noprint;
 by station season;
 var value;
 output out=size n=n;
run;

/* rank the dataset in order to determine which values are ties */

proc rank data=seas out=rtest ;
 by station season;
 var value;
 ranks rvalue;
run;

proc sort data=rtest; by station season rvalue; run;

/* determine the frequency of each rank, the rank is in variable rvalue */

proc univariate data=rtest noprint;
 by station season rvalue;
 var rvalue;
 output out=mranks n=t;
run;

/* keep only those ranks >1, since these are the ties or t */

data mranks; set mranks;
 if t>1;
run;

/* determine how many groups of multiple ranks there are, this is g */

proc univariate data=mranks noprint;
 by station season;
 var t;
```

```
 output out=mstat n=g;
run;

/* calculate the number of years with more than one value to determine u */

proc sort data=seas; by station season yr; run;

proc means data=seas noprint;
 by station season yr;
 var value;
 output out=dat_yr n=u;
run;

/* only keep the years with more than 1 value per year */

data dat_yr; set dat_yr;
 if u>1;
run;

/* determine the number of years in each season that */
/* have a u >1, this is h                            */

proc means data=dat_yr noprint;
 by station season;
 var yr;
 output out=yrs n=h;
run;

/* combine all these variance measures to get a dataset   */
/* that has all the information to calculate the variance */

data combtest; merge size mranks mstat dat_yr yrs; by station season;
 keep station season n t g u h ;
run;

/*****************************************************************************************************/
/* SECTION 6B. SEASONAL VARIANCE CALCULATION                                                         */
/*****************************************************************************************************/

proc sort data=combtest; by season station; run;

data calcvar; set combtest;
 if g=. then g=0;
 if h=. then h=0;
  if n<3 then delete;
run;

data calcvar; set calcvar; by season station;
```

```
 if first.station then do;

   place= _n_ - 1;

   subg=0;
   subg1=0;
   subg2=0;
   do i= 1 to g;
    dex=i+place;
    set calcvar point= dex;
    subvg= t*(t - 1)*(2*t + 5);
    subvg1= t*(t-1)*(t-2);
    subvg2= t*(t-1);
    subg=subg+subvg;
    subg1=subg1+subvg1;
    subg2=subg2+subvg2;
   end;

   subh=0;
   subh1=0;
   subh2=0;
   do i= 1 to h;
    dex2= i+place;
    set calcvar point= dex2;
    subvh= u*(u-1)*(2*u+5);
    subvh1= u*(u-1)*(u-2);
    subvh2= u*(u-1);
    subh= subh+subvh;
    subh1=subh1+subvh1;
    subh2=subh2+subvh2;
   end;

  variance= ( ( n*(n-1)*(2*n + 5) -subg -subh) / 18 )
           + ( (subg1*subh1) / (9*n*(n-1)*(n-2)) )
           + ( (subg2*subh2) / (2*n*(n-1)) );
end;

retain place;

run;

/* keep only the observations with a variance */

data calcvar; set calcvar (where=(variance^=.));
 keep station season variance;
run;

/* sum the variances over seasons */
```

```
proc sort data=calcvar; by station; run;

proc means data=calcvar noprint;
 by station;
 var variance;
 output out=sumvars sum=sumvar;
run;

/****************************************************************************************************/
/* end the variance calculation, and set up the dataset to start                                    */
/* calculating the kendall statistic                                                                */
/****************************************************************************************************/

data seas; merge seas size; by station season;
 drop _type_ _freq_;
run;

proc sort data=seas; by season station date; run;

/****************************************************************************************************/
/* SECTION 6C. set up an array to hold the index values so that                                     */
/* calculations can be done on them. the following code                                             */
/* calculates the K statistic. If there are more than 400 values                                    */
/* for a station, the array val and the array yrs will need to                                      */
/* be expanded, as explained in section 5B.                                                         */
/****************************************************************************************************/

data calc sen_s (keep= station season q count); set seas; by season station;

 if first.station then do;

 place= _n_ - 1;

 array val (400) x1-x400;
 do i = 1 to 400;
  val (i)= .;
 end;

 array yrs (400) y1-y400;
 do i= 1 to 400;
  yrs (i) =.;
 end;

   do i= 1 to n;
    dex=i+place;
    set seas point= dex;
    val (i)=value;
    yrs (i)=yr;
   end;
```

```
count=0;
sum=0;
 do c=2 to n;
  do i=1 to c-1;
   diff= val(c) - val(i);
   yrdiff= yrs (c) - yrs (i);
   if yrdiff=0 then sign=0;
   else do;
    if diff<0 then sign= -1;
    if diff=0 then sign= 0;
    if diff>0 then sign= 1;
    q= diff / yrdiff;
    count=count+1;
    output sen_s;
   end;
   sum=sum+sign;
  end;
 end;

end;
retain place;
output calc;
run;

/* determine the median slope value */

proc sort data=sen_s; by station; run;

/* rank the sen slopes so that confidence intervals can be calculated */

proc rank data=sen_s out=s_s_rank;
 by station;
 var q;
 ranks q1;
run;

proc sort data=s_s_rank; by station q1; run;

proc univariate noprint data= s_s_rank;
 by station;
 var q q1;
 output out=s_med_q median=median_q medcount max= max_q max_count;
run;

/* keep only the output records which have the sum data in them */

data calc; set calc (where=(sum^=.));
 keep station n sum season nyrs;
```

```
run;

/* sum up the sums across seasons */

proc sort data=calc; by station; run;

proc means data=calc noprint;
 var sum n;
 id nyrs;
 by station;
 output out=calcsum sum=sum n;
run;

data chekprob; merge sumvars calcsum;
 by station; keep station sumvar sum n nyrs;
 if sumvar<=0 then delete;
run;

/* calculate the probability of trend */

data chekprob; set chekprob;
  length qtrend $ 15;

  if sum>0 then do;
   zscore= (sum-1)/(sumvar**0.5);
  end;

  if sum = 0 then zscore=0;

  if sum<0 then do;
   zscore= (sum+1)/(sumvar**0.5);
  end;

  p= (1 - probnorm(abs(zscore)))*2;

/****************************************************************************************************/
/* SECTION 6D. modify the following line in order to alter the                                      */
/* two tailed probability. the original program is set at 0.10,                                     */
/* which can be changed as explained in section 5D.                                                 */
/****************************************************************************************************/

  if p< 0.10 then do;
   if sum>0 then qtrend='increasing';
   if sum<0 then qtrend='decreasing';
  end;
  else qtrend='insignificant';

  if p=. then qtrend='insufficient';
```

```
  calcmeth='seasonal';

run;

/*************************************************************************************************/
/* SECTION 7A. wrap up the sen slope calculations                                                */
/*************************************************************************************************/

data nons_var; set result; keep station var_ns; run;

data s_var; set sumvars; keep station sumvar;
 rename sumvar=var_ns; run;

data nons_var; set nons_var s_var; run;

data med_q; set med_q s_med_q; run;

proc sort data=med_q; by station; run;
proc sort data=nons_var; by station; run;

data c_lim ; merge med_q nons_var; by station; run;

/*************************************************************************************************/
/* SECTION 7B. In the following 2 lines, change the 1.645 to                                     */
/* something else to get other than 90% confidence limits on                                     */
/* the sen slope. Obtain the number from the normal                                              */
/* probability table. For example, if 80% confidence limits                                      */
/* are desired, use 1.2816.                                                                      */
/*************************************************************************************************/

data c_lim; set c_lim;
 lo_m= (max_count - (1.645*(var_ns**0.5)))/2;
 hi_m= (max_count + (1.645*(var_ns**0.5)))/2 +1;
run;

data sen_rank; set sen_rank s_s_rank; run;
proc sort data=sen_rank; by station; run;

data sen_calc; merge sen_rank c_lim; by station; run;

/* do an interpolation to get the value  */
/* for the confidence limits             */

data conf_lim; set sen_calc;

 if q1= int(lo_m) then do;
  lo=q;
  level=_n_+1;
  set sen_calc point=level;
```

```
 hi=q;
 num=int (lo_m);
 dec= lo_m - num;
 fract= dec*(hi-lo);
 lo_conf=lo+fract;
 if _error_ then abort;
end;

if q1= int(hi_m) then do;
 lo=q;
 level=_n_+1;
 set sen_calc point=level;
 hi=q;
 num=int (hi_m);
 dec= hi_m - num;
 fract= dec*(hi-lo);
 hi_conf=lo+fract;
 if _error_ then abort;
end;

run;

data conf_low (keep=station lo_conf) conf_hi (keep=station hi_conf);
 set conf_lim;
 if hi_conf^=. then output conf_hi;
 if lo_conf^=. then output conf_low;
run;

/*************************************************************************************************************/
/* SECTION 8. WRITE THE OUTPUT FILE.                                                                         */
/*************************************************************************************************************/

data kendalls; set result chekprob; run;

proc sort data=kendalls; by station; run;

data kendalls; merge kendalls med_q conf_low conf_hi; by station; run;

data kendalls; set kendalls;
 file data_out;
 if qtrend=' ' then qtrend='insufficient';
 if _n_=1 then do;
  put 'station,trend_n,trend_years,kendall_sum,zscore,zprob,
  trend,season,sen_slope,lo_confidence,hi_confidence';
 end;
 put station +(-1) ',' n +(-1) ',' nyrs +(-1) ','
  sum +(-1) ',' zscore 6.3 ',' p 6.3 ','
  qtrend +(-1) ',' calcmeth ',' median_q ',' lo_conf ',' hi_conf;
run;
```