NOAA Technical Note NCDC No. USCRN-05-1

Calculation of USCRN Precipitation from Geonor Weighing Precipitation Gauge

C. Bruce Baker
NOAA/National Climatic Data Center
Asheville, NC

Rodney Buckner
NOAA/National Climatic Data Center
SOURCECORP, Incorporated
Asheville, NC

William Collins
Short and Associates, Inc.
NOAA/Office of Systems Development and
NOAA/National Climatic Data Center
Suitland, MD / Asheville, NC

Mark Phillips
NOAA/National Climatic Data Center
SOURCECORP, Incorporated
Asheville, NC

April 2005

U.S. DEPARTMENT OF COMMERCE
Donald L. Evans, Secretary

National Oceanic and Atmospheric Administration
Vice Admiral Conrad C. Lautenbacher, Jr., U.S. Navy (Ret.),
Under Secretary

National Climatic Data Center
Thomas R. Karl, Director

Calculation of USCRN Precipitation from Geonor Weighing Precipitation Gauge

Bruce Baker, NOAA/NCDC Asheville, NC

Rodney Buckner NOAA/NCDC SourceCorp, Asheville, NC

William Collins, NOAA OSD & NCDC, Suitland, MD / Asheville, NC

Mark Phillips NOAA/NCDC SourceCorp, Asheville, NC

April 2005

U.S. DEPARTMENT OF COMMERCE
National Oceanic and Atmospheric Administration
National Climatic Data Center
Asheville, NC 28801-5001

Table of Contents

1. Introduction

The Geonor precipitation gauges of the USCRN network are fitted with three wires for the measurement of liquid depth. Ideally, the measurements from the individual wires should be identical, with identical changes during precipitation events. However, this is not the case and the redundancy of measurement is a powerful tool for making an accurate estimate of the true precipitation. This note discusses an algorithm that was developed to give as accurate an estimate of the precipitation as possible, even in the presence of "noise" in the wire measurements and other difficulties to be discussed below.

2. Nature of the Measurements

The USCRN network is an evolving network, with the quality of the precipitation measurements improving in time as problems of various kinds are worked out. In the past, particularly, there have been stations with gauge depth measurements that were noisy and with unusual changes. Figures 1 and 2 show examples of depths from ill-behaved stations for October 1, 2003 through September 30, 2004. The depths from Wolf Point MT 34 NE are particularly suspect and present a challenge even for the algorithm proposed in this note. The depth values from Stillwater OK 2 W, which have much noise, are handled well by the algorithm. The Stillwater depths clearly show two times when the gauges were emptied. Significant increases occur when antifreeze and oil are added. Also, it is possible for wires to break. All of these occurrences must be handled by the precipitation algorithm. With that in mind, the next section will discuss the objectives which led the development of the algorithm.

3. Objectives for the Algorithm Development

The objectives for the algorithm development are listed below, not necessarily in order of importance. Naturally, the overall objective is to produce a precipitation value that is as accurate as possible using the three wire depth measurements, for conditions ranging from light precipitation to heavy downpours, while eliminating false precipitation indications from "noise" and other occurrences.

1) Allow inclusion of light precipitation, both between more major precipitation events and at the tail end of a precipitation event. This objective led to a method that uses a time-average of past depth values, wire-by-wire, as a reference level for precipitation calculations. The method also continues for a period of time to recognize the depth measurements at the end of each precipitation amount.

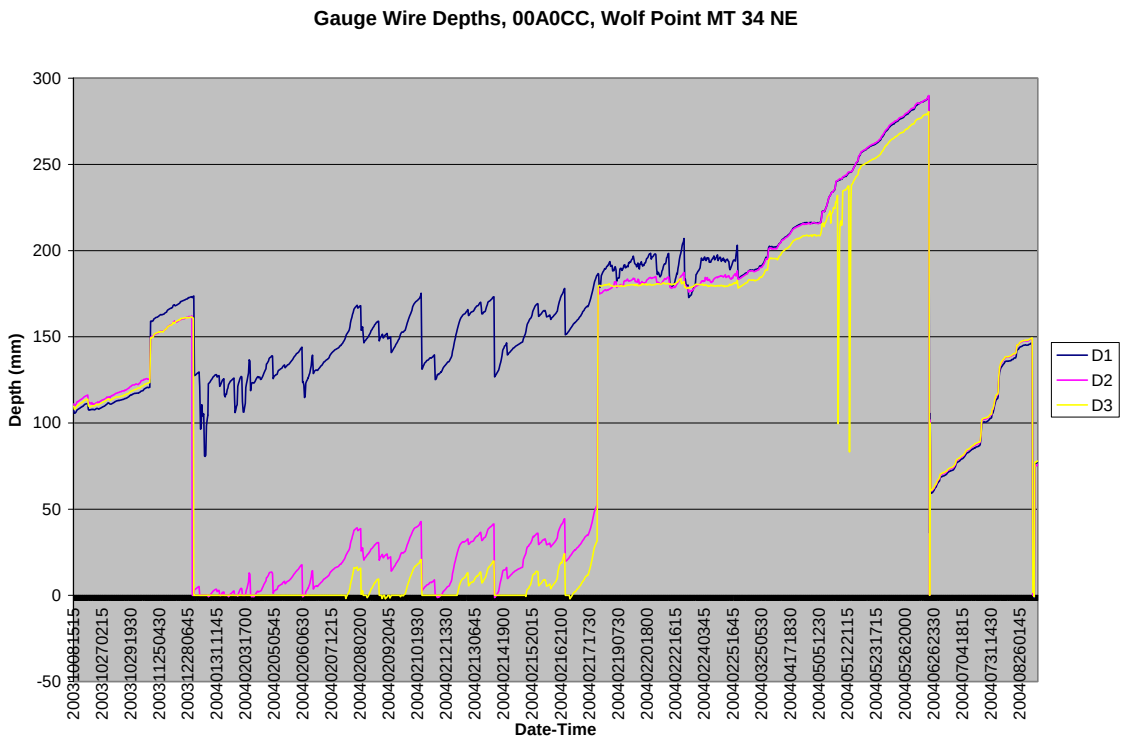


Figure 1. Gauge wire depths for Wolf Point MT 34 NE for October 1, 2003 through 30 September 2004. D1, D2 and D3 are the three wire depth measurements.

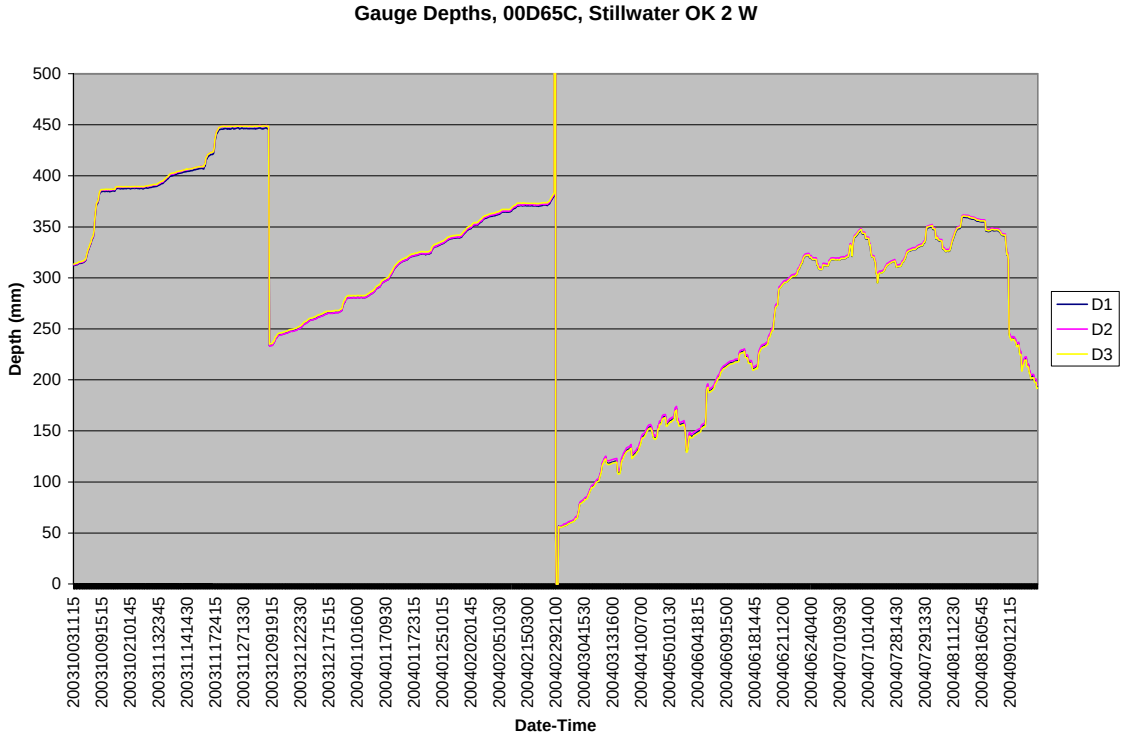


Figure 2. Gauge wire depths for Stillwater OK 2 W for October 1, 2003 through 30 September 2004. D1, D2 and D3 are the three wire depth measurements.

2) Minimize the number and magnitude of false reports. By smoothing "noise" in the depth measurements, using a time-averaged reference level also alleviates this problem. For stations where they are available, the use of wetness sensor readings will provide a more complete solution to eliminate false reports.
3) Minimize the effects of small-amplitude temporal variations in gauge depths. The time-average reference level likewise helps to minimize this problem.
4) Make intelligent decisions on which wires to use in precipitation calculations. The wire depths should increase and decrease together, even if the values are not identical. This fact is used to determine which wires can participate in making the precipitation estimate. The coincidence of change is relaxed for heavy precipitation events where divergence of wire depth measurements has been observed in network data.
5) Minimize the impact of extraordinary events, e.g. gauge emptying, adding of antifreeze and/or oil, wire breaks, missing data, etc. Special logic within the algorithm will check for depths and changes within certain bounds.
6) Allow a) easy recalculation after the fact, b) recovery from periods of data loss, c) calculation within the normal ingest of data at NCDC. In general, the data arrive each hour at NCDC in three-hour groups, with the first two hours repeated from previous transmissions. The algorithm is designed to use the data in that format. Each three-hour set of data stands on its own and the precipitation estimates generated for the last of the three hours depends on no other information, allowing easy recalculation, etc.

4. Description of the Algorithm

a. Input variables and parameters

The USCRN data are generally transmitted in overlapping blocks of data for a three-hour period. That is, each hour the present and two past hours of data are transmitted. This overlap guarantees nearly 100% receipt of the USCRN data at NCDC. This form of data receipt is also very useful for the calculation of the official precipitation from the Geonor gauges and allows objective 6) above to be fulfilled.

The only information used in the computation of the official precipitation are the depth measurements from the three Geonor wires. (The wetness sensor readings, when available can also be used to confirm the presence or absence of possible precipitation. This possibility will not be considered further in this note.) The wire measurements are not independent in that they are measuring the same liquid depth, but other factors make them vary somewhat from each other. The algorithm for the calculation of USCRN official precipitation uses the three wire depths for the complete three-hour period. Calculation begins with the first of the three hours, even though only the precipitation for the last of the three hours is required, the precipitation for the first two hours having been calculated previously.

There are some parameters upon which the performance depends. There is a maximal amount of precipitation that can reasonably be expected during the time between depth measurement reports. The value PMAX is used to denote the limit used by the algorithm. It is used to detect the addition of antifreeze or oil to the gauge and is also used to detect emptying of the gauge. The value will depend upon the reporting interval. Suggested values are 40.0mm for the present 15-minute interval and 25.0mm for the future 5-minute reporting interval. There is also a lower limit to the amount of precipitation that can be detected. The parameter EPS is used for this limit. Its value is 0.20 mm. Finally, there is the parameter ,NAVG, which is the maximum number of previous depths whose average determines the depth reference level for each wire. Its value corresponds to 2 hours of time: 8 for 15-minute interval; and 24 for 5-minute interval of data.

Computation of precipitation uses three hours of data as discussed above, but precipitation is valid only for the last hour. The results of the calculations made for the first two of the three hours are for the purpose of establishing valid reference levels and the likelihood of preceding precipitation; these results are not saved.

b. The calculation procedure

Proceeding from the second data time, from the first of the three hours of data, the calculations are as described below.

From the three hours of depth data, the depths for the present and previous time step are extracted. These are labeled D(j,k), where j = (1,2,3) for the three wires and k = (1,2) for the present and previous time step.

Some variables, used in the decision process are initialized. W1, W2 and W3 are used to indicate whether a wire's depth is used in the calculation. These are initially set to false (or 0). The precipitation (precip) is also initialized to 0.

The index I is used for the time step in the 3-hour period of available depth data. NREF is used to tell how many time steps to average for the reference period. It has dimension 3, having a (possibly) different value for each wire. And NPRECIP tells how many time steps since precipitation, initially set to a large value. And NPHR is the number of time steps per hour. The depth reference levels for each wire (DREF(j)) are set as follows:

(When the index j is shown and its value is not explicitly given, it is understood that the computation is performed for all values of this index.)

If I = 2 then
DREF(j) = D(j,1) Set the initial value of reference to first depth.
NREF(j) = 0
NPRECIP = 100 (any value > 2*NPHR)

```
Else
    If precip last time step
        DREF(j) = D(j,2)  Set reference depth to previous depth.
        NPRECIP = 1
        NREF(j) = 0
    Else If precip in last 2 hours
        NPRECIP = NPRECIP+1
        For j = 1 to 3
            If depth > 0. and depth < 600. and |D(j,2) – D(j,1)| < PMAX
                DREF(j) = DREF(j)  Keep present reference level.
                NREF(j) = NREF(j)+1 (limited to 2*NPHR)
            Else
                DREF(j) = D(j,1)  Set reference to current depth.
                NREF(j) = 0
            End If
        End For
    Else (no precip in last 2 hours)
        NPRECIP = NPRECIP+1
        For j = 1 to 3
            If depth > 0. and depth < 600. and |D(j,2) – D(j,1)| < PMAX
                NREF(j) = NREF(j)+1 (limited to 2*NPHR)
            Else
                NREF(j) = 0
            End If
            Average depths backward in time for NREF(j) times,
            using current depth if NREF(j) = 0
        End For
    End If
End If
```

In order to determine which wires will participate in the calculation of precipitation, the depth change from the reference value, *DREF(j)*, for each wire is first calculated for the depths, *D(j,k)*. If the calculated value is less than 0. or greater than PMAX, it is set to -1. Next, the inter-wire differences of the depth changes, δ_{ij}, are calculated. Thus,

$$DC(j) = D(j,1) - DREF(j)$$

and

$$\delta_{ij} = \left| DC(i) - DC(j) \right|.$$

for *ij* = {12,13,23}. If either DC(i) < EPS or DC(j) < EPS, then δ_{ij} is set to 100.

Now it can be determined which wires to use in the precipitation calculation by examination of the inter-wire depth change differences. Intermediate variables O_{ij} are used in this determination. If δ_{ij} is less than EPS for any of the pairs of ij, then O_{ij} is set

to true. Otherwise O_{ij} is false. The following Table 1. determines the value of a variable indicating whether a wire will be used in the precipitation calculation. Each W_i corresponds to the triplet of values for O_{12}, O_{13} and O_{23}. If W_i is true, then the corresponding wire depth is used in the precipitation calculation.

Table 1. Truth table to determine which wires to use in the precipitation calculation.

O_{12}	O_{13}	O_{23}		W_1	W_2	W_3
T	T	T	$\Rightarrow$	T	T	T
T	T	F	$\Rightarrow$	T	F	F
T	F	T	$\Rightarrow$	F	T	F
F	T	T	$\Rightarrow$	F	F	T
T	F	F	$\Rightarrow$	T	T	F
F	T	F	$\Rightarrow$	T	F	T
F	F	T	$\Rightarrow$	F	T	T
F	F	F	$\Rightarrow$	F	F	F

In the case when $W_1 = W_2 = W_3$ = F (false), then a special calculation is made to determine which wires will be used in the precipitation calculation. This is done to allow a greater difference in the inter-wire depth changes in cases of large precipitation rate. This calculation is made only if δ_{ij} > EPS and $\delta_{ij} < 100$ for all pairs of $ij = \{12,13,23\}$. In this case the variables R_{ij} are calculated from

R_{ij} = T if δ_{ij} /(DC(i) + DC(j)) ≤ 0.1; otherwise R_{ij} = F.

A truth table the same as the one above is used in this case to determine W_1, W_2 and W_3, but with O_{ij} replaced with R_{ij}. The precipitation is the average of the DC(j) for the wires for which W_j is true.

Each time is considered consecutively through the three-hour time period, beginning with the second time, with the precipitation values from the last hour considered to be the official values.

5. Examples of Precipitation Calculations

Several stations were chosen to test the precipitation algorithm because they presented problems for the present algorithm. Data from October 1, 2003 through September 30, 2004 were used. The computations for Wolf Point, MT 29 ENE are shown in Figure 3. Even in the presence of a wire break and gauge emptying, the accumulated precipitation is well-behaved.

Figure 4. shows the wire depths and accumulated precipitation for Lincoln, NE 11 SW. There are a number points of interest. Early in the period, around November 20, 2003, there is an increase of over 100 mm (anti-freeze addition) and very little response in the accumulated precipitation, as is proper. The small increase is caused by the depths taking some time to settle down; the resulting precipitation would be eliminated by use of the wetness sensor readings. The two times when the gauge is emptied are likewise

handled properly. It does appear that there is nearly coincident precipitation in July 2004, but the precipitation actually precedes by two days when the gauge is emptied. Note also that the method properly calculates no precipitation during the period beginning June 21, 2004 in which the depth is essentially constant. There is enough noise in the depths that the present method would have false tips during this period. Finally, the jump in depth, from unknown cause, in early September 2005 is handled well, with no adverse reflection in the precipitation. Note that in these figures the time does not advance uniformly as only times when there was precipitation diagnosed are included.

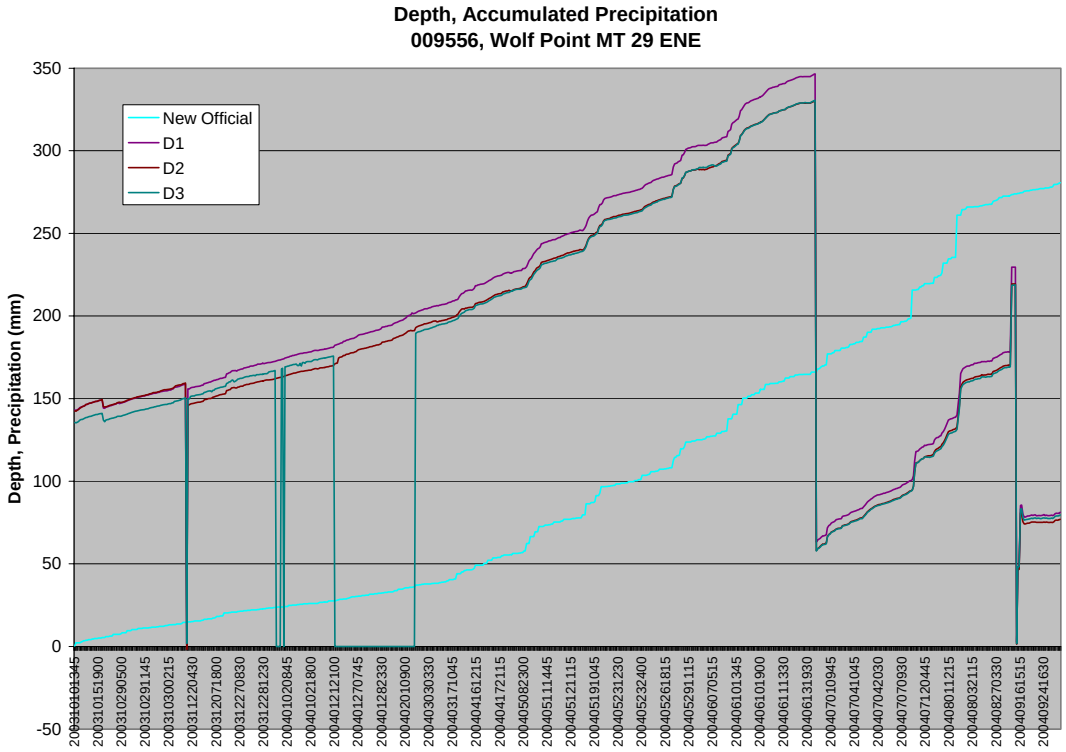


Figure 3. Wire depths and accumulated precipitation for Wolf Point MT 29 ENE for the period of October 1, 2003 through September 30, 2004.

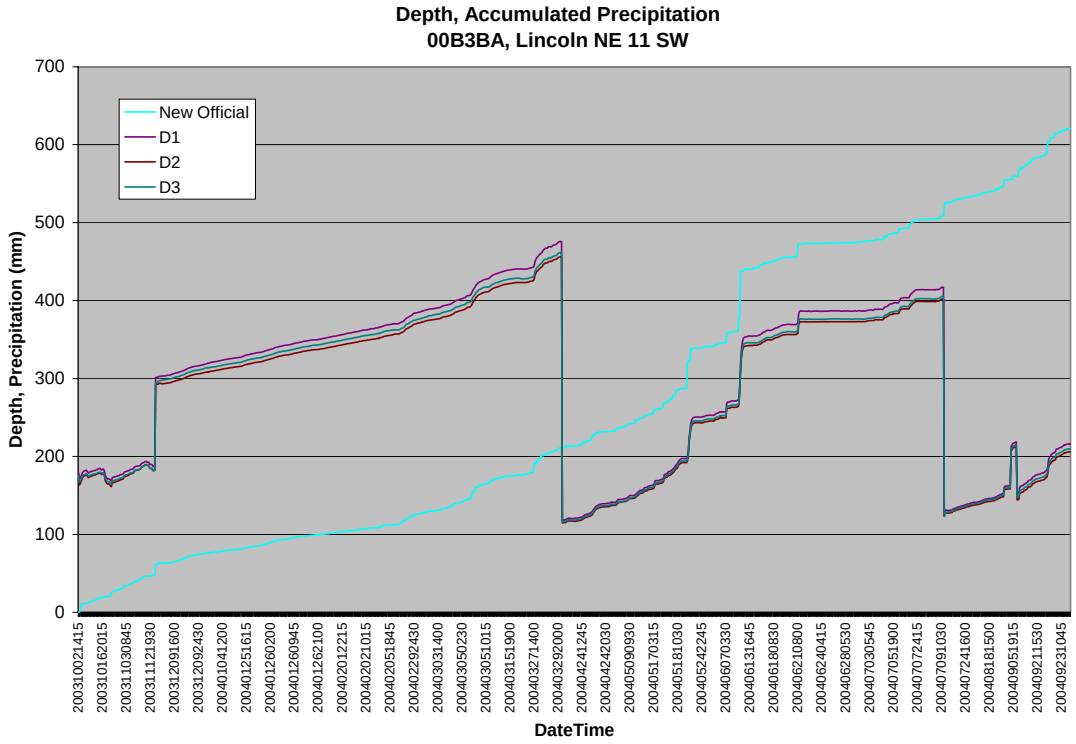
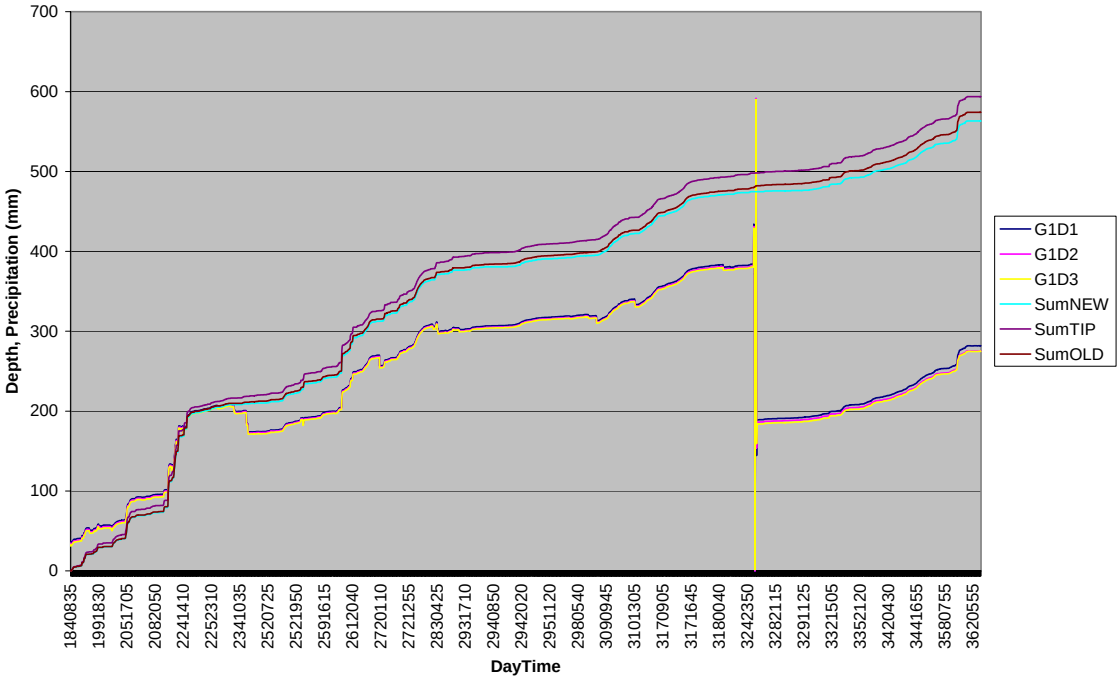


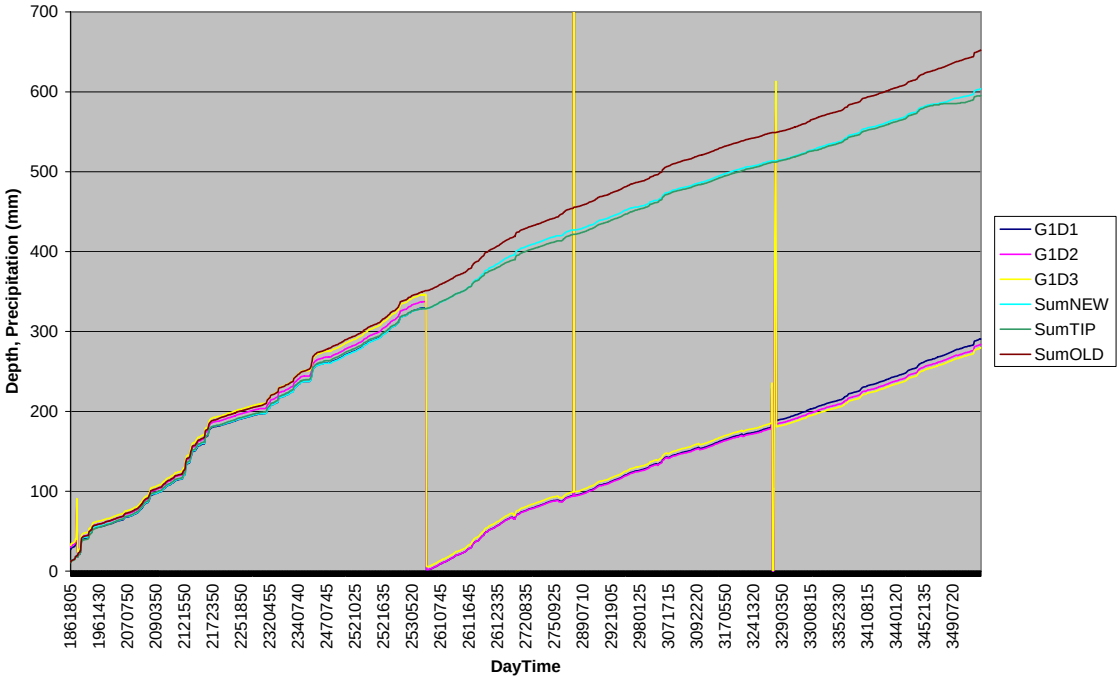
Figure 4. Wire depths and accumulated precipitation for Lincoln NE 11 ENE for the period of October 1, 2003 through September 30, 2004.

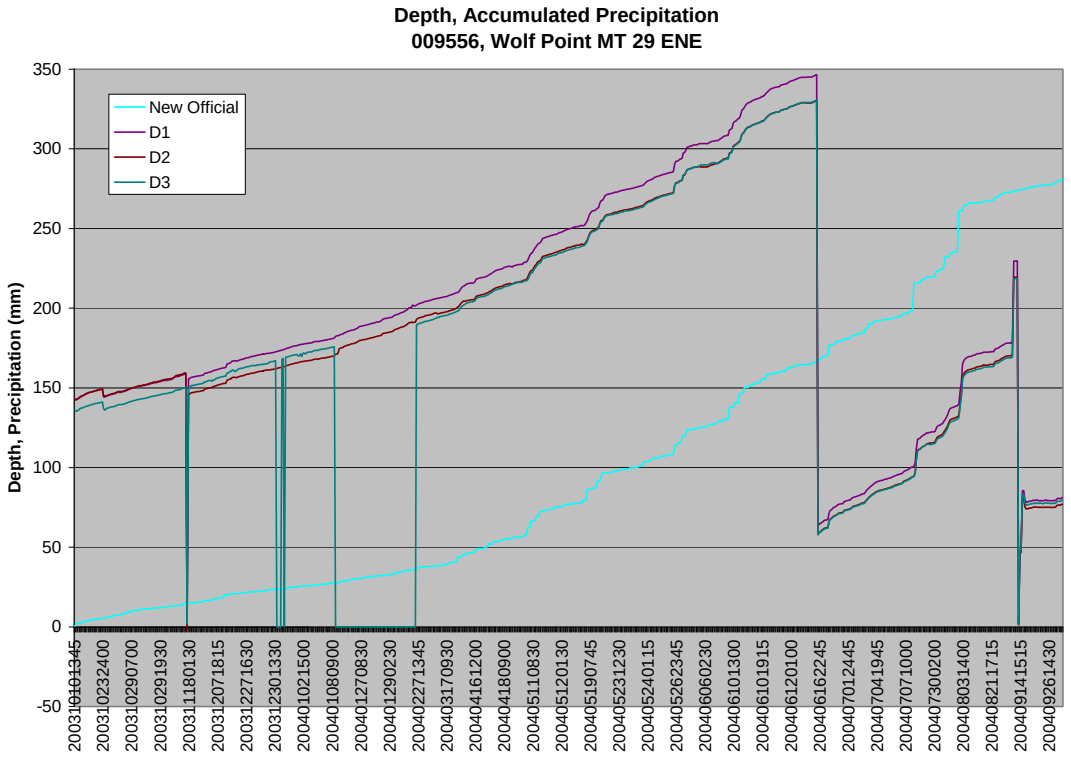
Accumulated Precipitation for Sterling, Johnstown and 13 CRN stations.

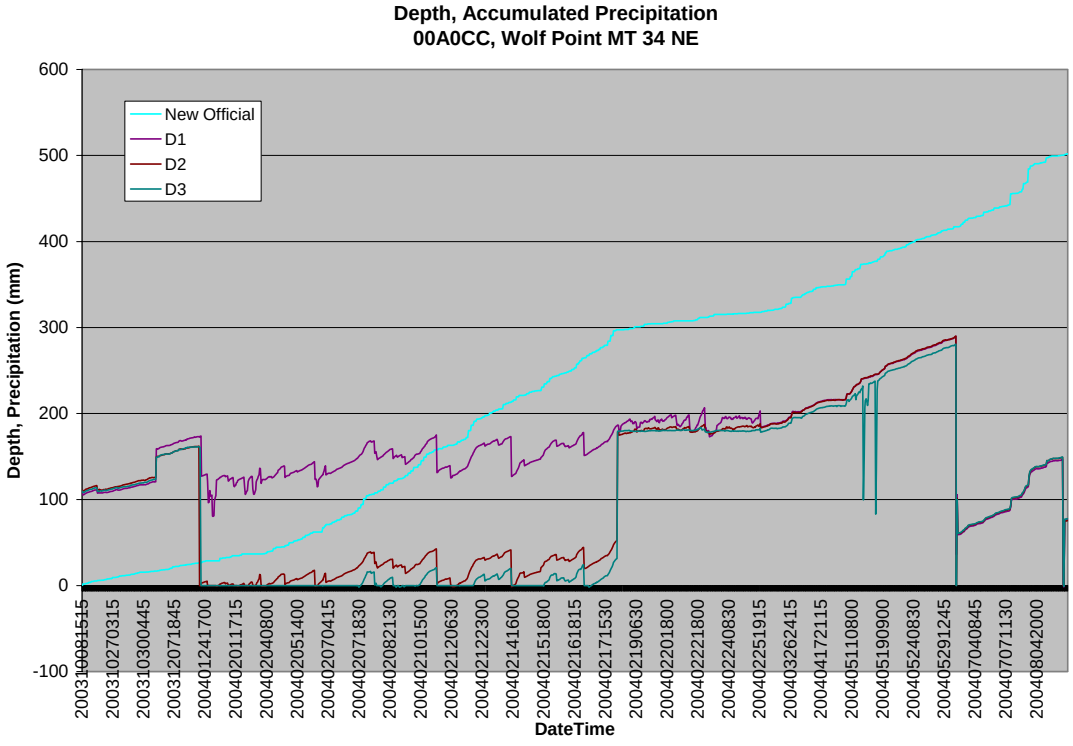
Depth, Accumulated Precipitation, Sterling VA

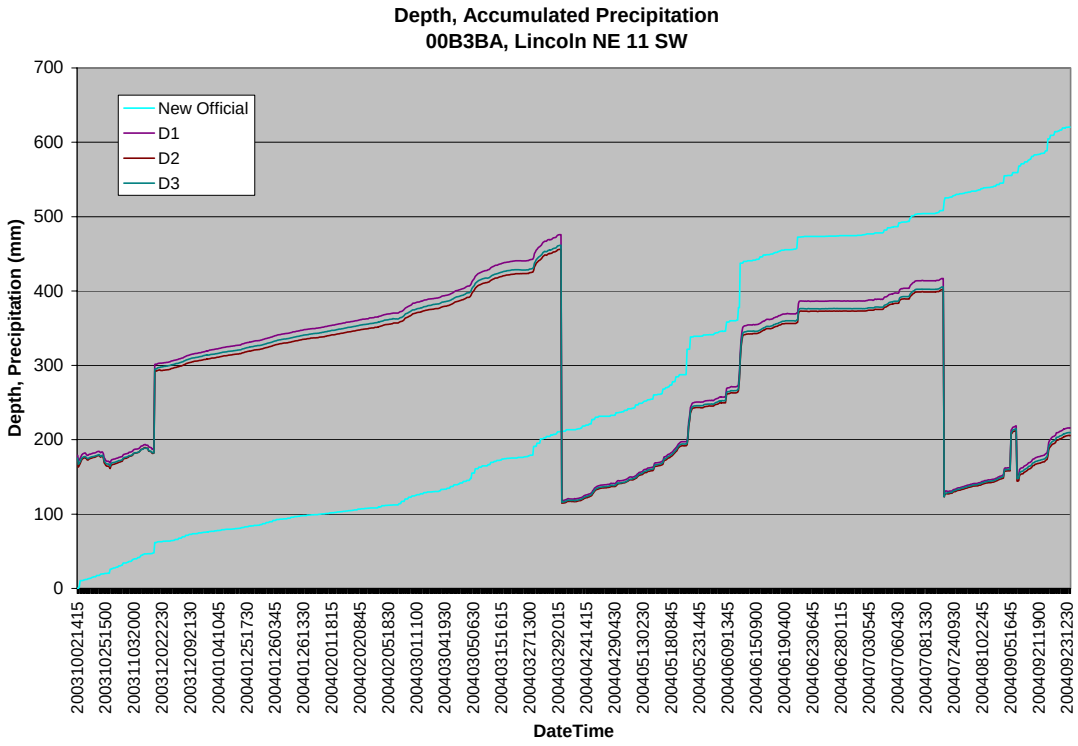


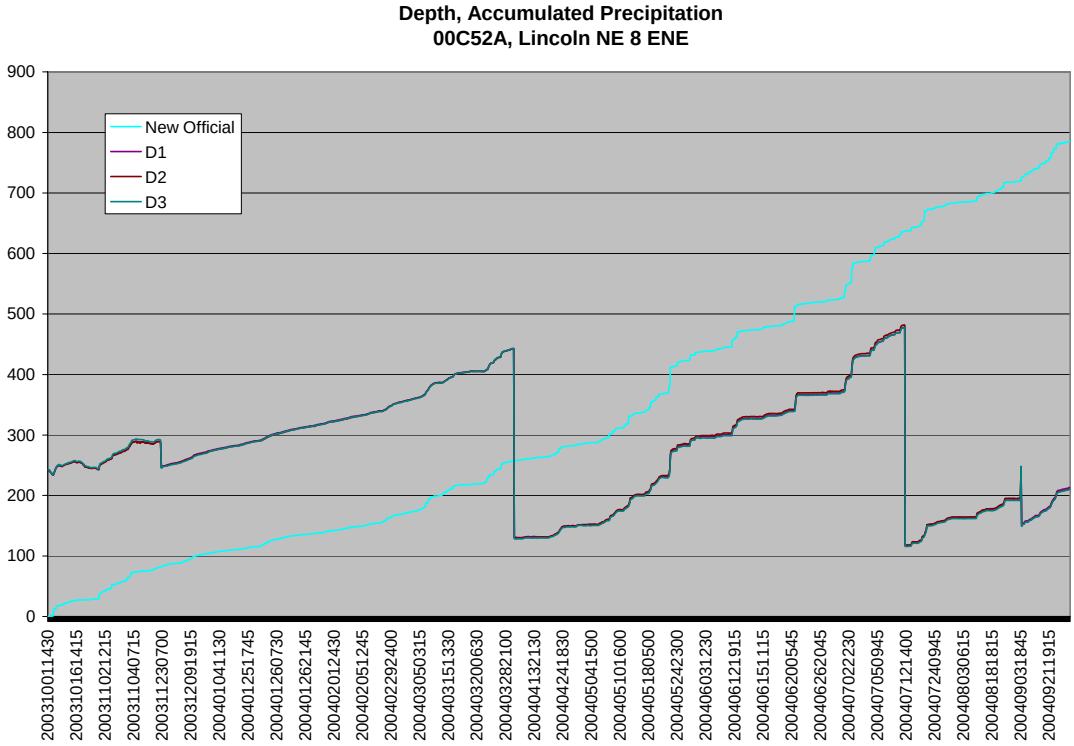
Depth, Accumulated Precipitation, Johnstown PA



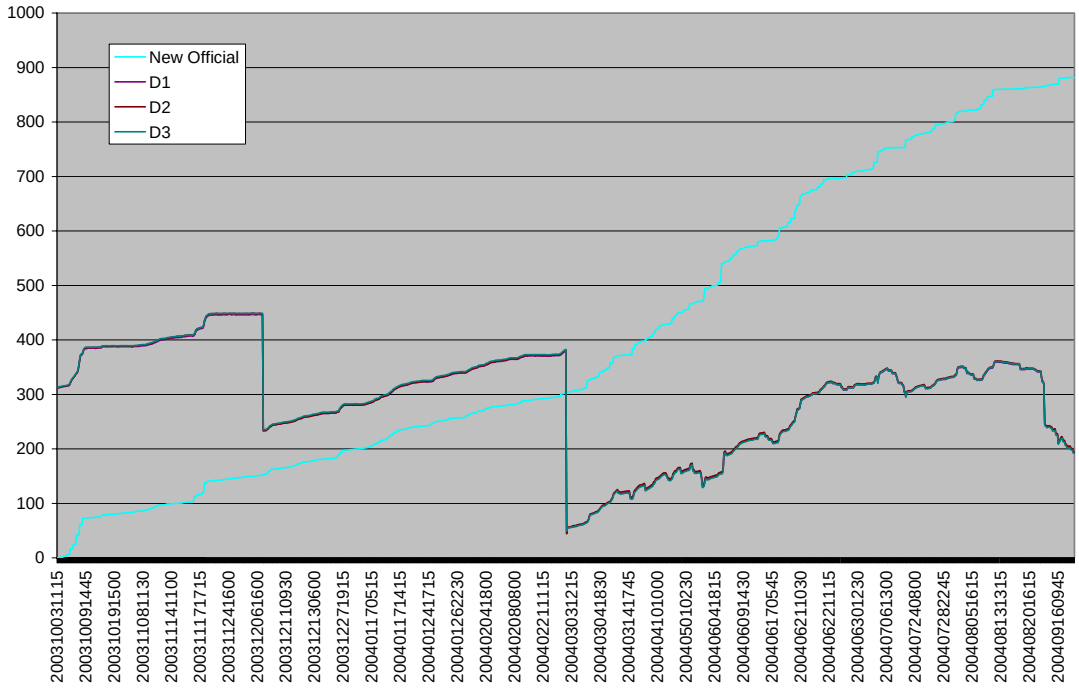

Depth, Accumulated Precipitation
009556, Wolf Point MT 29 ENE
New Official
D1
D2
D3
Depth, Precipitation (mm)
350
300
250
200
150
100
50
0
-50


Depth, Accumulated Precipitation
00A0CC, Wolf Point MT 34 NE
New Official
D1
D2
D3
Depth, Precipitation (mm)
600
500
400
300
200
100
0
-100
DateTime

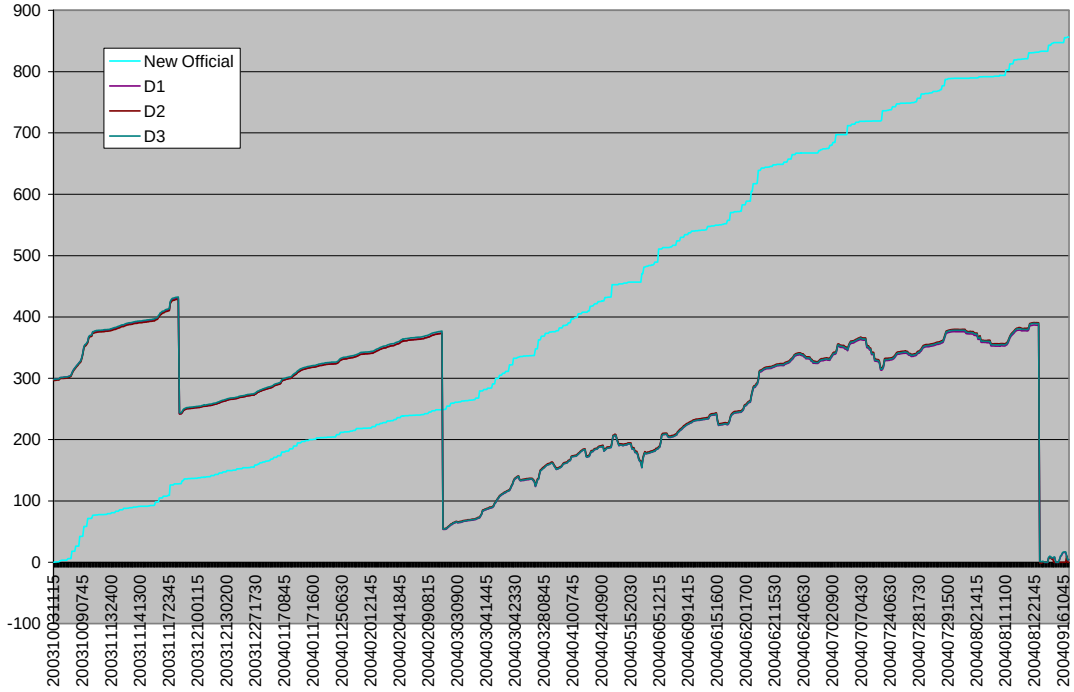

Depth, Accumulated Precipitation
00B3BA, Lincoln NE 11 SW
New Official
D1
D2
D3
Depth, Precipitation (mm)
0
100
200
300
400
500
600
700
200310021415
200310251500
200311032000
200312022230
200312092130
200401041045
200401251730
200401260345
200401261330
200402011815
200402020845
200402051830
200403011100
200403041930
200403050630
200403151615
200403271300
200403292015
200404241415
200404290430
200405130230
200405180845
200405231445
200406091345
200406150900
200406190400
200406230645
200406280115
200407030545
200407060430
200407081330
200407240930
200408102245
200409051645
200409211900
200409231230
DateTime

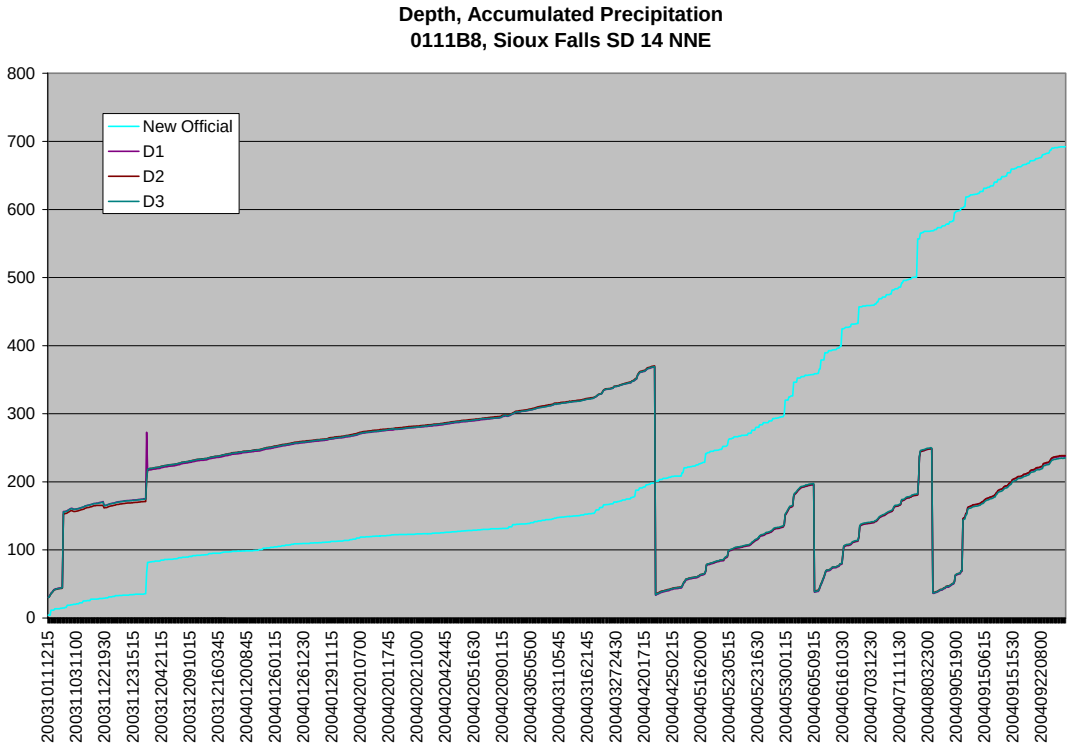

Depth, Accumulated Precipitation
00C52A, Lincoln NE 8 ENE
New Official
D1
D2
D3
0
100
200
300
400
500
600
700
800
900
200310011430
200310161415
200311021215
200311040715
200311230700
200312091915
200401041130
200401251745
200401260730
200401262145
200402012430
200402051245
200402292400
200403050315
200403151330
200403200630
200403282100
200404132130
200404241830
200405041500
200405101600
200405180500
200405242300
200406031230
200406121915
200406151115
200406200545
200406262045
200407022230
200407050945
200407121400
200407240945
200408030615
200408181815
200409031845
200409211915

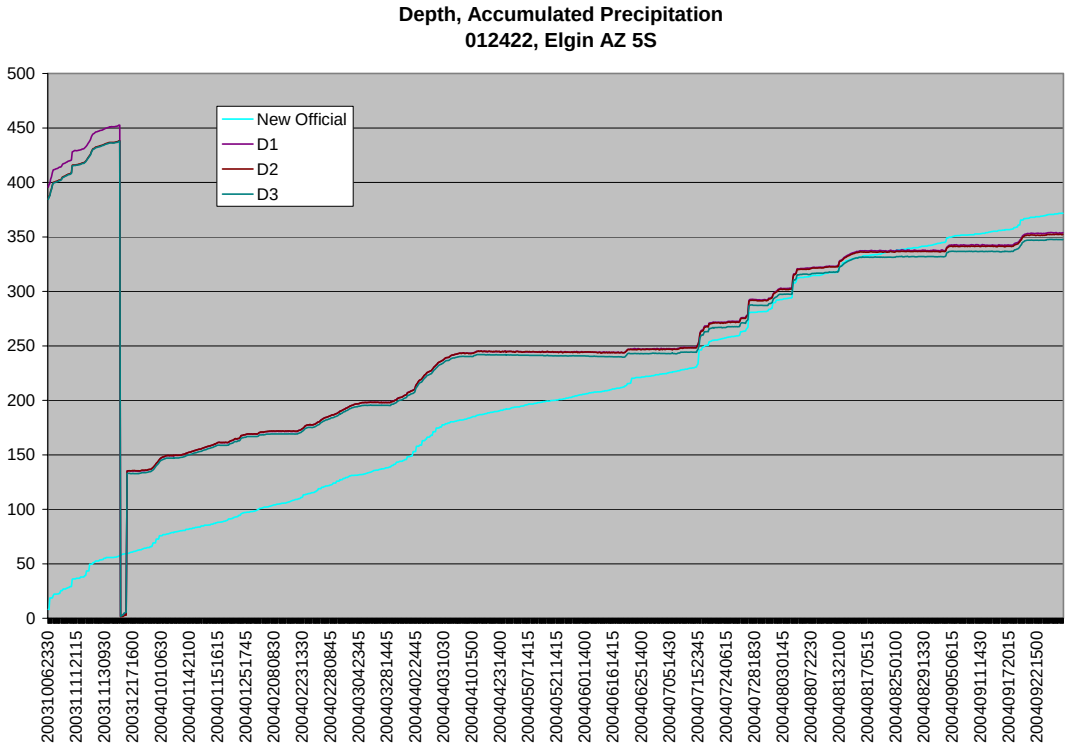
Depth, Accumulated Precipitation
00D65C, Stillwater OK 2 W



Depth, Accumulated Precipitation
00E3C6, Stillwater OK 5WNW

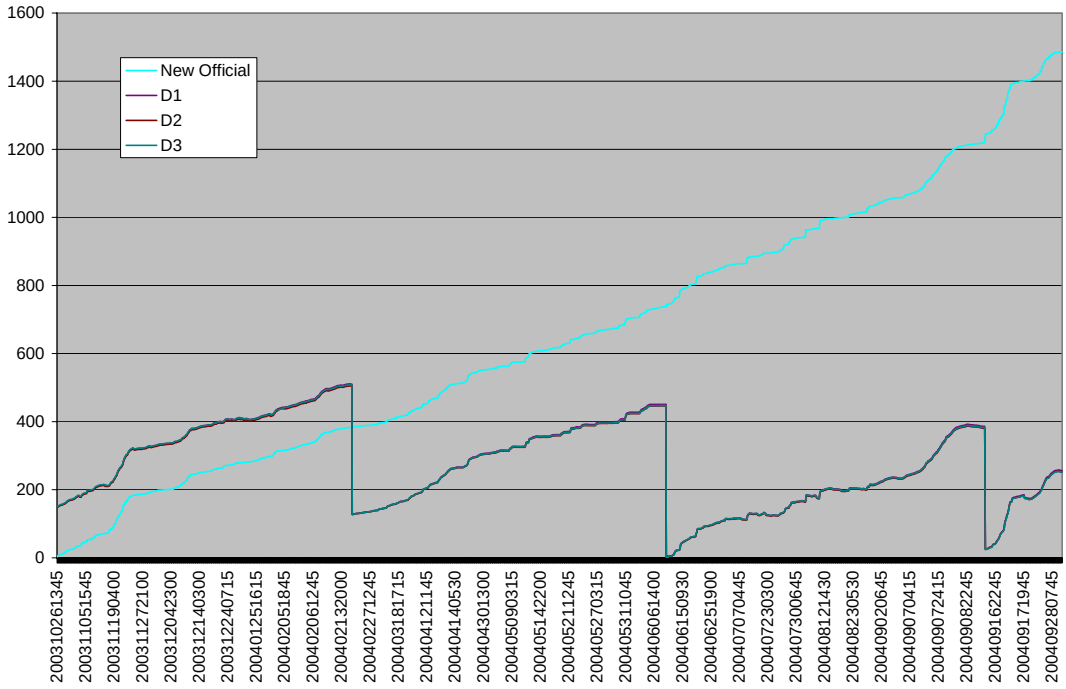




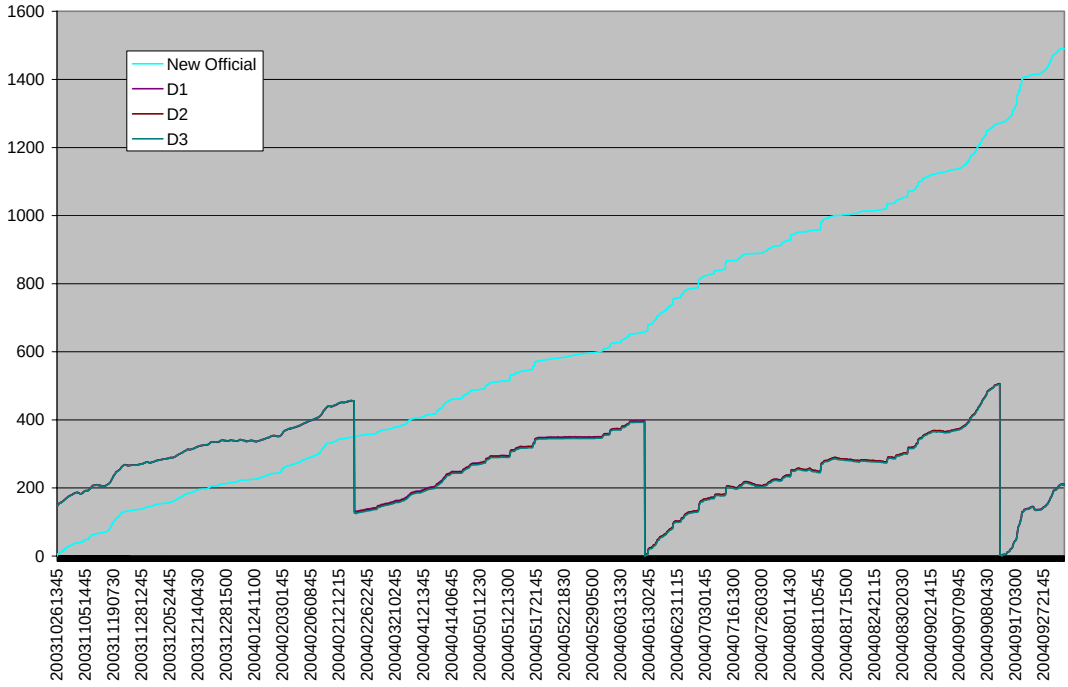


(Note! Amplitude of noise is too large to eliminate without using wetness sensor.)

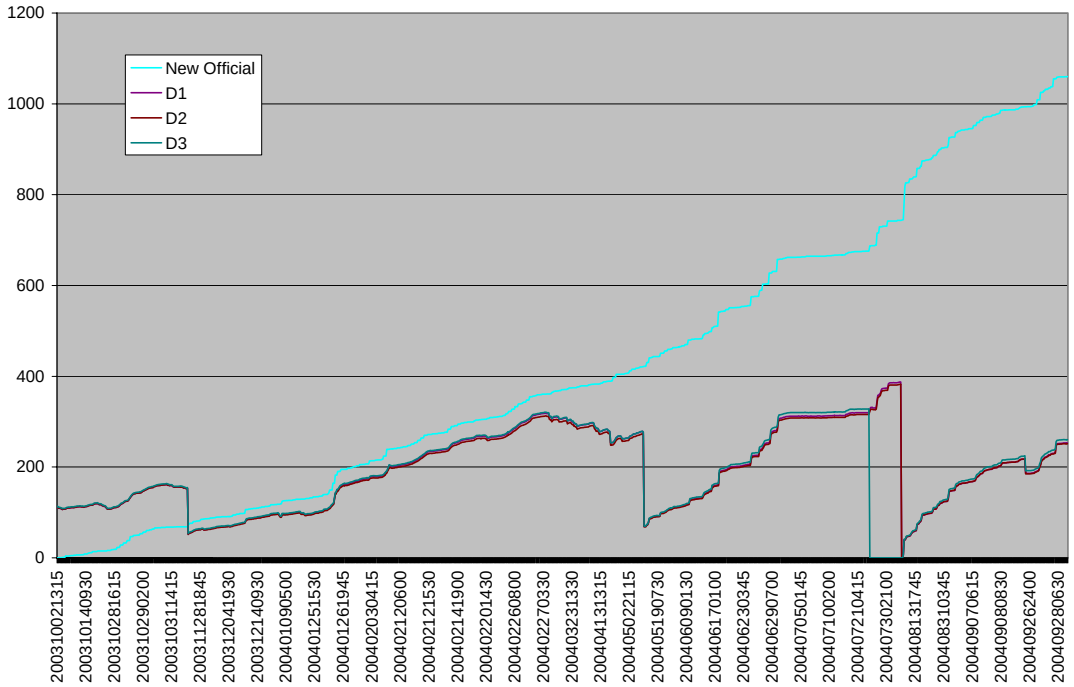
Depth, Accumulated Precipitation
0246CA, Asheville NC 8 SSW



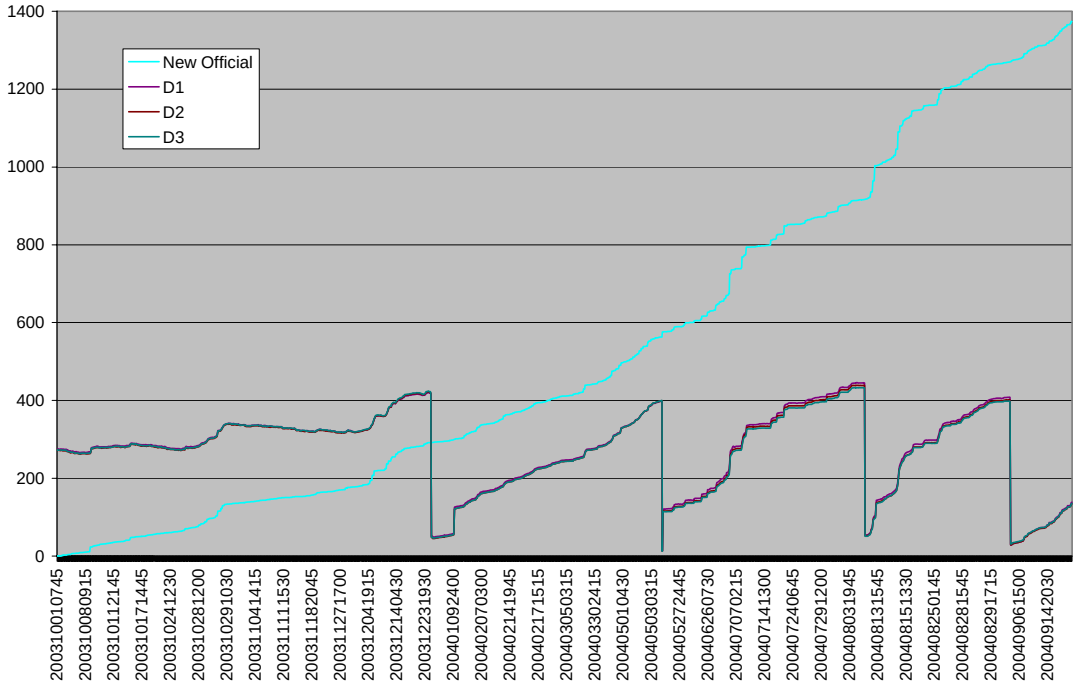
Depth, Accumulated Precipitation
0255BC, Asheville NC 13 S

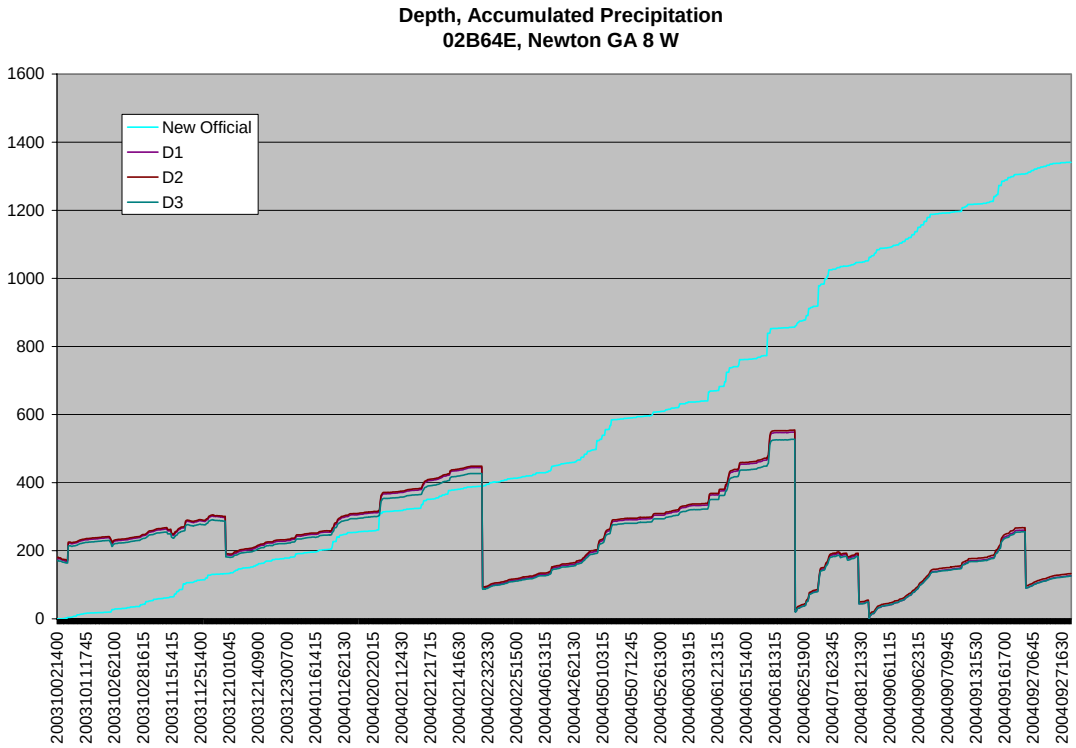


Depth, Accumulated Precipitation
0283D4, Blackville SC 3W



Depth, Accumulated Precipitation
0290A2, McClellanville SC 7 NE





6. Disclaimer

Mention of a commercial company or product is for information purposes only, and does not constitute an endorsement by NOAA. Use for publicity or advertising purposes of information from this publication concerning proprietary products or the tests of such products is not authorized.

Appendix 1

```
!*****************************************************************************
      SUBROUTINE POFFBETA6(P6,FLAG_OUT,DIN,NAVG,EPS,PMAX,NPRECIP,DREF6)

      PARAMETER (NPHR=12,XMSG=6999.)

!     THIS VERSION IS BASED UPON CODE FROM RODNEY BUCKNER, 11/23/04.
!     MODIFIED TO USE TIME-AVERAGED DEPTH AS REFERENCE LEVEL, 12/20/04.

!-------------------------------------------------------------------------------------------
!     INPUTS:
!       DIN     - DEPTH VALUES FOR 3 WIRES FOR LAST 3 HOURS, INDEX INCREASING WITH TIME
!       NAVG    - NUMBER OF TIMES TO AVERAGE THE DEPTH FOR REFERENCE LEVEL
!       EPS     - THE SMALLEST DEPTH CHANGE GIVING PRECIPITATION
!       PMAX    - THE LARGEST DEPTH CHANGE INTERPRETED AS PRECIPITATION
!                 IF DEPTH CHANGE > PMAX THIS IS DUE TO MANUAL ADDITION TO GAUGE
!                 IF DEPTH CHANGE < -PMAX THIS IS LIKELY DUE TO EMPTYING THE GAUGE
!
!     OUTPUTS:
!       P6      - PRECIPITATION (MM) (THIS ARRAY IS FOR 3-HOUR GROUP OF TIMES
!                 ONLY THE LAST NPHR VALUES ARE THE OFFICIAL PRECIPITATION)
!       FLAG_OUT – FLAG IS TRUE FOR HEAVY PRECIPITATION OF ‘FFF’ CASE
!-------------------------------------------------------------------------------------------

      REAL*4 POFF,PB6,D,EPS,BAD,P6(1:3*NPHR),DREF6(1:3,1:3*NPHR)
      REAL*4 DELTA(1:3),DREF(1:3),DD(1:3,1:2)
      REAL*4 DELTA12,DELTA13,DELTA23
      REAL*4 DIN(1:3,1:3*NPHR)
      INTEGER NPRECIP, NREF(1:3)
      LOGICAL W1,W2,W3,FLAG_OUT,FIRST

      BAD = XMSG

!     FILL DD(J,1) AND DD(J,2) AS NEEDED.
!     THESE ARE THE PRESENT AND PREVIOUS TIME LEVEL DEPTHS FOR THE 3 WIRES.

      FIRST = .TRUE.
      P6 = 0.           ! INITIALIZE THE PRECIPITATION (ALL VALUES)
      DREF = BAD

!     LOOP OVER TIME STEPS FOR 3 HOURS

      DO I=2,3*NPHR

        DO J=1,3
          DD(J,1) = DIN(J,I)
          DD(J,2) = DIN(J,I-1)
        ENDDO

!     INITIALIZE SOME QUANTITIES

        W1 = .FALSE.
        W2 = .FALSE.
        W3 = .FALSE.
        PB6 = 0.
        FLAG_OUT = .FALSE.

!     SET DREF,NPRECIP.

        IF(I.EQ.2) THEN   ! FIRST TIME FOR CALCULATING PRECIP WITHIN 3 HOURS
          NPRECIP = 100
          NREF = 0
          DO J=1,3
            DREF(J) = DIN(J,1)
          ENDDO
        ELSE
```

```
          IF(P6(I-1).NE.0.) THEN   ! PRECIP LAST TIME STEP
            NPRECIP = 1
            DO J=1,3
              DREF(J) = DD(J,2)   ! SET REFERENCE DEPTH TO PREVIOUS DEPTH
              NREF(J) = 0
            ENDDO
          ELSEIF(NPRECIP.LT.2*NPHR) THEN  ! PRECIP WITHIN LAST 2 HOURS
            NPRECIP = NPRECIP + 1
            DO J=1,3
              IF(DD(J,1).GE.0. .AND. DD(J,1).LT.600.   &        ! DEPTH IN RANGE
     &          .AND. ABS(DD(J,2)-DD(J,1)).LT.PMAX) THEN        ! CHANGE LESS THAN PMAX?
                DREF(J) = DREF(J)                               ! KEEP CURRENT REFERENCE
DEPTH
                NREF(J) = MIN(2*NPHR,NREF(J)+1)                 ! INCREMENT NREF
              ELSE   ! BAD DEPTH OR DEPTH CHANGE
                DREF(J) = DD(J,1)  ! SET REFERENCE TO CURRENT DEPTH
                NREF(J) = 0
              ENDIF
            ENDDO
          ELSE     ! NO PRECIP WITHIN THE LAST 2 HOURS
            NPRECIP = NPRECIP + 1
            DO J=1,3
              IF(DD(J,1).GE.0. .AND. DD(J,1).LT.600.   &        ! DEPTH IN RANGE
     &          .AND. ABS(DD(J,2)-DD(J,1)).LT.PMAX) THEN        ! CHANGE LESS THAN PMAX?
                NREF(J) = MIN(2*NPHR,NREF(J)+1)
              ELSE
                NREF(J) = 0   ! BAD DEPTH OR DEPTH CHANGE; START NEW REFERENCE
              ENDIF
              IF(NREF(J).EQ.0) THEN
                DREF(J) = DD(J,1)
              ELSE
                ISUM = 0
                SUM = 0
                DO II=1,NREF(J)
                  ISUM = ISUM + 1
                  SUM = SUM + DIN(J,I-II)
                ENDDO
                DREF(J) = SUM/ISUM
              ENDIF
            ENDDO
          ENDIF
        ENDIF

        DO J=1,3
          DREF6(J,I) = DREF(J)
        ENDDO

!     CALCULATE THE TIME CHANGE OF DEPTHS.  PUT INTO DELTA.

        DO J=1,3
          DELTA(J) = DD(J,1) - DREF(J)
        ENDDO

        DO J=1,3
          IF(DELTA(J).LT.0. .OR. DELTA(J).GE.PMAX) DELTA(J) = -1.
        ENDDO

!     CALCULATE THE DEPTH CHANGE DIFFERENCES BETWEEN WIRES
!     AND PUT INTO DELTA12, DELTA13, DELTA23.

        IF(DELTA(1).GE.EPS .AND. DELTA(2).GE.EPS) THEN
          DELTA12 = ABS(DELTA(1) - DELTA(2))
        ELSE
          DELTA12 = 100.
        ENDIF

        IF(DELTA(1).GE.EPS .AND. DELTA(3).GE.EPS) THEN
          DELTA13 = ABS(DELTA(1) - DELTA(3))
        ELSE
          DELTA13 = 100.
        ENDIF
```

```
        IF(DELTA(2).GE.EPS .AND. DELTA(3).GE.EPS) THEN
          DELTA23 = ABS(DELTA(2) - DELTA(3))
        ELSE
          DELTA23 = 100.
        ENDIF

!     DETERMINE WIRES VALID TO PARTICIPATE IN PRECIP CALCULATION.

        IF(    (DELTA12.LE.EPS .AND. DELTA13.LE.EPS .AND. DELTA23.LE.EPS)   &
     &    .OR. (DELTA12.LE.EPS .AND. DELTA13.LE.EPS .AND. DELTA23.GT.EPS)   &
     &    .OR. (DELTA12.LE.EPS .AND. DELTA13.GT.EPS .AND. DELTA23.GT.EPS)   &
     &    .OR. (DELTA12.GT.EPS .AND. DELTA13.LE.EPS .AND. DELTA23.GT.EPS)) THEN
          W1 = .TRUE.
        ENDIF

        IF((DELTA12.LE.EPS .AND. DELTA13.LE.EPS .AND. DELTA23.LE.EPS)   &
     &    .OR. (DELTA12.LE.EPS .AND. DELTA13.GT.EPS .AND. DELTA23.LE.EPS)   &
     &    .OR. (DELTA12.LE.EPS .AND. DELTA13.GT.EPS .AND. DELTA23.GT.EPS)   &
     &    .OR. (DELTA12.GT.EPS .AND. DELTA13.GT.EPS .AND. DELTA23.LE.EPS)) THEN
          W2 = .TRUE.
        ENDIF

        IF((DELTA12.LE.EPS .AND. DELTA13.LE.EPS .AND. DELTA23.LE.EPS)   &
     &    .OR. (DELTA12.GT.EPS .AND. DELTA13.LE.EPS .AND. DELTA23.LE.EPS)   &
     &    .OR. (DELTA12.GT.EPS .AND. DELTA13.LE.EPS .AND. DELTA23.GT.EPS)   &
     &    .OR. (DELTA12.GT.EPS .AND. DELTA13.GT.EPS .AND. DELTA23.LE.EPS)) THEN
          W3 = .TRUE.
        ENDIF

!     CALCULATE THE PRECIPITATION.

        PB6 = 0.
        IF(W1.AND.W2.AND.W3) THEN
          PB6 = (DELTA(1)+DELTA(2)+DELTA(3))/3.
        ELSEIF(W1.AND.W2.AND..NOT.W3) THEN
          PB6 = (DELTA(1)+DELTA(2))/2.
        ELSEIF(W1.AND..NOT.W2.AND.W3) THEN
          PB6 = (DELTA(1)+DELTA(3))/2.
        ELSEIF(.NOT.W1.AND.W2.AND.W3) THEN
          PB6 = (DELTA(2)+DELTA(3))/2.
        ELSEIF(W1.AND..NOT.W2.AND..NOT.W3) THEN
          PB6 = DELTA(1)
        ELSEIF(.NOT.W1.AND.W2.AND..NOT.W3) THEN
          PB6 = DELTA(2)
        ELSEIF(.NOT.W1.AND..NOT.W2.AND.W3) THEN
          PB6 = DELTA(3)
        ELSEIF(.NOT.W1.AND..NOT.W2.AND..NOT.W3) THEN

!     FOLLOWING IS FOR 'FFF' CASE.

          IF((DELTA12.GT.EPS .AND. DELTA12.LT.100.) .OR.   &
     &       (DELTA13.GT.EPS .AND. DELTA13.LT.100.) .OR.   &
     &       (DELTA23.GT.EPS .AND. DELTA23.LT.100.)) THEN
            FLAG_OUT = .TRUE.
            IF(DELTA(1)+DELTA(2).GT.0. .AND. DELTA(1)+DELTA(3).GT.0.   &
     &        .AND. DELTA(2)+DELTA(3).GT.0.) THEN
              IF((DELTA12/(DELTA(1)+DELTA(2)).LE.0.1 .AND.   &
     &          DELTA13/(DELTA(1)+DELTA(3)).LE.0.1 .AND.   &
     &          DELTA23/(DELTA(2)+DELTA(3)).LE.0.1) .OR.   &

     &         (DELTA12/(DELTA(1)+DELTA(2)).LE.0.1 .AND.   &
     &          DELTA13/(DELTA(1)+DELTA(3)).LE.0.1 .AND.   &
     &          DELTA23/(DELTA(2)+DELTA(3)).GE.0.1) .OR.   &

     &         (DELTA12/(DELTA(1)+DELTA(2)).LE.0.1 .AND.   &
     &          DELTA13/(DELTA(1)+DELTA(3)).GE.0.1 .AND.   &
     &          DELTA23/(DELTA(2)+DELTA(3)).GE.0.1) .OR.   &

     &         (DELTA12/(DELTA(1)+DELTA(2)).GE.0.1 .AND.   &
     &          DELTA13/(DELTA(1)+DELTA(3)).LE.0.1 .AND.   &
```

```
     &            DELTA23/(DELTA(2)+DELTA(3)).GE.0.1)) THEN
                W1 = .TRUE.
              ENDIF
              IF((DELTA12/(DELTA(1)+DELTA(2)).LE.0.1 .AND.   &
     &            DELTA13/(DELTA(1)+DELTA(3)).LE.0.1 .AND.   &
     &            DELTA23/(DELTA(2)+DELTA(3)).LE.0.1) .OR.   &

     &           (DELTA12/(DELTA(1)+DELTA(2)).LE.0.1 .AND.   &
     &            DELTA13/(DELTA(1)+DELTA(3)).GE.0.1 .AND.   &
     &            DELTA23/(DELTA(2)+DELTA(3)).LE.0.1) .OR.   &

     &           (DELTA12/(DELTA(1)+DELTA(2)).LE.0.1 .AND.   &
     &            DELTA13/(DELTA(1)+DELTA(3)).GE.0.1 .AND.   &
     &            DELTA23/(DELTA(2)+DELTA(3)).GE.0.1) .OR.   &

     &           (DELTA12/(DELTA(1)+DELTA(2)).GE.0.1 .AND.   &
     &            DELTA13/(DELTA(1)+DELTA(3)).GE.0.1 .AND.   &
     &            DELTA23/(DELTA(2)+DELTA(3)).LE.0.1)) THEN
                W2 = .TRUE.
              ENDIF
              IF((DELTA12/(DELTA(1)+DELTA(2)).LE.0.1 .AND.   &
     &            DELTA13/(DELTA(1)+DELTA(3)).LE.0.1 .AND.   &
     &            DELTA23/(DELTA(2)+DELTA(3)).LE.0.1) .OR.   &

     &           (DELTA12/(DELTA(1)+DELTA(2)).GE.0.1 .AND.   &
     &            DELTA13/(DELTA(1)+DELTA(3)).LE.0.1 .AND.   &
     &            DELTA23/(DELTA(2)+DELTA(3)).LE.0.1) .OR.   &

     &           (DELTA12/(DELTA(1)+DELTA(2)).GE.0.1 .AND.   &
     &            DELTA13/(DELTA(1)+DELTA(3)).LE.0.1 .AND.   &
     &            DELTA23/(DELTA(2)+DELTA(3)).GE.0.1) .OR.   &

     &           (DELTA12/(DELTA(1)+DELTA(2)).GE.0.1 .AND.   &
     &            DELTA13/(DELTA(1)+DELTA(3)).GE.0.1 .AND.   &
     &            DELTA23/(DELTA(2)+DELTA(3)).LE.0.1)) THEN
                W3 = .TRUE.
              ENDIF
            ENDIF
          ENDIF

!     CALCULATE THE PRECIPITATION FOR 'FFF' CASE.

          IF(W1.AND.W2.AND.W3) THEN
              PB6 = (DELTA(1)+DELTA(2)+DELTA(3))/3.
          ELSEIF(W1.AND.W2.AND..NOT.W3) THEN
              PB6 = (DELTA(1)+DELTA(2))/2.
          ELSEIF(W1.AND..NOT.W2.AND.W3) THEN
              PB6 = (DELTA(1)+DELTA(3))/2.
          ELSEIF(W1.AND..NOT.W2.AND..NOT.W3) THEN
              PB6 = DELTA(1)
          ELSEIF(.NOT.W1.AND.W2.AND.W3) THEN
              PB6 = (DELTA(2)+DELTA(3))/2.
          ELSEIF(.NOT.W1.AND.W2.AND..NOT.W3) THEN
              PB6 = DELTA(2)
          ELSEIF(.NOT.W1.AND..NOT.W2.AND.W3) THEN
              PB6 = DELTA(3)
          ELSEIF(.NOT.W1.AND..NOT.W2.AND..NOT.W3) THEN
            PB6 = 0.
          ENDIF
        ENDIF

        P6(I) = PB6

!     END OF DO LOOP OVER 3 HOURS

      ENDDO

      RETURN
      END
!*******************************************************************************
```

Appendix 2

```
################################################################################
#     Method: p_calc
#     Author: Rodney Buckner
#       File: Crn.pm
#    Created: 1/30/2005
#   Synopsis: calculates precipt from current and 2 previous hours' gauge depths
#
# Parameters: array of gauge depths
#       NOTE: req'd structure of gauge depth array is:
#             hourhourhour - 3 hours of gauge depths ordered from oldest to newest,
#              within each hour w1w2w3,
#               within each w 1,2,3,...n
#               where n = number of samples per hour
#
# Modification History
#  Date    Who    Description
#
################################################################################
sub p_calc{
    my @depths = @_;

    croak "usage: p_calc(@depths), @depths % 3 == 0"  if (@depths % 3);

    # test for null values
    foreach (@depths){
        if (!defined($_)){
            return undef;
        }
    }

    my $n = @depths/3;
    my $nphr = $n/3;
    my $nprecipt;
    my $pmax = ($nphr==4) ? 40:25;

    # $eps1 - time step epsilon
    # $eps2 - interwire delta epsilon
    my ($eps1,$eps2) = (0.2,0.2);

    my @d1 = @depths[0 .. $nphr-1, $n .. $n + $nphr - 1, 2*$n .. 2*$n + $nphr - 1];
    my @d2 = @depths[$nphr .. $nphr + 3, $n + $nphr .. $n + $nphr + 3, 2*$n + $nphr ..
              2*$n + $nphr + 3];
    my @d3 = @depths[2*$nphr .. 2*$nphr + 3, $n + 2*$nphr .. $n + 2*$nphr + 3, 2*$n +
              2*$nphr .. 2*$n + 2*$nphr + 3];

    my @d = (\@d1,\@d2,\@d3);

    my @p = ();
    my @dref = ();
    my @nref = ();

    my $p_total = 0;

    for (my $i=0; $i<=$n-1; $i++){
        # determine reference depth, dref(j)
        if($i==0){
            $nprecipt = 100;
            for (my $j=0; $j<=2; $j++){
                $dref[$j]=$d[$j][0];
                $nref[$j] = 0;
            }
        }else{
            if($p[$i-1]!=0){
                $nprecipt = 1;
                for (my $j=0; $j<=2; $j++){
                    $dref[$j] = $d[$j][$i-1];
                    $nref[$j] = 0;
```

```
        }
    }elsif($nprecipt < 2*$nphr) {
        $nprecipt++;
        for (my $j=0; $j<=2; $j++){
            if(($d[$j][$i]>0 && $d[$j][$i]<600) &&
               (abs($d[$j][$i] - $d[$j][$i-1])<$pmax)){
                $dref[$j] = $dref[$j];
                $nref[$j] = min(2*$nphr,$nref[$j]+1);
            }else{
                $dref[$j] = $d[$j][$i];
                $nref[$j] = 0;
            }
        }
    }else{
        $nprecipt++;
        for (my $j=0; $j<=2; $j++){
            if(($d[$j][$i]>0 && $d[$j][$i]<600) &&
               (abs($d[$j][$i] - $d[$j][$i-1])<$pmax)){
                $nref[$j] = min(2*$nphr,$nref[$j]+1);
            }else{
                $nref[$j] = 0;
            }
            if($nref[$j]==0){
                $dref[$j] = $d[$j][$i];
            }else{
                my $isum = 0;
                my $sum = 0;
                for (my $k=0; $k<=$nref[$j]-1; $k++){
                    $isum++;
                    $sum += $d[$j][$i-$k-1];
                }
                $dref[$j] = $sum/$isum;
            }
        }
    }
}

my @delta = ();
my $pcpt;

# determine time step difference, delta(j)
for(my $j=0; $j<=2; $j++){
    $delta[$j] = $d[$j][$i] - $dref[$j];
    if($delta[$j]<0 || $delta[$j]>$pmax){$delta[$j]=-1;}
}

# determine interwire differences: d12, d13, d23
my $d12 = ($delta[0]>=$eps1 && $delta[1]>=$eps1) ? abs($delta[0]-$delta[1]) : 100;
my $d13 = ($delta[0]>=$eps1 && $delta[2]>=$eps1) ? abs($delta[0]-$delta[2]) : 100;
my $d23 = ($delta[1]>=$eps1 && $delta[2]>=$eps1) ? abs($delta[1]-$delta[2]) : 100;

my ($w1,$w2,$w3) = (0,0,0);

# validate wires
if (($d12 <= $eps2 && $d13 <= $eps2 && $d23 <= $eps2) ||
    ($d12 <= $eps2 && $d13 <= $eps2 && $d23 >  $eps2) ||
    ($d12 <= $eps2 && $d13 >  $eps2 && $d23 >  $eps2) ||
    ($d12 >  $eps2 && $d13 <= $eps2 && $d23 >  $eps2)){
    $w1=1;}

if (($d12 <= $eps2 && $d13 <= $eps2 && $d23 <= $eps2) ||
    ($d12 <= $eps2 && $d13 >  $eps2 && $d23 <= $eps2) ||
    ($d12 <= $eps2 && $d13 >  $eps2 && $d23 >  $eps2) ||
    ($d12 >  $eps2 && $d13 >  $eps2 && $d23 <= $eps2)){
    $w2=1;}

if (($d12 <= $eps2 && $d13 <= $eps2 && $d23 <= $eps2) ||
    ($d12 >  $eps2 && $d13 <= $eps2 && $d23 <= $eps2) ||
    ($d12 >  $eps2 && $d13 <= $eps2 && $d23 >  $eps2) ||
    ($d12 >  $eps2 && $d13 >  $eps2 && $d23 <= $eps2)){
    $w3=1;}
```

```
 # check the case of 3 invalid wires for an excessive precipt event
 if ($w1==0 && $w2==0 && $w3==0){
     if (($d12 > eps1 && $d12 < 100) ||
         ($d13 > eps1 && $d13 < 100) ||
         ($d23 > eps1 && $d23 < 100)){
         if ((($delta[0] + $delta[1]) > 0) &&
             (($delta[0] + $delta[2]) > 0) &&
             (($delta[1] + $delta[2]) > 0)){
             if ((($d12/($delta[0]+$delta[1]) <= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) <= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) <= 0.1)) ||
                 (($d12/($delta[0]+$delta[1]) <= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) <= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) >= 0.1)) ||
                ((($d12/($delta[0]+$delta[1]) <= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) >= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) >= 0.1)) ||
                 ((($d12/($delta[0]+$delta[1]) >= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) <= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) >= 0.1))){
                $w1 = 1;
             }

             if ((($d12/($delta[0]+$delta[1]) <= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) <= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) <= 0.1)) ||
                 (($d12/($delta[0]+$delta[1]) <= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) >= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) <= 0.1)) ||
                ((($d12/($delta[0]+$delta[1]) <= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) >= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) >= 0.1)) ||
                 ((($d12/($delta[0]+$delta[1]) >= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) >= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) <= 0.1))){
                $w2 = 1;
             }

             if ((($d12/($delta[0]+$delta[1]) <= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) <= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) <= 0.1)) ||
                 (($d12/($delta[0]+$delta[1]) >= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) <= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) <= 0.1)) ||
                ((($d12/($delta[0]+$delta[1]) >= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) <= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) >= 0.1)) ||
                 ((($d12/($delta[0]+$delta[1]) >= 0.1) &&
                  ($d13/($delta[0]+$delta[2]) >= 0.1) &&
                  ($d23/($delta[1]+$delta[2]) <= 0.1))){
                $w3 = 1;
             }
         }
     }
 }

 # determine precipt based on valid wires
PRECIPT: {
   ($w1==1 && $w2==1 && $w3==1) &&
         do{$pcpt=avg($delta[0],$delta[1],$delta[2]); last PRECIPT;};
   ($w1==1 && $w2==1 && $w3==0) &&
         do{$pcpt=avg($delta[0],$delta[1]);           last PRECIPT;};
   ($w1==1 && $w2==0 && $w3==1) &&
         do{$pcpt=avg($delta[0],$delta[2]);           last PRECIPT;};
   ($w1==1 && $w2==0 && $w3==0) &&
         do{$pcpt=($delta[0]);                        last PRECIPT;};
   ($w1==0 && $w2==1 && $w3==1) &&
         do{$pcpt=avg($delta[1],$delta[2]);           last PRECIPT;};
   ($w1==0 && $w2==1 && $w3==0) &&
         do{$pcpt=($delta[1]);                        last PRECIPT;};
```

```
          ($w1==0 && $w2==0 && $w3==1) &&
                do{$pcpt=($delta[2]);                          last PRECIPT;};
          ($w1==0 && $w2==0 && $w3==0) &&
                do{$pcpt=0;                                    last PRECIPT;};
      }


        $p[$i]=round($pcpt,1);

        # calculated precipt is the  sum of precipt in the last hour only
        if ($i >= ($n - $nphr)){$p_total += $pcpt;}

    }
    return $p_total;
}


sub min{
    my $min_so_far = shift @_;
    foreach (@_){
        if ($_ < $min_so_far){
            $min_so_far = $_;
        }
    }
    return $min_so_far;
}

sub avg{
    my $sum = 0;
    my $n = 0;

    foreach (@_){
        $sum += $_;
        $n++;
        }
    return ($n>0) ? $sum/$n : undef;
}

sub round{
    my $value = shift;
    my $decimals = shift;

    for (my $i = 0; $i<$decimals; $i++){$value *= 10;}
    $value = int($value + 0.5);
    for (my $i = 0; $i<$decimals; $i++){$value /= 10;}
        return $value;
}
```