A user guide to CORLAG and CORDUMP

software for antecedent and lag correlation analysis of WaVES hydrology outputs

Peter R. Briggs

Technical Report **29/97** (1997)

A User Guide to CORLAG and CORDUMP

Software for Antecedent and Lag Correlation Analysis of WaVES Hydrology Outputs

August 1995

Peter R. Briggs
Centre for Environmental Mechanics
CSIRO

Published as CSIRO Land and Water
Technical Report 29/97
1997

A User Guide to CORLAG and CORDUMP:

Software for Antecedent and Lag Correlation Analysis of WaVES Hydrology Outputs

August 1995

Peter R. Briggs
Centre for Environmental Mechanics
CSIRO

Contents

Introduction

This is a brief guide to the use of CORLAG and CORDUMP, two Fortran programs designed to aid the analysis of hydrology output from WaVES24, CSIRO Division of Water Resources' hydroecological model. Included is a sample analysis using the drainage, net rainfall, and storage columns from a 100-year WaVES hydrology file.

Familiarity with the WaVES24 hydrology file (*.hyd) format is assumed. While every effort has been made to eliminate the need for them, bug reports are gratefully received by *peter.briggs@cbr.clw.csiro.au*.

Files associated with this package are:

CORDOC.DOC	This documentation, a Word 6.0 file
CORLAG.FOR	A program to create tables of Pearson's r values from WaVES *.hyd time-series for various time-averagings or summings, antecedent accumulation periods, and lags between variables.

CORDUMP.FOR	A program to extract time-averaged or summed time-series for specific antecedent accumulation periods and lags based on examination of the CORLAG correlation tables.
CORLAG.PAR	The input parameter file used by CORLAG and CORDUMP, which should be edited by the user.
BENDIGO.XLS	An Excel 5.0 spreadsheet containing output generated by CORLAG and CORDUMP, and a number of associated graphs. The input to the analysis was a 100-year hydrology time-series at daily resolution produced by WaVES using a Bendigo parameterisation.

Antecedent Accumulation And Lag Correlation Analysis

The motivation behind CORLAG and CORDUMP was to explore a specific question about the hydrology of WaVES model output: How is the drainage over some time period simulated by WaVES related to the model's *accumulated antecedent* rainfall inputs? For example, how is this week's drainage related to the total rainfall over the past 2, 4, 8, etc. weeks of the model run? How might it be related to the total rainfall over an 2, 4, 8-week period starting 1 week ago, or 2 weeks ago? Is one week an appropriate time period over which to aggregate the WaVES daily outputs? Should it be monthly, or should it be left as a daily series?

To answer these questions we examine the correlation of hydrological time-series constructed with reference to three parameters:

Aggregation period (in days)

To filter out noise and reduce the occurrence of strings of zeroes in the daily time-series it may be appropriate to aggregate the WaVES .hyd output over longer time-periods. The time-series are summed over discrete aggregation periods. If the series represents a storage variable or parameter (such as leaf-area index) the mean over the aggregation period is taken. Further operations are done only on the aggregated time-series. In CORLAG and CORDUMP the aggregation period (in days) is assigned to the variable IAGGPER.

Antecedent accumulation period (in aggregation periods)

We expect the drainage occurring at some time to be dependent on how much rainfall has fallen in the recent past. The 'recent past' we define as an arbitrary number of aggregation periods going back in time from the current period. The creation of an accumulated antecedent rainfall series involves running a moving window of some length along the aggregated series, summing within the window (and averaging for storage variables). In CORLAG and CORDUMP the size of this window (the antecedent accumulation period) is assigned to the variable IACCUM.

Lag period (in aggregation periods)

If it takes some time for rainfall to become part of drainage flow, very recent rainfall events may not show an immediate association with current drainage. We can examine this possibility by lagging the antecent accumulation period by some number of aggregation periods, thus ignoring the most recent rainfall. In CORLAG and CORDUMP the lag period (in aggregation periods) is assigned to the variable LAG.

CORLAG: Correlation Of Aggregated, Accumulated, And Lagged Time-Series

For a chosen accumulation period and lag, we determine the degree of association between the aggregated drainage series and the aggregated, antecedent-accumulated and/or lagged rainfall series by calculating a Pearson's r correlation coefficient. In practice, we have no prior knowledge of the time constants involved, so we experiment with a range of aggregation periods, antecedent accumulation periods, and lags to produce tables of r-values. Our interest is in the combinations that produce the highest correlations.

The output of CORLAG is a 2-D table of Pearson's r-values for a range of antecedent accumulation periods (the rows) and lags (the columns). The range of accumulation periods and lags is set by the user in the CORLAG.PAR parameter file and assigned to the variables MINACCUM, MAXACCUM, MINLAG and MAXLAG in CORLAG. A single value for the aggregation period is also set in CORLAG.PAR. This can be altered in successive runs.

CORLAG allows the user to specify (in CORLAG.PAR) up to 10 pairs of time-series from the hydrology file (by column number) to produce correlation tables for. For example, we are also interested in the relationship between storage and the accumulated antecedent net rainfall.

A full description of the options is given in the comment section of CORLAG.PAR. A sample CORLAG.PAR file is included here as an appendix.

Below is a portion of the correlation table produced by CORLAG for various antecedent accumulation periods and lags of net rainfall against drainage from the 100-year Bendigo hydrology file. The aggregation period in this case was one week. The shaded cells show peak correlations occurring at lag-1 accumulation periods of 8, 9, and 10 weeks, and a lag-0 accumulation period of 10 weeks.

		LAG 0	LAG 1	LAG 2	LAG 3	LAG 4	LAG 5
	1	0.141	0.246	0.34	0.347	0.32	0.287
	2	0.266	0.402	0.472	0.458	0.417	0.368
A	3	0.398	0.51	0.551	0.522	0.468	0.412
C	4	0.496	0.579	0.597	0.555	0.496	0.435
C	5	0.564	0.623	0.624	0.574	0.51	0.446
U	6	0.609	0.648	0.638	0.583	0.516	0.451
M	7	0.636	0.662	0.643	0.584	0.516	0.451
U	8	0.652	0.667	0.641	0.581	0.513	0.449
L	9	0.661	0.667	0.637	0.576	0.509	0.446
A	10	0.664	0.664	0.631	0.57	0.504	0.439
T	11	0.662	0.657	0.624	0.563	0.495	0.428
I	12	0.657	0.65	0.615	0.552	0.482	0.415
O	13	0.652	0.643	0.604	0.539	0.469	0.403
N	14	0.647	0.633	0.59	0.525	0.456	0.391
	15	0.638	0.62	0.576	0.512	0.443	0.375
P	16	0.626	0.606	0.562	0.497	0.426	0.355
E	17	0.613	0.592	0.547	0.48	0.406	0.333
R	18	0.6	0.577	0.529	0.459	0.383	0.309
I	19	0.586	0.559	0.508	0.436	0.358	0.281
O	20	0.569	0.539	0.485	0.411	0.331	0.252
D	21	0.55	0.516	0.46	0.383	0.301	0.221

CORDUMP: Examination Of Interesting Aggregated, Accumulated, And Lagged Time Series

We may be interested to examine the aggregated time-series of drainage and accumulated/lagged net rainfall that produce such high correlations. These time-series can be recreated by running CORDUMP, which uses the information in CORLAG.PAR to generate a file of the aggregated and accumulated/lagged series. The particular accumulation periods and lags of interest (e.g. 10 and 0; 10 and 1; 9 and 1; 8 and 1) are specified interactively by the user at run-time. The output of CORDUMP is a file containing a 'time' column (1 to the number of aggregation periods), an aggregated series for every .hyd column mentioned in the parameter file, and an accumulated and lagged series for every accumulation/lag combination specified by the user at run-time, for every .hyd column given in the parameter file as the second of a pair (i.e. to be lagged and accumulated). An example of this type of file is given below. Note that due to the combination of lagging and incomplete accumulation, the shaded cells at the beginning and end of the file should not be plotted. The unused cells at the end are set to a missing value as a reminder. The same could be done at the beginning, but it is comforting to be able to check the progress of the accumulation. Remember to ignore the first `Accum-Lag-1` cells at the beginning of each column.

Time Periods (Days=7)	Drainage Accum=1 Lag=0	Net R Accum=1 Lag=0	Storage Accum=1 Lag=0	Net R Accum=10 Lag=0	Net R Accum=9 Lag=1	Net R Accum=8 Lag=1
1	0.3672	0	299.7471	0	0	0
2	0.098	0	282.0043	0	15.68	15.68
3	0.0043	15.68	285.9814	15.68	15.68	15.68
4	0.0004	0	280.6	15.68	24.26	24.26
5	0	8.58	282.7571	24.26	42.82	42.82
6	0	18.56	284.54	42.82	42.82	42.82
7	0	0	284.9971	42.82	43.1	43.1
8	0	0.28	280.5743	43.1	43.1	43.1
9	0	0	280.5386	43.1	76.46	76.46
10	0	33.36	293.18	76.46	76.46	60.78
...	...	...	...	...	...	...
5210	0.314	16.86	307.6957	98.1398	78.6399	61.2799
5211	0.1293	0	284.49	93.1798	67.8399	65.6799
5212	0.0094	6.56	282.4043	85.1998	74.2599	68.6799
5213	0.0009	8.58	284.8614	76.4198	101.2199	96.3399
5214	0.0001	32.54	294.5085	106.7998	96.3399	70.5999
5215	0.0005	0	290.88	101.2198	70.5999	70.5999
5216	0.0002	0	280.6043	96.3398	95.7599	89.6999
5217	0	25.16	290.51	95.7598	89.6999	72.8399
5218	0	0	281.9343	95.7598	-999.9999	-999.9999

Sample Plots From The Output Products Of CORLAG And CORDUMP

The output products of both CORLAG and CORDUMP are formatted for easy importing to spreadsheets, where they can be used to create plots along the lines of those given in BENDIGO.XLS, and discussed briefly below. The correlation tables and aggregated series files can be imported to Excel 5.0 as *delimited text files* with *space delimiters*, *double-quote text qualifiers*, and the *'treat consecutive delimiters as one'* flag turned on.

Figure 1: Weekly Drainage and Rainfall

This is a plot of the aggregated time series for drainage and net rainfall. It shows the difficulty in identifying the relationship between these two variables, even at weekly resolution, without antecedent accumulation of the net rainfall.

Figure 2: Weekly Drainage vs. Weekly Net Rainfall

This plot shows further the lack of a strong relationship between drainage and unaccumulated net rainfall.

Figure 3: Correlation of Weekly Drainage with Weekly Antecedent Rainfall

This is a 'correlogram' showing the relationship between correlation-with-drainage, and the antecedent accumulation period of net rainfall for lags of zero to ten weeks. The eleven curves are simply the first eleven lag columns from the correlation table. The correlation peak for the lag-zero curve (coloured) occurs at 10 weeks, and at 8 and 9 weeks for the lag-1 curve.

Figures 4, 5: Weekly Drainage, 10-Week Antecedent Net Rainfall, and Weekly Storage

These plots (equivalent to Figure 1) show the first and last 25 years of the simulation, and the clear relationship between drainage, storage and the lag-zero 10-week antecedent net rainfall, identified as having a peak correlation in Figure 3.

Figure 6: Weekly Drainage vs. 10-Week Antecedent Net Rainfall

This plot (equivalent to figure 2) shows the much stronger relationship between drainage and antecedent net rainfall than between drainage and unaccumulated rainfall. There is still considerable scatter in the relationship, mainly due to the importance of storage levels on drainage flow.

Figure 7: Annual Drainage vs. Annual Net Rainfall

The time-series for this plot were produced by running CORDUMP with the value of IAGGPER set to 365 in CORLAG.PAR.

Step-By-Step Procedure For The Correlation Analysis

1. *Edit CORLAG.PAR* and follow the instructions therein. Choose values for the aggregation period (in days), and the minimum and maximum accumulation periods and lags to generate r-values for, and the pairs of columns in the .hyd file to do the analysis on.

2. *Run the CORLAG executable***.**

3. *Examine the correlation table produced by CORLAG*, and choose any interesting pairs of accumulation period and lag for further investigation.

4. *Run the CORDUMP executable* without modifying the CORLAG.PAR file. Specify the output file and accumulation/lag pairs when requested.

5. *Examine the output file from CORDUMP* and import it to a spreadsheet.

6. If time-series over other aggregation periods are desired (e.g. annual), *change the aggregation period in CORLAG.PAR* (e.g. to 365) and *re-run CORDUMP*.

Appendix A - Release Notes and Bug Reports

CORLAG and CORDUMP do a bit of general input error-checking but are hardly robust, and have not been checked extensively for their response to strange requests. Use them with care and with appropriate reference to this documentation.

If you happen to find any of the bugs, especially the serious ones, a quick note to *peter.briggs@cbr.clw.csiro.au* would be greatly appreciated. Bugs, bug fixes, and cautionary tales may be noted here:

CT: If you are getting an unexpected number of tables or columns of output from CORLAG or CORDUMP, check that the value of ICORPAIRS is set correctly in CORLAG.PAR. This is easy to forget when you make changes.

CT: In the lagged or accumulated column output of CORDUMP, ignore the first `Accum-Lag-1` rows of each column, which are not fully accumulated.

Appendix B - Sample CORLAG.PAR

```
'c:\waves\bendigo\bendigo.hyd'    * HYDFILE
'c:\waves\bendigo\bendigo.cor'    * CORFILE
7                                 * ISUMDAYS
1,104                             * MINACCUM, MAXACCUM
0,20                              * MINLAG, MAXLAG
1                                 * ICORPAIRS
12, 5                             * IXCOL(1), ILAGCOL(1)
##,##                             * IXCOL(2), ILAGCOL(2)
##,##                             * IXCOL(3), ILAGCOL(3)
##,##                             * IXCOL(4), ILAGCOL(4)
##,##                             * IXCOL(5), ILAGCOL(5)
##,##                             * IXCOL(6), ILAGCOL(6)
##,##                             * IXCOL(7), ILAGCOL(7)
##,##                             * IXCOL(8), ILAGCOL(8)
##,##                             * IXCOL(9), ILAGCOL(9)
##,##                             * IXCOL(10), ILAGCOL(10)
-----------------------------------------------------------------
CORLAG.PAR - COMMENTS TO RIGHT OF * AND FROM DASHED LINE DOWN

You are looking at CORLAG.PAR, input parameter file for CORLAG.FOR.
Make sure this file is in the same directory as the executable, or
add a pathname to the CORLAG.PAR open statement in CORLAG.FOR.

The subroutine GETPARS reads the contents of the first 6 lines
(before the *'s) in free format and assigns them to the variables
listed after the *'s.  Depending on the value of ICORPAIRS, GETPARS
will look for up to 10 pairs of numbers to assign to elements of
IXCOL and ILAGCOL.  Replace the # signs with .HYD column numbers
(see below) as required by the value of ICORPAIRS.

VARIABLES:

HYDFILE:   Full pathname for hydrology file, delimited by single quotes
CORFILE:   Full pathname for output correlation table file, delimited
           by single quotes.
ISUMDAYS:  Take the incoming time series and average or sum it into
           block means or totals over ISUMDAYS time steps.  Fluxes are
           summed, storage terms (Columns 6,14,15,16) are averaged.
MINACCUM, MAXACCUM:
           The columns specified as elements of ILAGCOL will be
           be accumulated over MINACCUM to MAXACCUM periods, forming
           the rows of the output correlation table.  Suggested values
           are 1 and 104. (104 = two years when ISUMDAYS is 7).
MINLAG, MAXLAG
           Lag the accumulated ILAGCOL column means or totals behind
           corresponding IXCOL column means or totals by MINLAG to
           MAXLAG time periods (each time period is ISUMDAYS long).
           These form the columns of the output correlation table.
           Suggested values are 0 and 20.
ICORPAIRS: Number of pairs of columns from HYDFILE to do correlation
           analysis on.  The same number of IXCOL, ILAGCOL pairs of
           numbers should follow.
IXCOL(1..10), ILAGCOL(1..10)
           Corresponding elements are the pairs of column numbers from
           the HYDFILE to do the correlation analysis on.  Columns
           given in ILAGCOL will have antecedent accumulation performed
           (from MINACCUM to MAXACCUM) and be lagged (from MINLAG to
           MAXLAG) with respect to the column given in the corresponding
           element of IXCOL.  The 10-pair maximum is arbitrary.  Replace
           the ##,## with column number pairs as needed depending on
           ICORPAIRS.  Columns in the HYDFILE are as follows:

           Column   Variable

           1        Time (days)
           2        Gross Rainfall
           3        Overstorey Interception
           4        Understorey Interception
           5        Net Rainfall
           6        Storage
           7        Soil Evaporation
           8        Overstorey Transpiration
           9        Understorey Transpiration
           10       Total Evapotranspiration
           11       Total Lateral Flux
           12       Drainage
           13       Overland Flow
           14       Overstorey LAI
           15       Understorey LAI
           16       Litter Pool (kg)
```

Appendix C: Sample Sessions With CORLAG And CORDUMP

CORLAG (Non-Interactive)

```
CORLAG:  Fetching parameters...
CORLAG:  Reading .HYD file c:\waves\bendigo\bendigo.hyd...
CORLAG:  Accumulating and correlating...
CORLAG:  Normal Finish
```

CORDUMP (Interactive)

```
CORDUMP:  Fetching parameters from CORLAG.PAR...
CORDUMP:  Reading and block-averaging .HYD file  c:\waves\bendigo\bendigo.hyd...

CORDUMP:  Enter accumulation period (1-500): 10
CORDUMP:                  Enter lag (0-250): 0

CORDUMP:                      Another (y/n): y

CORDUMP:  Enter accumulation period (1-500): 9
CORDUMP:                  Enter lag (0-250): 1

CORDUMP:                      Another (y/n): Y

CORDUMP:  Enter accumulation period (1-500): 8
CORDUMP:                  Enter lag (0-250): 2

CORDUMP:                      Another (y/n): n
CORDUMP:             Write results to file : c:\waves\bendigo.ser

CORDUMP:  Normal Finish
```

```
      PROGRAM CORLAG
C*********************************************************************
C
C Performs correlation analysis of hydrology output from DWR Waves
C model by block-aggregation or averaging, antecedent accumulation, and
C lagging of the original time series in the Waves .HYD file.
C
C Created July 95 by Peter Briggs
C CSIRO Centre for Environmental Mechanics
C
C Bug reports gratefully received: peter.briggs@cbr.clw.csiro.au
C
C*********************************************************************
C
      IMPLICIT LOGICAL (A-Z)  ! Force explicit typing of variables
C
C VARIABLES AND CONSTANTS (* User-assigned in CORLAG.PAR)
C
C * HYDFILE - Name (including path) of the .HYD file.
C * CORFILE - Name (including path) of the output correlation file.
C   K,J,IACCUM - Loop counters
C   N - Sum counter, eventually the length of correlation series <=
C     NUMAGGS depending on LAG and IACCUM.  See R below for N < 30.
C   LAG - Number of aggregation periods to lag Y behind X series.
C * IAGGPER - Sum or average the incoming hydrology time series over
C     this many timesteps to create NUMAGGS aggregated totals or means.
C   NUMAGGS - Number of aggregated totals/means after summing over IAGGPER.
C     All other operations will be done on this many values.
C   MAXAGG - Maximum NUMAGGSs allowed (13036), which allows a 250-year
C     daily time series with IAGGPER set to 7.
C * MINACCUM, MAXACCUM - Min and max antecedent accumulation periods
C     for which to generate correlations between X and accumulated antecedent Y.
C     Unless the dimensions of RVALS are reset, MINACCUM must be >=1 and
C     MAXACCUM must be <=500.
C * MINLAG,MAXLAG - Min and max lags for which to generate
C     correlations between X and lagged accumulated antecedent Y.  Unless
C     the dimensions of RVALS are reset, MINLAG must be >= 0 and MAXLAG
C     must be <=500.
C * ICORPAIRS - Number of pairs of columns in .HYD file to do analysis on.
C * IXCOL(10),ILAGCOL(10) - Corresponding elements are pairs of column numbers
C     in HYDFILE on which to do the analysis.  The ILAGCOL columns will be
C     accumulated and lagged.  The choice of 10 is arbitrary.
C   COLSTORE(16,MAXAGG) - Stores the NUMAGGS number of aggregated means/totals
C     collected from the .HYD columns specified in IXCOL.  If IXCOL(K)=n, The
C     aggregated means/totals for column n will be in COLSTORE(n,1..NUMAGGS).
C   CUMSTORE(16,MAXAGG) - Stores the NUMAGGS number of aggregated means/totals
C     collected from the .HYD columns specified in ILAGCOL with antecedent
C     accumulation over IACCUM periods.  If ILAGCOL(K)=n, The
C     accumulated values for column n will be in CUMSTORE(n,1..NUMAGGS).
C   COLHEADS(16) - Stores the text headings for the 16 columns in HYDFILE.
C   R - Single value of Pearson's r.  If an attempt to calculate R would
C     result in a division by 0 (e.g. a time series of zeroes) a warning
C     is printed and R is set to 88.  If N is less than 30, R is set to 99.
C   RVALS(500,0:250) - Output table of r-values including each combination
C     of antecedent accumulation period (1..500) and lag (0..250).  See
C     above for error values 88 & 99.
C   X(MAXAGG),Y(MAXAGG) - Carriage variables for whatever straight,
C     accumulated or soon-to-be-lagged series are to be passed to
C     the correlation subroutine.
C   LSTNBL(C) - Function that returns the position in the string C
C     of the last non-blank. For trimming blanks.
C
      INTEGER IAGGPER, MINACCUM, MAXACCUM, MINLAG, MAXLAG, ICORPAIRS,
     &        NUMAGGS, IACCUM, LAG, MAXAGG, K, N, J
      INTEGER IXCOL(10), ILAGCOL(10)
      INTEGER LSTNBL
```

```
      CHARACTER*80 HYDFILE,CORFILE
      CHARACTER*9  COLHEADS(16)
      PARAMETER (MAXAGG = 13036)
      REAL COLSTORE(16,MAXAGG), CUMSTORE(16,MAXAGG),
     &     R, RVALS(500,0:250), X(MAXAGG), Y(MAXAGG)
C
C Get the user-assigned parameters for the correlation table, and
C open the corelation output file.
C
      PRINT *,'CORLAG:  Fetching parameters...'
      CALL GETPARS (HYDFILE,CORFILE,IAGGPER,MINACCUM,MAXACCUM,
     &              MINLAG,MAXLAG,ICORPAIRS,IXCOL,ILAGCOL)
      OPEN (22,FILE=CORFILE,STATUS='UNKNOWN')
C
C Read the 16 raw time series from the .HYD file, totalling or
C averaging over periods of IAGGPER length and returning the
C resulting series in COLSTORE.
C
      PRINT *,'CORLAG:  Reading .HYD file ',
     &         HYDFILE(:LSTNBL(HYDFILE)),'...'
      CALL READSUM (HYDFILE,NUMAGGS,MAXAGG,IAGGPER,
     &              COLHEADS,COLSTORE)
C
C For each of the pairs of columns given in the parameter file...
C
      PRINT *,'CORLAG:  Accumulating and correlating...'
      DO 150, K=1,ICORPAIRS
C
C For each in the range of accumulation periods requested...
C
        DO 130, IACCUM = MINACCUM,MAXACCUM
C
C Create the accumulated antecedent time series' for the ILAGCOL
C columns in COLSTORE, returning the accumulated time-series in
C the corresponding column of CUMSTORE.
C
          CALL ACCUM (ICORPAIRS,ILAGCOL,IACCUM,NUMAGGS,
     &                COLSTORE,CUMSTORE)
C
C Starting with the first fully-accumulated element of the
C accumulated time-series, copy the corresponding IXCOL and
C ILAGCOL-defined time-series into carriage variables X and Y,
C counting the series length N as you go.
C
          N = 0
          DO 160, J=IACCUM,NUMAGGS
            N = N + 1
            X(N) = COLSTORE(IXCOL(K),J)
            Y(N) = CUMSTORE(ILAGCOL(K),J)
  160     CONTINUE
C
C For each in the range of lags requested...
C Send the X and Y series, their length N and the LAG off to be
C correlated.  The actual lagging of the Y series is done in LAGCORR.
C Put the returned R value in the appropriate element of the output
C table RVALS.
C
          DO 135, LAG = MINLAG,MAXLAG
            CALL LAGCORR (X,Y,N,LAG,R)
            RVALS(IACCUM,LAG) = R
  135     CONTINUE
C
  130   CONTINUE
C
C Write table headings and the contents of the RVALS correlation
C table for all accumulation periods and lags to the output file,
```

```
C
        WRITE (22,1010) IAGGPER,COLHEADS(IXCOL(K)),COLHEADS(ILAGCOL(K))
 1010   FORMAT (' "PEARSON''S R FOR',I5,' DAY TOTALS OF ',A9,
     &          ' VS. LAGGED & CUMULATIVE-ANTECEDENT ',A9,'"')
        WRITE (22,1015)
 1015   FORMAT (' "COLUMNS = LAGS , ROWS = PERIODS OF ACCUMULATION"')
        WRITE (22,1020) (LAG,LAG=MINLAG,MAXLAG)
 1020   FORMAT ('   *   ',251I7)
        DO 650, IACCUM = MINACCUM,MAXACCUM
          WRITE (22,1030) IACCUM, (RVALS(IACCUM,LAG),LAG=MINLAG,MAXLAG)
 1030     FORMAT (I5,251(F7.3))
  650   CONTINUE
  150 CONTINUE
C
C Finished processing and producing correlation tables for all
C user-specified pairs of columns.
C
      STOP 'CORLAG:  Normal Finish'
C
      END
C
C*********************************************************************
C
      SUBROUTINE GETPARS
     &  (HYDFILE,CORFILE,IAGGPER,MINACCUM,MAXACCUM,MINLAG,MAXLAG,
     &   ICORPAIRS,IXCOL,ILAGCOL)
C
C Get the program parameters from CORLAG.PAR
C
      IMPLICIT LOGICAL (A-Z)  ! Force explicit typing of variables
      CHARACTER*80 HYDFILE,CORFILE
      INTEGER IXCOL(10), ILAGCOL(10), I, IAGGPER,
     &        MINACCUM, MAXACCUM, MINLAG, MAXLAG, ICORPAIRS
C
      OPEN (2,FILE='CORLAG.PAR',STATUS='OLD')
      READ (2,*) HYDFILE
      READ (2,*) CORFILE
      READ (2,*) IAGGPER
      READ (2,*) MINACCUM,MAXACCUM
      READ (2,*) MINLAG,MAXLAG
      READ (2,*) ICORPAIRS
C
C Read ICORPAIRS of column numbers.
C
      DO 10,I=1,ICORPAIRS
        READ (2,*) IXCOL(I),ILAGCOL(I)
   10 CONTINUE
      CLOSE (2)
      RETURN
      END
C
C*********************************************************************
C
      SUBROUTINE READSUM (HYDFILE,NUMAGGS,MAXAGG,IAGGPER,
     &                    COLHEADS,COLSTORE)
C
C Reads the 16 raw time series from the .HYD file, totalling or
C averaging over periods of IAGGPER length and returning the
C resulting series in COLSTORE.  All columns are processed whether
C used or not (as the overhead is mostly in the reading).
C
C VARIABLES
C
C FIRSTCHAR - First character of the current HYDFILE input line.
C INDAYS, KK - Loop counters.
C COL(16) - Stores one 16-column line of data as read from the HYDFILE.
```

```
C COLSUM(16) - Stores the sum or average of COL over IAGGPER.
C
      IMPLICIT LOGICAL (A-Z)  ! Force explicit typing of variables
      CHARACTER*80 HYDFILE
      CHARACTER*9  COLHEADS(16)
      CHARACTER*1  FIRSTCHAR
      INTEGER NUMAGGS, IAGGPER, KK, K, INDAYS, J, MAXAGG
      INTEGER LSTNBL
      REAL    COL(16), COLSUM(16), COLSTORE(16,MAXAGG)

C
C Open the WAVES hydrology file with trailing blanks from its name.
C
      OPEN (3,FILE=HYDFILE(:LSTNBL(HYDFILE)),STATUS='OLD')
C
C Read header lines until the first occurrence of an initial '#'.
C Backspace the file and reread this line, which has the column headings.
C
      DO 33, KK=1,999
        READ (3,'(A1)') FIRSTCHAR
        IF (FIRSTCHAR.EQ.'#') THEN
          BACKSPACE (3)
          READ(3,'(16(1X,A9))')(COLHEADS(J),J=1,16)
          GOTO 331
        END IF
   33 CONTINUE
  331 CONTINUE
C
C Count the number of aggregated means/totals with NUMAGGS.
C For any number of input lines...
C
      NUMAGGS = 0
      DO 50,KK=1,999999
C
C If the time series in the HYDFILE is too long, stop and advise a fix.
C
        IF (((KK*IAGGPER)/IAGGPER).GT.MAXAGG) THEN
          STOP 'CORLAG ERR:  TOO MUCH DATA, ADJUST IAGGPER OR MAXAGG'
          END IF
C
C Initialise the summation variables.
C
        DO 55,K=1,16
          COLSUM(K)   = 0.0
          COLSUM(K)   = 0.0
   55   CONTINUE
C
C For 1 to the number of days to sum over...
C
        DO 60,INDAYS=1,IAGGPER
C
C Read the line and check it's not a comment trailer.
C If not, reread the line as 16 columns of data and add them to
C the sum variables for this period.
C
          READ (3,'(A1)',END=99,ERR=999) FIRSTCHAR
          IF (FIRSTCHAR.EQ.'#' .OR. FIRSTCHAR.EQ.'&') GOTO 99
          BACKSPACE (3)
          READ(3,*) (COL(J),J=1,16)
          DO 70,K=1,16
            COLSUM(K) = COLSUM(K) + COL(K)
   70     CONTINUE
   60   CONTINUE
C
C Increment the aggregation period counter.  For columns containing storage
C terms, take their mean.  Store the aggregated totals or means in the
```

```
C next elements of COLSTORE.
C
        NUMAGGS = NUMAGGS + 1
        DO 80,K=1,16
          IF (K.EQ.6 .OR. K.GE.14) THEN
            COLSUM(K) = COLSUM(K)/REAL(IAGGPER)
          END IF
          COLSTORE(K,NUMAGGS) = COLSUM(K)
   80   CONTINUE
   50 CONTINUE
   99 CONTINUE
      RETURN
C
  999 STOP 'CORLAG ERR: READ ERROR IN .HYD FILE'
      RETURN
      END
C
C*====================================================================*
C
      SUBROUTINE ACCUM (ICORPAIRS,ILAGCOL,IACCUM,NUMAGGS,
     &                  COLSTORE,CUMSTORE)
C
C Create the accumulated antecedent time series' for the ILAGCOL
C columns in COLSTORE, returning the accumulated time-series in
C the corresponding column of CUMSTORE.
C
C VARIABLES:
C
C DONE(16) - Logical variable keeping track of whether we have done
C            accumulation on this column already, in case an
C            ILAGCOL column is specified more than once.
C
      IMPLICIT LOGICAL (A-Z)  ! Force explicit typing of variables
      INTEGER NUMAGGS, ICORPAIRS, IACCUM, I, K, ILAGCOL(10)
      REAL COLSTORE(16,NUMAGGS), CUMSTORE(16,NUMAGGS)
      LOGICAL DONE(16)
C
C Initialize the column-done flags.
C
      DO 81,I=1,16
        DONE(I) = .FALSE.
   81 CONTINUE
C
C For each column-pair specified by the user...
C
      DO 84,K=1,ICORPAIRS
C
C If we haven't done so, initialise the column in CUMSTORE corresponding
C to the ILAGCOL column of this pair.
C
        IF (.NOT.DONE(ILAGCOL(K))) THEN
          DO 82,I=1,NUMAGGS
            CUMSTORE(ILAGCOL(K),I) = 0.0
   82     CONTINUE
          DONE(ILAGCOL(K)) = .TRUE.
        END IF
   84 CONTINUE
C
C Reinitialize the column-done flags.
C
      DO 86,I=1,16
        DONE(I) = .FALSE.
   86 CONTINUE
C
C For each column-pair specified by the user...
C
```

```
      DO 100, K=1,ICORPAIRS
C
C If we haven't already done this column...
C The cumulative total for the first row in the ILAGCOL column of
C CUMSTORE is just the value in the first row of COLSTORE. For each
C succeeding output row, the cumulative total is this row plus the
C cumulative total from the previous row.  Once we have accumulated more
C than IACCUM periods, subtract values from the cumulative total as the
C IACCUM window moves past them.
C
        IF (.NOT.DONE(ILAGCOL(K))) THEN
          CUMSTORE(ILAGCOL(K),1) = COLSTORE(ILAGCOL(K),1)
          DO 90,I=2,NUMAGGS
            CUMSTORE(ILAGCOL(K),I) = COLSTORE(ILAGCOL(K),I) +
     &                               CUMSTORE(ILAGCOL(K),I-1)
            IF (I.GT.IACCUM)
     &        CUMSTORE(ILAGCOL(K),I) = CUMSTORE(ILAGCOL(K),I) -
     &                                 COLSTORE(ILAGCOL(K),I-IACCUM)
   90     CONTINUE
C
C If it's a storage variable, take the mean over the number of values
C accumulated (which is <= the accumulation period for the first
C IACCUM values in the series).
C
          IF (ILAGCOL(K).EQ.6 .OR. ILAGCOL(K).GE.14) THEN
            DO 95, I=1,NUMAGGS
              CUMSTORE(ILAGCOL(K),I) =
     &           CUMSTORE(ILAGCOL(K),I) / REAL(MIN(I,IACCUM))
   95       CONTINUE
          END IF
        END IF
  100 CONTINUE
C
      RETURN
      END
C
C*====================================================================*
C
      SUBROUTINE LAGCORR (X,Y,N,LAG,R)
C
C Lag Y wrt X by LAG elements.  This is done by moving X forward by
C LAG elements rather than moving Y back.  Send the lagged series to
C a Pearson's r routine.
C
C VARIABLES:
C
C XLAGGED(NMAX) - Stores the forwardly lagged X-array when LAG>0.
C NL - The (smaller) size of N implied by a lag.
C NMIN, NMAX - Allowable limits on N.
C
      IMPLICIT LOGICAL (A-Z)  ! Force explicit typing of variables
      INTEGER N, NL, LAG, I, NMAX, NMIN
      PARAMETER (NMAX = 13036, NMIN = 30)
      REAL X(N), Y(N), XLAGGED(NMAX), R
C
C Check for N too small for statistical validity.  Set R=99 if so.
C
      IF (N.LT.NMIN) THEN
        R=99.
        RETURN
      END IF
C
C If no lag, just send the series to CORR as they are.  Otherwise,
C calculate the N implied by the lag (NL), check it for size, and
C if it's ok, copy the X series lagged positively to the XLAGGED
C array and send it with the Y series to CORR.
```

```
C
      IF (LAG.EQ.0) THEN
        CALL CORR(N,X,Y,R)
      ELSE
        NL = N-LAG
        IF (NL.LT.NMIN) THEN
          R = 99.
          RETURN
        END IF
        DO 210, I = 1,NL
          XLAGGED(I) = X(I+LAG)
  210   CONTINUE
        CALL CORR(NL,XLAGGED,Y,R)
      END IF
C
      RETURN
      END
C
C**********************************************************************
C
      subroutine corr(n,x,y,r)
      implicit logical (a-z)  ! Force explicit typing of variables
C
C  Numerical Recipes 'pearsn' without the significance testing.
C  See Press et al., 1992 (2nd ed).  Error check added as noted.
C
      integer n, j
      real r,x(n),y(n)
      real ax,ay,sxx,sxy,syy,xt,yt
C
      ax=0.
      ay=0.
      do 11, j=1,n
        ax=ax+x(j)
        ay=ay+y(j)
   11 continue
      ax=ax/n
      ay=ay/n
      sxx=0.
      syy=0.
      sxy=0.
      do 12, j=1,n
        xt=x(j)-ax
        yt=y(j)-ay
        sxx=sxx+(xt*xt)
        syy=syy+(yt*yt)
        sxy=sxy+(xt*yt)
   12 continue
C
C P.Briggs: Check for a potential zero-divide and flag it with r=88
C
      if (sxx*syy.eq.0.) then
        write (22,*) n,sxx,syy,sxy
        PRINT *, 'CORLAG WARN:  DIVIDE BY 0 FLAG IN CORR, SET R = 88.0'
        r=88.0
      else
        r=sxy/sqrt(sxx*syy)
      end if
      return
      end
C
C**********************************************************************
C
      FUNCTION LSTNBL(C)
C
C Returns position of the last non-blank in C, or zero if all blank.
```

```
C
      INTEGER LSTNBL
      CHARACTER*(*) C
C ---------------------------------
      DO 1 LSTNBL = LEN(C), 1, -1
    1 IF( C(LSTNBL:LSTNBL) .NE. ' ' ) RETURN
      LSTNBL      = 0
C
      END
```

```
      PROGRAM CORDUMP
C*********************************************************************
C
C Performs block-aggregation/averaging and antecedent accumulation of
C hydrology output from DWR Waves model.  Prints out the resulting
C time series' for a lag and antecedent accumulation period specified
C interactively by the user.
C
C Created July 95 by Peter Briggs
C CSIRO Centre for Environmental Mechanics
C
C*********************************************************************
C
      IMPLICIT LOGICAL (A-Z)  ! Force explicit typing of variables
C
C VARIABLES AND CONSTANTS (* User-assigned in CORLAG.PAR)
C                         (@ User-assigned interactively at runtime)
C
C * HYDFILE - Name (including path) of the .HYD file.
C @ DUMPFILE - Name (and optional path) of the aggregated and
C     accumulated time series output file.
C   I,J,K - Loop counters
C   DONE(16) - Logical variable keeping track of whether we have done
C     operations on this column already, in case a column is specified
C     more than once in ILAGCOL or IXCOL.
C @ LAG - Lag the accumulated antecedent time-series' this many
C         IAGGPER time periods.
C * IAGGPER - Sum or average the incoming hydrology time series over this
C     many timesteps to create NUMAGGS initial aggregated totals/means.
C   NUMAGGS - Number of aggregated totals/means after summing over IAGGPER.
C     All other operations will be done on this many values.
C   MAXAGG - Maximum NUMAGGSs allowed (13036), which allows a 250-year
C     daily time series with IAGGPER set to 7.
C * ICORPAIRS - Number of pairs of columns in .HYD file to do analysis on.
C * IXCOL(10),ILAGCOL(10) - Corresponding elements are pairs of column numbers
C     in HYDFILE on which to do the analysis.  The ILAGCOL columns will be
C     accumulated and lagged.  The maximum of 10 is arbitrary.
C   COLSTORE(16,MAXAGG) - Stores the NUMAGGS number of aggregated means/totals
C     collected from the .HYD columns specified in IXCOL.  If IXCOL(K)=n, The
C     aggregated means/totals for column n will be in COLSTORE(n,1..NUMAGGS).
C   CUMSTORE(16,MAXAGG) - Stores the NUMAGGS number of aggregated means/totals
C     collected from the .HYD columns specified in ILAGCOL with antecedent
C     accumulation over IACCUM aggregation periods.  If ILAGCOL(K)=n, The
C     accumulated values for column n will be in CUMSTORE(n,1..NUMAGGS).
C   COLHEADS(16) - Stores the text headings for the 16 columns in HYDFILE.
C @ YN - Stores the yes or no answer to CORDUMP questions to the user.
C   OUTSTORE(50,MAXAGG) - Stores up to 50 columns of aggregated, lagged, or
C     accumulated time series for output to the DUMPFILE.
C   OUTHEAD(50,3) - Stores up to 3 rows of header for each output column
C     time series for output to the DUMPFILE.  Except for the first column,
C     header rows are 1st=variable, 2nd=accumulation period, 3rd=lag.
C   IOUT - Counts the number of columns used in OUTSTORE and OUTHEAD,
C     incrementing every time a new output column is created.
C   LSTNBL(C) - Function that returns the position in the string C
C     of the last non-blank. For trimming blanks.
C   NXTNBL(C) - Function that returns the position in the string C
C     of the next non-blank. For trimming blanks.
C
      INTEGER IAGGPER, ICORPAIRS,
     &        NUMAGGS, IACCUM, LAG, MAXAGG, I, K, J, IOUT, NL
      INTEGER IXCOL(10), ILAGCOL(10)
      INTEGER LSTNBL, NXTNBL
      CHARACTER*80 HYDFILE,DUMPFILE
      CHARACTER*9  COLHEADS(16)
      CHARACTER*12 OUTHEAD(50,3), YN
      PARAMETER (MAXAGG = 13036)
```

```
      REAL COLSTORE(16,MAXAGG), CUMSTORE(16,MAXAGG)
      REAL OUTSTORE(50,MAXAGG)
      LOGICAL DONE(16)

C
C Get the necessary parameters from CORLAG.PAR.
C
      PRINT *,'CORDUMP:  Fetching parameters from CORLAG.PAR...'
      CALL GETPARS (HYDFILE,IAGGPER,ICORPAIRS,IXCOL,ILAGCOL)
C
C Read the 16 raw time series from the .HYD file, totalling or
C averaging over aggregation periods of IAGGPER length and returning the
C resulting series in COLSTORE.
C


      PRINT *,'CORDUMP:  Reading and block-averaging .HYD file ',
     &         HYDFILE(:LSTNBL(HYDFILE)),'...'
      CALL READSUM (HYDFILE,NUMAGGS,MAXAGG,IAGGPER,
     &              COLHEADS,COLSTORE)
C
C Create a 'time periods' header and column, containing just the numbers
C 1 to NUMAGGS.
C

      IOUT = 1
      OUTHEAD(IOUT,1) = '"Time"'
      OUTHEAD(IOUT,2) = '"Periods"'
      WRITE (OUTHEAD(IOUT,3),4040) IAGGPER
 4040 FORMAT ('"(Days=',I3,')"')
C
      DO 640, J=1,NUMAGGS
        OUTSTORE(IOUT,J) = J
  640 CONTINUE
C
C Initialize the column-done flags.
C
      DO 810,I=1,16
        DONE(I) = .FALSE.
  810 CONTINUE
C
C For each pair of column numbers given in IXCOL and ILAGCOL...
C
      DO 950, K=1,ICORPAIRS
C
C Copy the aggregated time series as-is to the
C output array, i.e. before any lagging and antecedent accumulation:
C
C If we haven't done this IXCOL column yet, increment the output
C column counter, create a heading by stripping leading and
C trailing blanks from the column title, and copy the series
C to the output array.
C
        IF (.NOT.DONE(IXCOL(K))) THEN
          IOUT = IOUT + 1
          OUTHEAD(IOUT,1) =
     &       '"'//COLHEADS(IXCOL(K))(NXTNBL(COLHEADS(IXCOL(K)))
     &                :LSTNBL(COLHEADS(IXCOL(K))))//'"'
          OUTHEAD(IOUT,2) = '"Accum=1"'
          OUTHEAD(IOUT,3) = '"Lag=0"'
          DO 660,I=1,NUMAGGS
            OUTSTORE(IOUT,I) = COLSTORE(IXCOL(K),I)
  660     CONTINUE
          DONE(IXCOL(K)) = .TRUE.
        END IF
C
```

```
C Do the same as above for each of the
C
          IOUT = IOUT + 1
     &       '"'//COLHEADS(ILAGCOL(K))(NXTNBL(COLHEADS(ILAGCOL(K)))
          OUTHEAD(IOUT,2) = '"Accum=1"'
          DO 670,I=1,NUMAGGS
  670     CONTINUE
        END IF
C
C in the correlation table produced by CORLAG.FOR.  Accumulated and
C this accumulation period and lag.
  244 CONTINUE       ! Return here for more
        WRITE (*,*)
 2340   FORMAT ($,' CORDUMP:  Enter accumulation period (1-500): ')
        WRITE (*,2350)
        READ (*,*) LAG
C
C
          DONE(I) = .FALSE.
        DO 150, K=1,ICORPAIRS
C Create the accumulated antecedent time series' for the ILAGCOL
C the corresponding column of CUMSTORE (as yet
C
     &                COLSTORE,CUMSTORE)
C Lag the CUMSTORE time series' as needed and write them to
C
C we're out of columns, print a warning but write the columns we've
C variable name (blanks stripped), the accumulation period, and the lag.
C to 'done'.
C If the lag is > 0 do the same, but copy the series to the output column
C

            IF (.NOT.DONE(ILAGCOL(K))) THEN
              IF (IOUT.GT.50) THEN
 2800           FORMAT (' CORDUMP:  EXCEEDED OUTPUT COLUMN SPACE... ',
                GOTO 344
```

```
              END IF
              OUTHEAD(IOUT,1) =
     &         '"'//COLHEADS(ILAGCOL(K))(NXTNBL(COLHEADS(ILAGCOL(K)))
     &                :LSTNBL(COLHEADS(ILAGCOL(K))))//'"'
              WRITE (OUTHEAD(IOUT,2),4035) IACCUM
 4035         FORMAT ('"Accum=',I3,'"')
              OUTHEAD(IOUT,3) = '"Lag=0"'
              DO 680,I=1,NUMAGGS
                OUTSTORE(IOUT,I) = CUMSTORE(ILAGCOL(K),I)
  680         CONTINUE
              DONE(ILAGCOL(K)) = .TRUE.
            END IF
          ELSE
            IF (.NOT.DONE(ILAGCOL(K))) THEN
              IOUT = IOUT + 1
              IF (IOUT.GT.50) THEN
                WRITE (*,2800)
                GOTO 344
              END IF
              OUTHEAD(IOUT,1) =
     &         '"'//COLHEADS(ILAGCOL(K))(NXTNBL(COLHEADS(ILAGCOL(K)))
     &                :LSTNBL(COLHEADS(ILAGCOL(K))))//'"'
              WRITE (OUTHEAD(IOUT,2),4035) IACCUM
              WRITE (OUTHEAD(IOUT,3),4060) LAG
 4060         FORMAT ('"Lag=',I3,'"')
              NL = NUMAGGS-LAG
              DO 690, I = 1,NL
                OUTSTORE(IOUT,I) = CUMSTORE(ILAGCOL(K),I+LAG)
  690         CONTINUE
C
C Fill up the leftover elements from the lagging with missing values.
C
              DO 691, I = NL+1,NUMAGGS
                OUTSTORE(IOUT,I) = -999.9999
  691         CONTINUE
              DONE(ILAGCOL(K)) = .TRUE.
            END IF
          END IF
C
  150   CONTINUE
C
C We've produced accumulated and lagged time series for all the ILAGCOL
C columns.  Ask if user wants to create series for another accumulation
C period/lag combination.  If yes go back to get more input.
C
        WRITE (*,*)
        WRITE (*,2360)
 2360   FORMAT
     &    ($,' CORDUMP:                       Another (y/n): ')
        READ (*,'(A12)') YN
C
C Strip all blanks from the input line and check for an affirmative answer.
C
        CALL STRIP(YN)
       IF (YN(12:).EQ.'Y' .OR. YN(12:).EQ.'y') GOTO 244
C
C Come here ready to choose an output file name if we've run out of
C output columns or the file name is bad.
C
  344 CONTINUE
C
C Ask for and open an output file.
C
       WRITE (*,2366)
 2366 FORMAT ($,' CORDUMP:             Write results to file : ')
       READ (*,'(A80)') DUMPFILE
```

```
      OPEN (22,FILE=DUMPFILE,STATUS='UNKNOWN',ERR=344)
C
C Write column headings to the output file, stripped of blanks or
C right-justified where necessary to make the columns line up.
C
      DO 755, I=1,3
        DO 497, J=1,IOUT
          IF (I.EQ.1) THEN
            CALL RJUST(OUTHEAD(J,I))
          ELSE
            CALL STRIP(OUTHEAD(J,I))
          END IF
  497   CONTINUE
        WRITE (22,9647) (OUTHEAD(J,I),J=1,IOUT)
 9647   FORMAT(20A12)
  755 CONTINUE
C
C Write the columns of the output array.
C

        WRITE (22,1030) NINT(OUTSTORE(1,I)),(OUTSTORE(J,I),J=2,IOUT)
 1030   FORMAT (1X,I10,19(1X,F11.4))

C
C All done.

      WRITE (*,*)
      STOP 'CORDUMP:  Normal Finish'

      END
C


C
      SUBROUTINE GETPARS

C
C Get the program parameters from CORLAG.PAR

C IDUMMY, CDUMMY - Integer and character dummy variables for unneeded
C   CORLAG.PAR parameters.

      IMPLICIT LOGICAL (A-Z)  ! Force explicit typing of variables
      CHARACTER*80 HYDFILE, CDUMMY

C
      OPEN (2,FILE='CORLAG.PAR',STATUS='OLD')

      READ (2,*) CDUMMY
      READ (2,*) IAGGPER

      READ (2,*) IDUMMY,IDUMMY
      READ (2,*) ICORPAIRS

C Read ICORPAIRS of column numbers.
C

        READ (2,*) IXCOL(I),ILAGCOL(I)
   10 CONTINUE

      RETURN
      END

C**********************************************************************
C

     &                     COLHEADS,COLSTORE)
```

```
C
C Reads the 16 raw time series from the .HYD file, totalling or
C averaging over accumulation periods of IAGGPER length and returning the
C resulting series in COLSTORE.  All columns are processed whether
C used or not (as the overhead is mostly in the reading).
C
C VARIABLES
C
C FIRSTCHAR - First character of the current HYDFILE input line.
C INDAYS, KK - Loop counters.
C COL(16) - Stores one 16-column line of data as read from the HYDFILE.
C COLSUM(16) - Stores the sum or average of COL over IAGGPER.
C
      IMPLICIT LOGICAL (A-Z)  ! Force explicit typing of variables
      CHARACTER*80 HYDFILE
      CHARACTER*9  COLHEADS(16)
      CHARACTER*1  FIRSTCHAR
      INTEGER NUMAGGS, IAGGPER, KK, K, INDAYS, J, MAXAGG
      INTEGER LSTNBL
      REAL    COL(16), COLSUM(16), COLSTORE(16,MAXAGG)

C
C Open the WAVES hydrology file with trailing blanks from its name.
C
      OPEN (3,FILE=HYDFILE(:LSTNBL(HYDFILE)),STATUS='OLD')
C
C Read header lines until the first occurrence of an initial '#'.
C Backspace the file and reread this line, which has the column headings.
C
      DO 33, KK=1,999
        READ (3,'(A1)') FIRSTCHAR
        IF (FIRSTCHAR.EQ.'#') THEN
          BACKSPACE (3)
          READ(3,'(16(1X,A9))')(COLHEADS(J),J=1,16)
          GOTO 331
        END IF
   33 CONTINUE
  331 CONTINUE
C
C Count the number of aggregated means/totals with NUMAGGS.
C For any number of input lines...
C
      NUMAGGS = 0
      DO 50,KK=1,999999
C
C If the time series in the HYDFILE is too long, stop and advise a fix.
C
        IF (((KK*IAGGPER)/IAGGPER).GT.MAXAGG) THEN
          STOP 'CORDUMP ERR:  TOO MUCH DATA, ADJUST IAGGPER OR MAXAGG'
          END IF
C
C Initialise the summation variables.
C
        DO 55,K=1,16
          COLSUM(K)   = 0.0
          COLSUM(K)   = 0.0
   55   CONTINUE
C
C For 1 to the number of days to sum over...
C
        DO 60,INDAYS=1,IAGGPER
C
C Read the line and check it's not a comment trailer.
C If not, reread the line as 16 columns of data and add them to
C the sum variables for this period.
C
```

```
          IF (FIRSTCHAR.EQ.'#' .OR. FIRSTCHAR.EQ.'&') GOTO 99
          BACKSPACE (3)

          DO 70,K=1,16
            COLSUM(K) = COLSUM(K) + COL(K)

   60   CONTINUE
C

C storage terms, take their mean.  Store the aggregated totals or
C means in the next elements of COLSTORE.

        NUMAGGS = NUMAGGS + 1
        DO 80,K=1,16

            COLSUM(K) = COLSUM(K)/REAL(IAGGPER)
          END IF

   80   CONTINUE
   50 CONTINUE

      RETURN
C

      RETURN
      END

C*********************************************************************
C

     &                  COLSTORE,CUMSTORE)
C

C columns in COLSTORE, returning the accumulated time-series in
C the corresponding column of CUMSTORE.

C VARIABLES:
C

      IMPLICIT LOGICAL (A-Z)  ! Force explicit typing of variables
      INTEGER NUMAGGS, ICORPAIRS, IACCUM, I, K, ILAGCOL(10)

      LOGICAL DONE(16)
C
  Initialize the column-done flags.
C

        DONE(I) = .FALSE.
   81 CONTINUE

C For each column-pair specified by the user...
C

C
C If we haven't done so, initialise the column in CUMSTORE corresponding

C
        IF (.NOT.DONE(ILAGCOL(K))) THEN

            CUMSTORE(ILAGCOL(K),I) = 0.0
   82     CONTINUE

        END IF
   84 CONTINUE
```

```
C Reinitialize the column-done flags.
C
      DO 86,I=1,16
        DONE(I) = .FALSE.
   86 CONTINUE
C
C For each column-pair specified by the user...
C
      DO 100, K=1,ICORPAIRS
C
C If we haven't already done this column...
C The cumulative total for the first row in the ILAGCOL column of
C CUMSTORE is just the value in the first row of COLSTORE. For each
C succeeding output row, the cumulative total is this row plus the
C cumulative total from the previous row.  Once we have accumulated more
C than IACCUM periods, subtract values from the cumulative total as the
C IACCUM window moves past them.
C
        IF (.NOT.DONE(ILAGCOL(K))) THEN
          CUMSTORE(ILAGCOL(K),1) = COLSTORE(ILAGCOL(K),1)
          DO 90,I=2,NUMAGGS
            CUMSTORE(ILAGCOL(K),I) = COLSTORE(ILAGCOL(K),I) +
     &                               CUMSTORE(ILAGCOL(K),I-1)
            IF (I.GT.IACCUM)
     &        CUMSTORE(ILAGCOL(K),I) = CUMSTORE(ILAGCOL(K),I) -
     &                                 COLSTORE(ILAGCOL(K),I-IACCUM)
   90     CONTINUE
C
C If it's a storage variable, take the mean over the number of values
C accumulated (which is <= the accumulation period for the first
C IACCUM values in the series).
C
          IF (ILAGCOL(K).EQ.6 .OR. ILAGCOL(K).GE.14) THEN
            DO 95, I=1,NUMAGGS
              CUMSTORE(ILAGCOL(K),I) =
     &           CUMSTORE(ILAGCOL(K),I) / REAL(MIN(I,IACCUM))
   95       CONTINUE
          END IF
        END IF
  100 CONTINUE
C
      RETURN
      END
C
C**********************************************************************
C
      SUBROUTINE STRIP(C)
C
C Strip blanks from and right justifies C.
C
      CHARACTER*(*) C
      LENGTH = LEN(C)
      J = LENGTH + 1
C
C Starting with the last character, if it's non-blank or non-null copy
C it to the end of the string.
C
      DO 10, I=LENGTH,1,-1
        IF (C(I:I).NE.' ' .AND. C(I:I).NE.CHAR(0)) THEN
          J = J - 1
          C(J:J) = C(I:I)
        END IF
   10 CONTINUE
C
C Fill up the front of the string with blanks now that we've copied all
C non-blank/null characters to the end.
```

```
C
      J = J - 1
      DO 20, I=J,1,-1
        IF (C(I:I).NE.' ' .AND. C(I:I).NE.CHAR(0)) C(I:I)=' '
   20 CONTINUE
      RETURN
      END
C
C*********************************************************************
C
      SUBROUTINE RJUST(C)
C
C Right justifies the first to last non-blank characters in C.
C
      CHARACTER*(*) C
C
      LENGTH = LEN(C)
      IBEGIN = NXTNBL(C)
      IEND   = LSTNBL(C)
      ITOEND = LENGTH - IEND
C
C Copy the substring bounded by non-blank characters to the end of C.
C
      DO 10, I = IEND,IBEGIN,-1
        IPOS = I + ITOEND
        C(IPOS:IPOS) = C(I:I)
   10 CONTINUE
C
C Fill up the front of the string with blanks if they're not blank or
C null already.
C
      IPOS = IPOS - 1
      DO 20, J = IPOS,1,-1
        IF (C(J:J).NE.' ' .AND. C(J:J).NE.CHAR(0)) C(J:J)=' '
   20 CONTINUE
      RETURN
      END
C
C*********************************************************************
C
      FUNCTION LSTNBL(C)
C
C Returns position of the last non-blank in C, or zero if all blank.
C
      INTEGER LSTNBL
      CHARACTER*(*) C
C ---------------------------------
      DO 1 LSTNBL = LEN(C), 1, -1
    1 IF( C(LSTNBL:LSTNBL) .NE. ' ' ) RETURN
      LSTNBL      = 0
C
      END

C
C*********************************************************************
C
      FUNCTION NXTNBL( C )
C
C Returns the position of the next non-blank in C, or zero if there
C are none.
C
      CHARACTER*(*) C
C ---------------------------------
      L = LEN( C )
      DO 1 I = 1, L
      IF( C(I:I) .NE. ' ' ) THEN
```

```
         NXTNBL = I
         RETURN
      END IF
    1 CONTINUE
C
      NXTNBL = 0
C
      END
```

```
'c:\waves\bendigo\bendigo.hyd'    * HYDFILE
'c:\waves\bendigo\bendigo.cor'    * CORFILE
7                                 * ISUMDAYS
1,104                             * MINACCUM, MAXACCUM
0,20                              * MINLAG, MAXLAG
2                                 * ICORPAIRS
12, 5                             * IXCOL(1), ILAGCOL(1)
 6, 5                             * IXCOL(2), ILAGCOL(2)
##,##                             * IXCOL(3), ILAGCOL(3)
##,##                             * IXCOL(4), ILAGCOL(4)
##,##                             * IXCOL(5), ILAGCOL(5)
##,##                             * IXCOL(6), ILAGCOL(6)
##,##                             * IXCOL(7), ILAGCOL(7)
##,##                             * IXCOL(8), ILAGCOL(8)
##,##                             * IXCOL(9), ILAGCOL(9)
##,##                             * IXCOL(10), ILAGCOL(10)
-----------------------------------------------------------------
CORLAG.PAR - COMMENTS TO RIGHT OF * AND FROM DASHED LINE DOWN

You are looking at CORLAG.PAR, input parameter file for CORLAG.FOR.
Make sure this file is in the same directory as the executable, or
add a pathname to the CORLAG.PAR open statement in CORLAG.FOR.

The subroutine GETPARS reads the contents of the first 6 lines
(before the *'s) in free format and assigns them to the variables
listed after the *'s.  Depending on the value of ICORPAIRS, GETPARS
will look for up to 10 pairs of numbers to assign to elements of
IXCOL and ILAGCOL.  Replace the # signs with .HYD column numbers
(see below) as required by the value of ICORPAIRS.

VARIABLES:

HYDFILE:   Full pathname for hydrology file, delimited by single quotes
CORFILE:   Full pathname for output correlation table file, delimited
           by single quotes.
ISUMDAYS:  Take the incoming time series and average or sum it into
           block means or totals over ISUMDAYS time steps.  Fluxes are
           summed, storage terms (Columns 6,14,15,16) are averaged.
MINACCUM, MAXACCUM:
           The columns specified as elements of ILAGCOL will be
           be accumulated over MINACCUM to MAXACCUM periods, forming
           the rows of the output correlation table.  Suggested values
           are 1 and 104. (104 = two years when ISUMDAYS is 7).
MINLAG, MAXLAG
           Lag the accumulated ILAGCOL column means or totals behind
           corresponding IXCOL column means or totals by MINLAG to
           MAXLAG time periods (each time period is ISUMDAYS long).
           These form the columns of the output correlation table.
           Suggested values are 0 and 20.
ICORPAIRS: Number of pairs of columns from HYDFILE to do correlation
           analysis on.  The same number of IXCOL, ILAGCOL pairs of
           numbers should follow.
IXCOL(1..10), ILAGCOL(1..10)
           Corresponding elements are the pairs of column numbers from
           the HYDFILE to do the correlation analysis on.  Columns
           given in ILAGCOL will have antecedent accumulation performed
           (from MINACCUM to MAXACCUM) and be lagged (from MINLAG to
           MAXLAG) with respect to the column given in the corresponding
           element of IXCOL.  The 10-pair maximum is arbitrary.  Replace
           the ##,## with column number pairs as needed depending on
           ICORPAIRS.  Columns in the HYDFILE are as follows:
```

Column	Variable
1	Time (days)
2	Gross Rainfall
3	Overstorey Interception
4	Understorey Interception
5	Net Rainfall
6	Storage
7	Soil Evaporation
8	Overstorey Transpiration
9	Understorey Transpiration
10	Total Evapotranspiration
11	Total Lateral Flux
12	Drainage
13	Overland Flow
14	Overstorey LAI
15	Understorey LAI
16	Litter Pool (kg)

Figure 1: Weekly Drainage and Rainfall

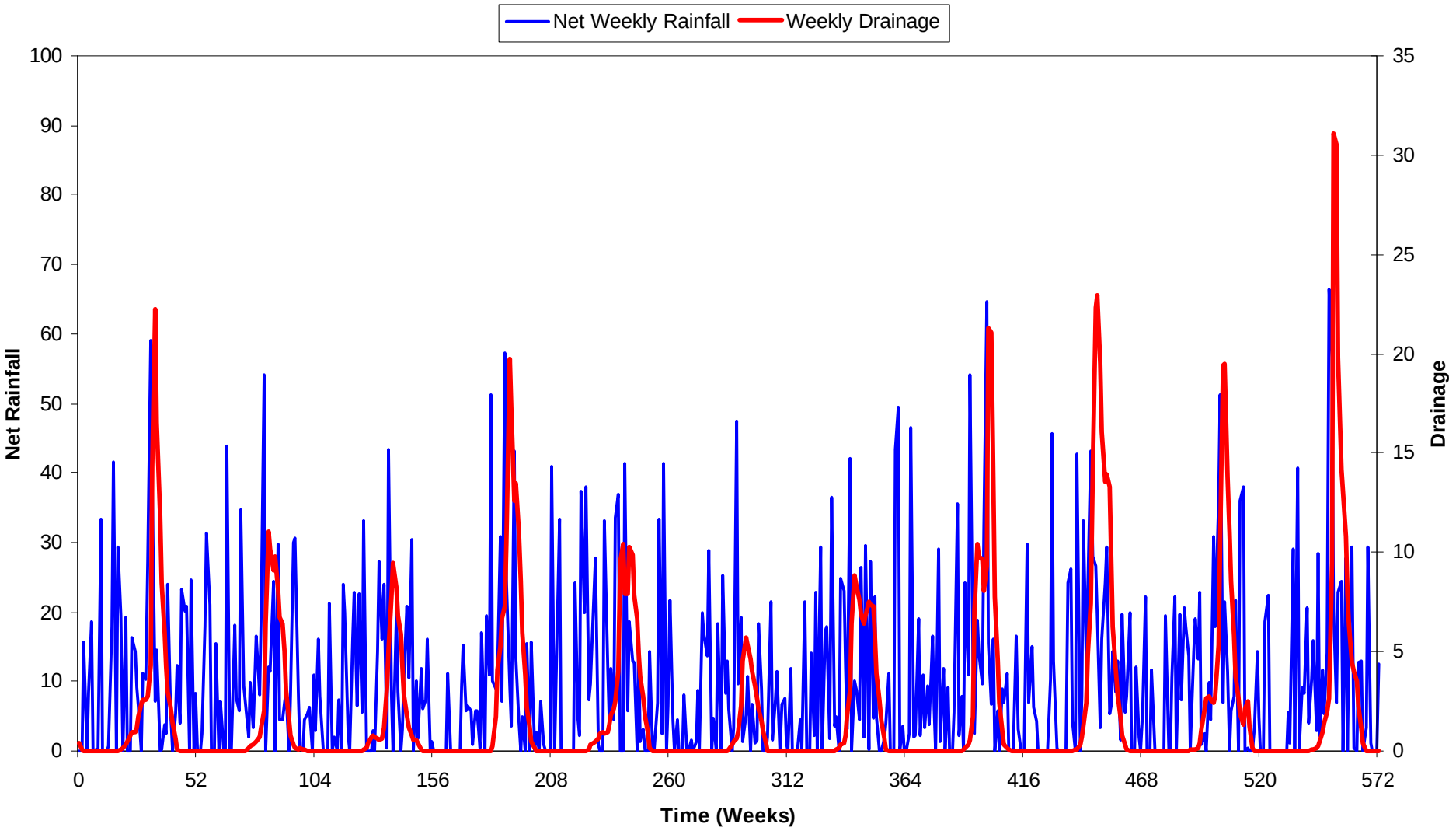

Figure 2: Bendigo Weekly Drainage vs. Weekly Net Rainfall
Last 4000 Weeks of 100-Year Series

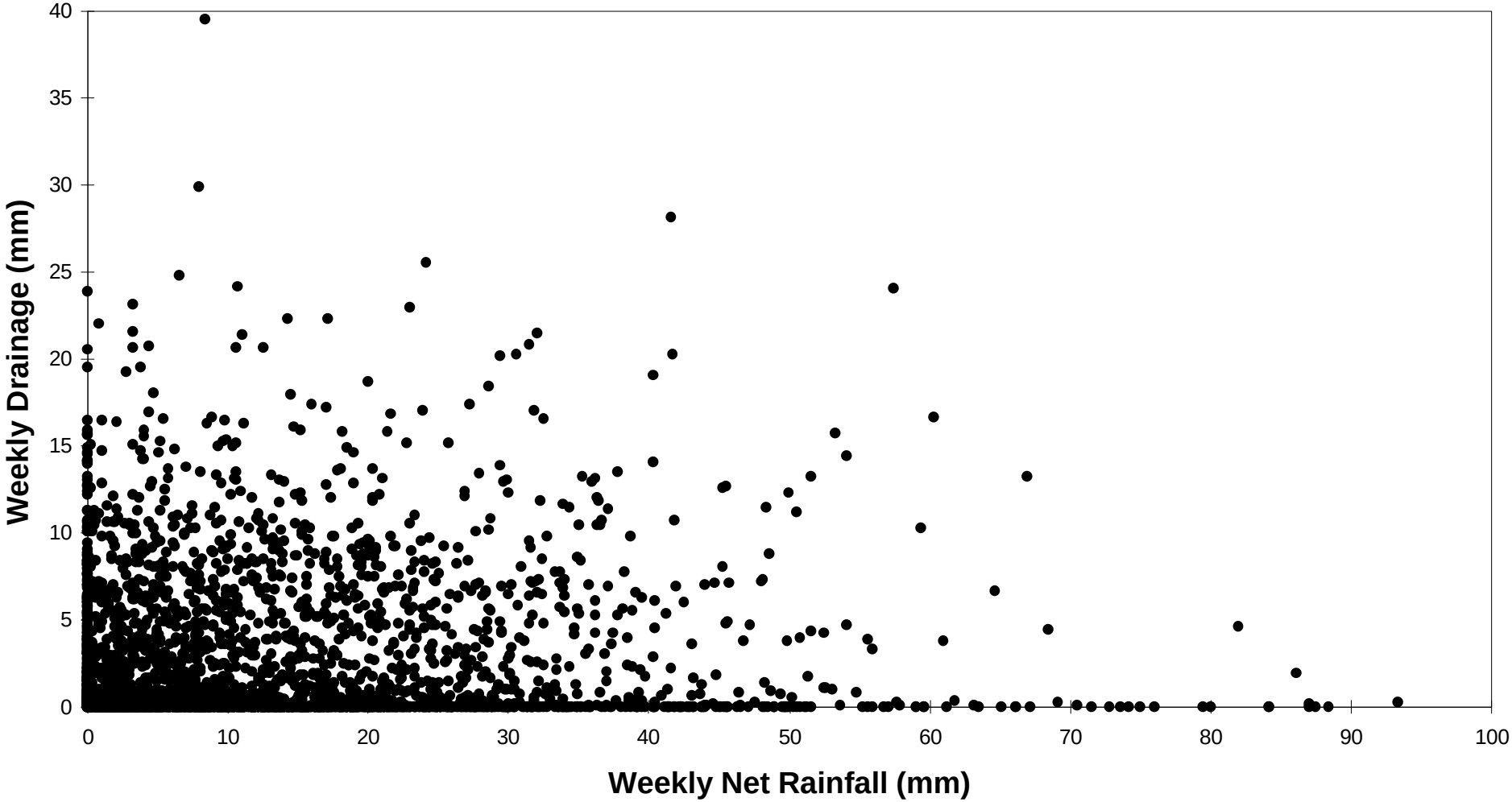

Figure 3: Correlation of Weekly Drainage With Weekly Antecedent Rainfall For Bendigo 100-Year Series

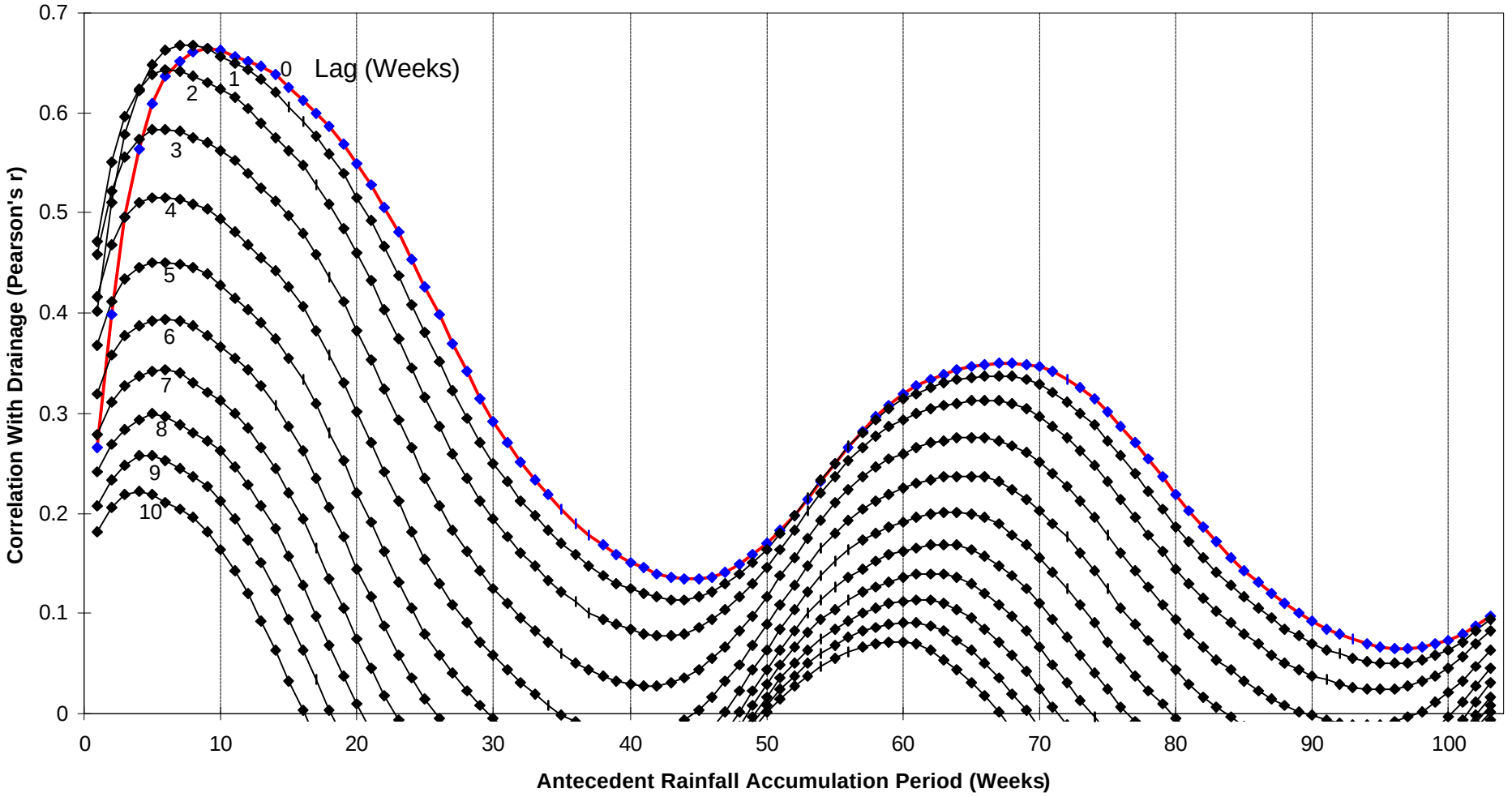

Figure 4: Bendigo Weekly Drainage, 10-Week Antecedent Net Rainfall, and Weekly Storage - Years 1 to 25

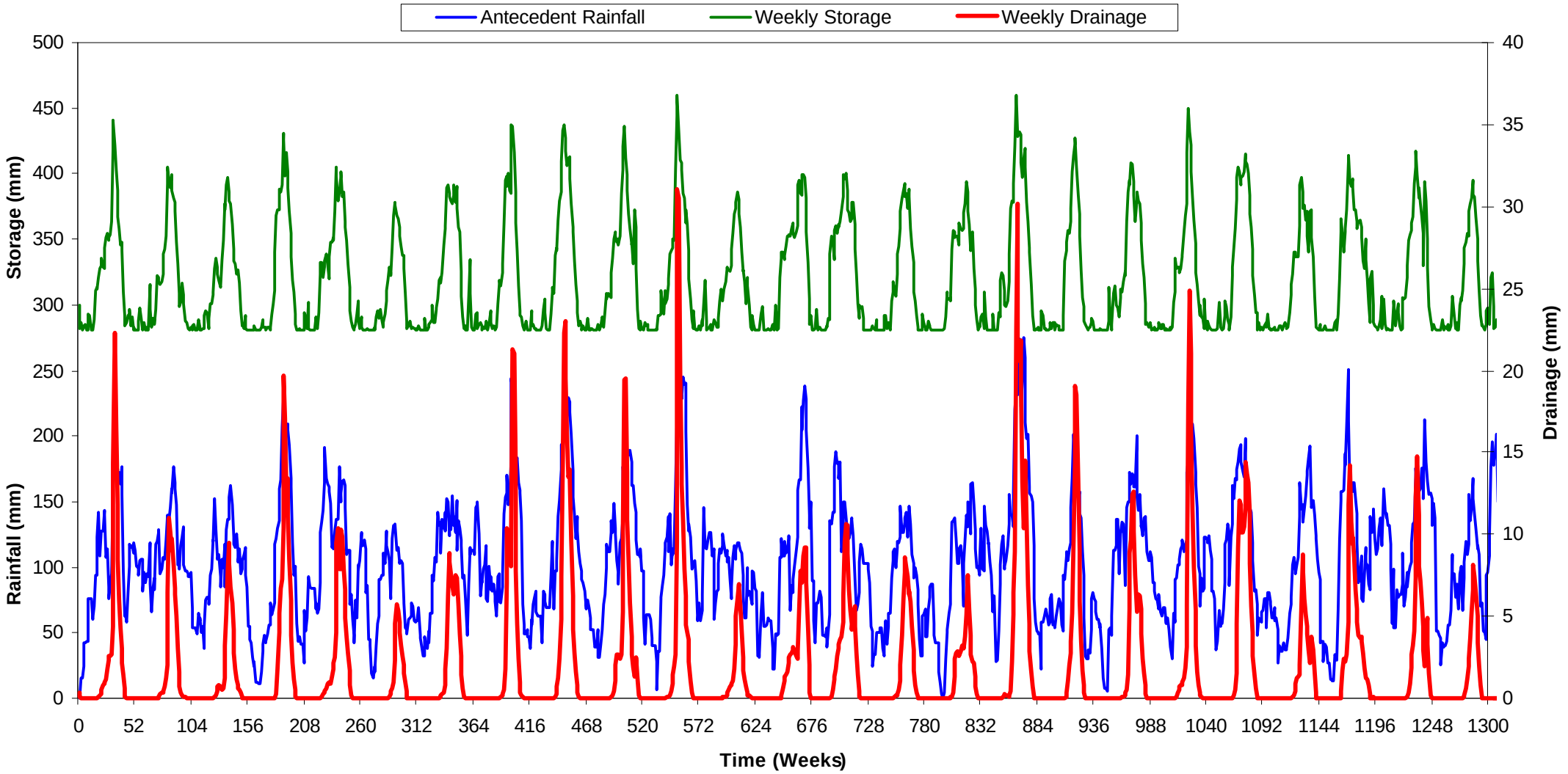

Figure 5: Bendigo Weekly Drainage, 10-Week Antecedent Net Rainfall, and Weekly Storage - Years 75 to 100

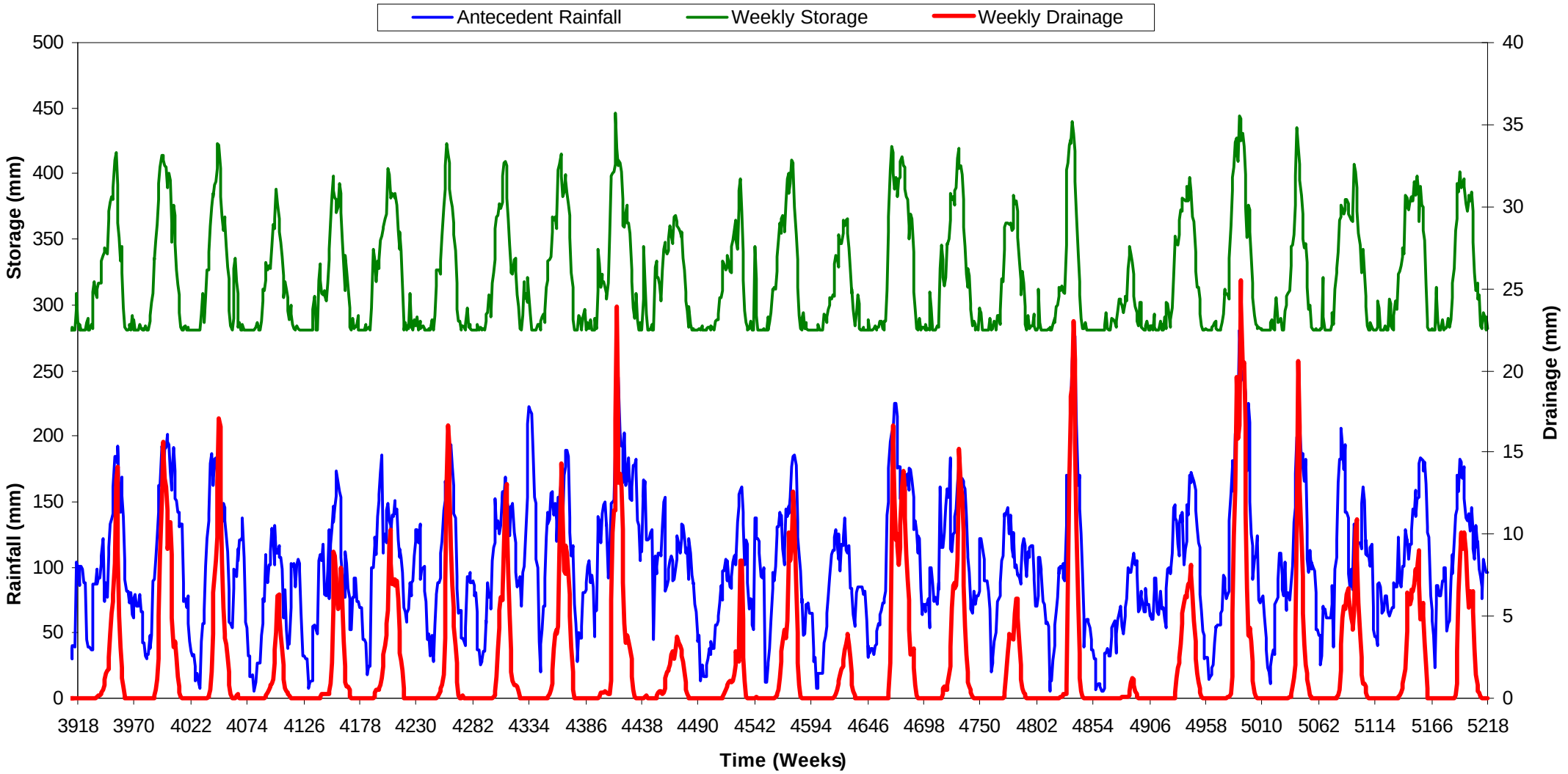

Figure 6: Bendigo Weekly Drainage vs. 10-Week Antecedent Net Rainfall For Last 4000 Weeks of 100-Year Series

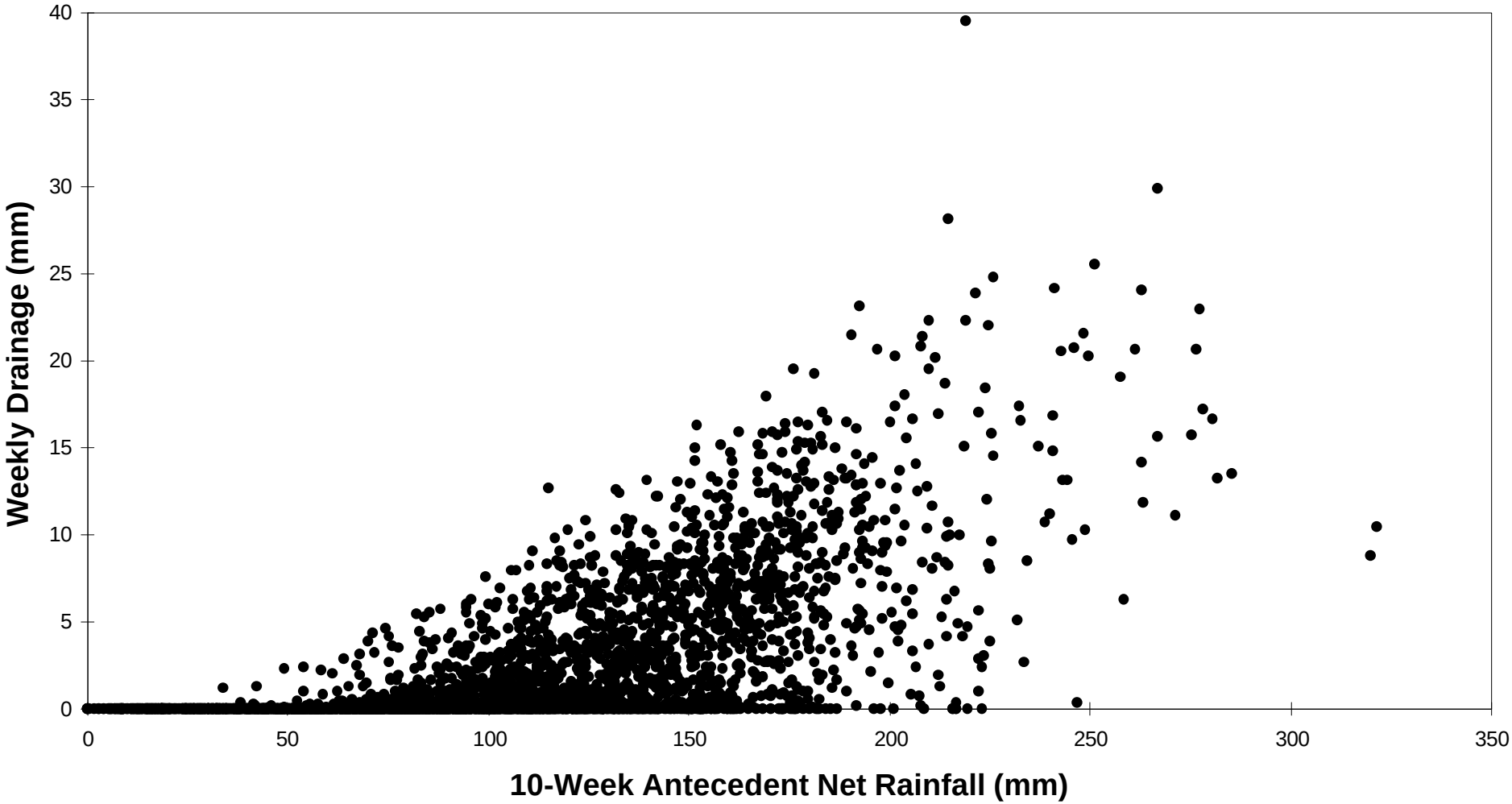

Figure 7: Annual Drainage vs. Annual Net Rainfall
Bendigo 100-Year Series