Technical Note 18

Issued 4.10.05

CALCULATING SUNSHINE HOURS FROM PYRANOMETER / SOLARIMETER DATA

CONTENTS

1. INTRODUCTION

For many years sunshine hours have been recorded using manually operated meteorological stations. This measurement was originally developed as a means of recording integrated global solar radiation, before the advent of electronics. By using modern instruments and datalogging systems, there are now often more appropriate ways of recording solar radiation for scientific applications. However, because there is an archive of historical sunshine data, and also because sunshine hours are now commonly used as a measure which the public can relate to (normally in weather forecasts and tourist brochures), there is a requirement to continue making these measurements.

Traditionally this has been done using a Campbell-Stokes recorder, which consists of a glass sphere with a piece of card held at a defined distance behind the glass. The card is graduated with time intervals which match the movement of the image of the sun as it tracks across the sky. If the sun is strong enough then the focused beam carbonises the card leaving a trace. The length of the trace equates to sunshine duration. However, reading the cards always involves a degree of subjectivity and the design does not lend itself to automation.

A relatively recent formal definition of sunshine duration by the World Meteorological Organisation [1] has allowed the development of automatic instruments for measuring sunshine hours. This defines sunshine hours as 'the sum of the time intervals (in hours) during which the direct (normal) solar irradiance exceeds a threshold of 120Wm^{-2} '.

The most complex (and most accurate) of these instruments are tracking pyrheliometers, where a collimated sensor automatically moves to track the movement of the sun. This reads the direct beam radiation only. Any reading over 120Wm^{-2} is defined as being sunshine.

Other motorised sensors have been developed which, rather than tracking the sun, have quickly rotating shade bands. These sensors detect the direct beam component of radiation by measuring the maximum difference in measured radiation as the shade band rotates. However, as this type of sensor does not truly track the sun it requires regular adjustment to take into account the seasonal changes in solar declination.

Both of the above types of sensor are relatively expensive and have moving parts requiring extra power. An alternative technique which does not involve any moving parts is the combination of two pyranometers, one with a shadow ring (which limits detection to diffuse radiation) and one which is normally exposed to record total global radiation. The difference between these two measurements is the direct beam component. One complication with these sensors is that they are conventionally set up on a horizontal plane while the definition of 'sunshine' is for a surface normal to the sun's rays. As these sensors usually have a good 'cosine response', it is possible to correct these readings to give an equivalent normal reading if the sun's elevation angle is known. A Campbell Scientific datalogger can be programmed to estimate the elevation angle at any instant, where the latitude and longitude of the site are known (using virtually identical theory to that given below), thus enabling accurate estimates of sunshine hours. However, this type of sensor set-up is still relatively expensive and the shadow ring requires regular adjustment to compensate for seasonal changes in solar declination.

A much simpler approach is to try to estimate sunshine hours from the single pyranometer/ solarimeter of the type normally installed on most weather stations. As these sensors measure total global radiation the normal definition of ‘sunshine’ cannot be used. Simple fixed thresholds, as often used in low grade weather stations, do not give reliable answers either, as diffuse radiation from a completely cloudy sky in the summer will often exceed direct beam radiation in the winter.

An alternative algorithm has recently been suggested by workers at the Royal Dutch Meteorological Institute (KNMI)[2]. They have proposed and tested an algorithm which defines sunshine as being when the measured global radiation (S) is greater than 0.4 times the potential solar radiation outside the earth’s atmosphere on a horizontal surface (S_o). One long term test of this algorithm showed that estimates of sunshine hours were on average within 0.9 hours of the daily total.

While this might appear to give rather poor accuracy compared to that one would expect for totalised solar radiation, they consider it accurate enough for normal non-scientific use of sunshine hour data. As the sensor can also be used to make accurate solar radiation measurements, scientific data can be collected at the same time.

The remainder of this Technical Note describes how such an algorithm might be programmed into a Campbell Scientific datalogger.

2. ESTIMATING POTENTIAL SOLAR RADIATION

The solar radiation outside the earth’s atmosphere is well defined and is known as the solar constant. Although this varies slightly during the year an accepted average value is 1373Wm^{-2}. This value is constant for a surface normal to the sun’s rays, as opposed to the horizontal exposure of a solarimeter. The potential solar radiation (S_o) on a horizontal surface outside the earth’s atmosphere is calculated in Wm^{-2} from:

$$S_o = 1373 \sin \phi \qquad (1)$$

where ϕ is the elevation angle of the sun. Sin ϕ is computed from

$$\sin \phi = \sin d \sin l + \cos d \cos l \ \cos [15(t-t_o)] \qquad (2)$$

where d is the solar declination angle, l is the latitude of the site, t is clock time and t_o is the time of solar noon. The declination angle is often evaluated using several terms of a Fourier series, but, since Campbell Scientific dataloggers are particularly adept at evaluating polynomials, it is better to approximate sin d using a polynomial:

$$\sin d = -0.37726-0.10564j+1.2458j^2 -0.75478j^3+0.13627j^4-00572j^5 \qquad (3)$$

where j is day of the year/100. As the dataloggers (with the exception of the CR9000) do not include a cosine function the cosine is computed from the trigonometric identity:

$$\cos d = (1 - \sin^2 d)^{1/2} \qquad (4)$$

For running the sunshine hours algorithm, it is assumed that the user always sets the clock to standard time (not daylight saving time). The time, t, needed for eq. 2 is therefore just the datalogger clock time. The time of solar noon is given by:

$$t_o = 12 - L_c - E_t \text{ (hr)} \qquad (5)$$

where L_c is a longitude correction and E_t is the ‘Equation of Time’. The longitude correction is a user-supplied parameter. It is calculated by determining the difference between the longitude of the site and the longitude of the standard meridian. Standard meridians are at 0, 15, 30..345 degrees. Generally time zones run approximately +7.5 to -7.5 degrees on either side of a standard meridian, but this varies depending on political boundaries. The user should check an atlas to get both the longitude and the standard meridian for the site (as well as the latitude, which is also needed for eq. 2). The longitude correction is computed from:

$$L_c = (L_s - L)/15 \qquad (6)$$

If the longitude of the site were L=117 degrees, and the longitude of the standard meridian were L_s=120 degrees, then L_c would be (120-117)/15 = 0.2 hr. If the longitude of the site were 123 degrees, then L_c would be -0.2 hour.

The Equation of Time is an additional correction to the time of solar noon that depends on day of the year. Again, a polynomial is used for the computation. Two equations are used, one for the first half of the year, and one for the second. For the first half,

$$E_t = -0.04056-0.74503j+0.08823j^2+2.0516j^3 -1.8111j^4+0.42832j^5, \qquad (7)$$

where j = day of the year/100 (as defined above).

For the second half (day of the year>180),

$$E_t = -0.05039-0.33954j_2 +0.04084j_2^2+1.8928j_2^3 -1.7619j_2^4+0.4224j_2^5, \qquad (8)$$

where j_2 = (day of the year-180)/100.

3. TOTALISING SUNSHINE HOURS

Using the above equation S_o can be calculated at any instant. The program example that follows shows how these equations can be entered in a datalogger program in subroutine 1. You must know the longitude and latitude of your installation and calculate the latitude correction manually for entry into the program.

To totalise sunshine hours the program compares 0.4 times the current solar radiation with the current value of S_o. If the value is higher than S_o then a constant (equivalent to the scan interval in hours) is added to the daily total. This program also calculates the more scientifically valid integrated solar radiation. (See program instructions in Table 1, entries 14 and 15.)

One of the limitations in the above theory is that the threshold figure of 0.4 * S_o can become very small at low sun angles when the radiation is low anyway. However, most radiation sensors used do not have a perfect 'cosine response' to variations in sun angle especially at low elevation angles and they can also suffer from zero offsets (especially when temperature is changing rapidly as it can do at dawn and dusk). Furthermore errors in levelling the sensor can cause proportionally large errors in the estimate of solar radiation at low angles. This makes this technique subject to large errors at low solar elevations. As direct beam radiation is attenuated greatly at these angles other standard techniques of measuring sunshine duration would normally not record 'sunshine' anyway, so this technique will often lead to an overestimation of sunshine hours.

To overcome this problem an additional refinement can be made to ignore 'sunshine' when the elevation is very low. In the program below, this is incorporated by only adding to the total of sunshine hours if sin ϕ is greater than, say, 0.1, which equates to an elevation greater than six degrees. The exact cut-off point is, however, a subjective decision. For instance, you may choose to ignore readings if the site layout prevents the sensor 'seeing' the direct beam at low elevations.

WARNING: Campbell Scientific recommends the use of a high quality sun screen lotion when exposing your skin to solar radiation for large values of sunshine hours!

4. REFERENCES

[1] WMO, 1986: *Revised instruction manual on radiation instruments and measurements* WMO/TD - No, 149, ed. C. Frohlich and J. London (World Climate Research Programme publications series 7) WMO - Geneva Switzerland.

[2] Slob, W.H. and W.A.A. Monna 1991: *Bepaling van directe en diffuse straling en van zonneschijndurr uit 10-minuutwaarden van globale straling*. KNMI TR-136 (FM), Koninklijk Nederlands Meteorologisch Instituut - De Bilt/The Netherlands.

Acknowledgements
Some of the theory and text used in this Technical Note was produced by G.S. Campbell, Dept. of Crop and Soil Sciences, Washington State University.

5. EXAMPLE PROGRAM

```
Program: Example sunshine hours program
Flag Usage: None
Input Channel Usage: 1 (SE) for pyranometer
Excitation Channel Usage: None
Control Port Usage: None
Pulse Input Channel Usage: None
Output Array Definitions:
Array 110 gives Day,Hrs-Min,Total sunshine hrs,
Total solar radiation (MJ m-2)
Sunshine hours so far today stored in location 3

*     1          Table 1 Programs
  01:  60        Sec. Execution Interval
```

Every minute

```
 01: P1          Volt (SE)
  01:  1         Rep
  02:  35        2500 mV 50Hz rejection Range
  03:  1         IN Chan
  04:  1         Loc [:W m-2 ]
  05:  100       Mult
  06:  0.0000    Offset
```

Example measurement of a pyranometer giving 10mV at 1000 Wm^{-2}

```
 02: P86         Do
  01:  01        Call Subroutine 1
```

Call sub 1 to calculate S_o and the sunshine threshold

```
 03: P88         If X<=>Y
  01:  1         X Loc W m-2
  02:  3         >=
  03:  2         Y Loc Threshold
  04:  30        Then Do
```

If the solar radiation is greater than the threshold

```
 04: P89         If X<=>F
  01:  51        X Loc sin(l)
  02:  3         >=
  03:  0.1       F
  04:  30        Then Do
```

AND sin(l) is >=0.1 (elevation angle > 6 degrees)

```
05:  P34         Z=X+F
 01:   3         X Loc Sun hrs
 02:   .01667    F
 03:   3         Z Loc [:Sun hrs ]
```

Increment today's sunshine hours

```
06: P95        End

07: P95        End

08: P92        If time is
 01:  0        minutes into a
 02:  1440     minute interval
 03:  10       Set high Flag 0 (output)
```

At midnight set the output flag

```
09: P77        Real Time
 01:  120      Day,Hour-Minute
```

Store the time

```
10: P70        Sample
 01:  1        Reps
 02:  3        Loc Sun hrs
```

Store sunshine total for previous day

Then reset the total

```
11: P91        If Flag/Port
 01:  10       Do if flag 0 (output) is high
 02:  30       Then Do
```

Set above at midnight only

```
12: P30        Z=F
 01:  0.0000   F
 02:  00       Exponent of 10
 03:  3        Z Loc [:Sun hrs ]
```

Reset sunshine hours to zero

```
13: P95        End

14: P37        Z=X*F
 01:  1        X Loc W m-2
 02:  0.06     F 60 sec / 1000
 03:  5        Z Loc [:MJ m-2 ]
```

Convert S into MJ m^{-2} per scan interval

```
15: P72        Totalize
 01:  1        Rep
 02:  5        Loc MJ m-2
```

Totalise radiation as MJ m^{-2}

```
16:   P        End Table 1

*     2        Table 2 Programs
 01:  0.0000   Sec. Execution Interval

01:   P        End Table 2

*     3        Table 3 Subroutines

01: P85        Beginning of Subroutine
 01:  1        Subroutine Number
```

Subroutine which calculates S_o and threshold radiation for sunshine

```
02:  P30       Z=F
  01:  54      F
  02:  0       Exponent of 10
  03:  47      Z Loc [:latitude ] !!! User entry
```

User entered latitude

```
03:  P30       Z=F
  01:  0       F
  02:  0       Exponent of 10
  03:  48      Z Loc [:lngt. cor] !!! User entry
```

User entered longitude correction

```
04:  P18       Time
  01:  2       Hours into current year (max. 8784)
  02:  0       Mod/by
  03:  42      Loc [:clndr day]

05:   P37      Z=X*F
  01:  42      X Loc clndr day
  02:  .04167  F
  03:  42      Z Loc [:clndr day]
```

Convert hours into Julian day

```
06:  P37       Z=X*F
  01:  42      X Loc clndr day
  02:  .01     F
  03:  41      Z Loc [:day/100 ]
```

Scale days for polynomial

```
 07: P55       Polynomial
  01:  1       Rep
  02:  41      X Loc day/100
  03:  43      F(X) Loc [:sin(d) ]
  04:  -.37726 C0
  05:  -.10564 C1
  06:  1.2458  C2
  07:  -.75478 C3
  08:  .13627  C4
  09:  -.00572 C5
```

Calculate sin(d) using polynomial approximation

```
08:  P89       If X<=>F
  01:  42      X Loc clndr day
  02:  3       >=
  03:  180     F
  04:  30      Then Do

09:  P34       Z=X+F
  01:  41      X Loc day/100
  02:  -1.8    F
  03:  40      Z Loc [:eq of tim]
```

Equation of time polynomial for the 2nd half of year

```
10:  P55       Polynomial
  01:  1       Rep
  02:  40      X Loc eq of tim
  03:  40      F(X) Loc [:eq of tim]
  04:  -.05039 C0
  05:  -.33954 C1
  06: .04084   C2
  07:  1.8928  C3
  08: -1.7619  C4
  09:  .4224   C5

11:  P94       Else
```

Equation of time for the first half of the year

```
12:  P55       Polynomial
  01:  1       Rep
  02:  41      X Loc day/100
  03:  40      F(X) Loc [:eq of tim]
  04:  -.04056 C0
  05:  -.74503 C1
  06:  .08823  C2
  07:  2.0516  C3
  08:  -1.8111 C4
  09:  .42832  C5

13:  P95       End

14:  P36       Z=X*Y
  01:  43      X Loc sin(d)
  02:  43      Y Loc sin(d)
  03:  44      Z Loc [:cos(d) ]

15:  P37       Z=X*F
  01:  44      X Loc cos(d)
  02:  -1      F
  03:  44      Z Loc [:cos(d) ]

16:  P32       Z=Z+1
  01:  44      Z Loc [:cos(d) ]

17:  P39       Z=SQRT(X)
  01:  44      X Loc cos(d)
  02:  44      Z Loc [:cos(d) ]
```

*Above calcs cos(d) = SQRT(1-sin(d)*sin(d))*

```
18:  P48       Z=SIN(X)
  01:  47      X Loc latitude
  02:  45      Z Loc [:sinl*sind]
```

sine of latitude

```
19:  P36       Z=X*Y
  01:  43      X Loc sin(d)
  02:  45      Y Loc sinl*sind
  03:  45      Z Loc [:sinl*sind]

20:  P34       Z=X+F
  01:  47      X Loc latitude
  02:  90      F
  03:  46      Z Loc [:cosl*cosd]

21:  P48       Z=SIN(X)
  01:  46      X Loc cosl*cosd
  02:  46      Z Loc [:cosl*cosd]
```

Estimates cos(l) = sin (l+90)

```
22:  P36       Z=X*Y
  01:  44      X Loc cos(d)
  02:  46      Y Loc cosl*cosd
  03:  46      Z Loc [:cosl*cosd]

23:  P18       Time
  01:  1       Minutes into current day (max.1440)
  02:  0       Mod/by
  03:  49      Loc [:t ]

24:  P37       Z=X*F
  01:  49      X Loc t
  02:  .01667  F
  03:  49      Z Loc [:t ]
```

Estimate time of day as a decimal number of hours

```
25:  P34       Z=X+F
  01:  49      X Loc t
  02:  -12     F
  03:  50      Z Loc [:t-to ]
```

deduct solar noon

```
26:  P33       Z=X+Y
  01:  50      X Loc t-to
  02:  48      Y Loc lngt. cor
  03:  50      Z Loc [:t-to ]
```

add latitude correction

```
27:  P33       Z=X+Y
  01:  50      X Loc t-to
  02:  40      Y Loc eq of tim
  03:  50      Z Loc [:t-to ]
```

add equation of time

```
28:  P37       Z=X*F
  01:  50      X Loc t-to
  02:  15      F
  03:  51      Z Loc [:sin(l) ]
```

convert to degrees

```
29:  P34       Z=X+F
  01:  51      X Loc sin(l)
  02:  90      F
  03:  51      Z Loc [:sin(l) ]
```

Add 90 degrees to allow calculations of cos using cos(x) = sin (x+90)

```
30:  P48       Z=SIN(X)
  01:  51      X Loc sin(l)
  02:  51      Z Loc [:sin(l) ]
```

```
31:  P36        Z=X*Y
  01:  51       X Loc sin(l)
  02:  46       Y Loc cosl*cosd
  03:  51       Z Loc [:sin(l) ]

32:  P33        Z=X+Y
  01:  51       X Loc sin(l)
  02:  45       Y Loc sinl*sind
  03:  51       Z Loc [:sin(l) ]
```

above calculates
*sin(l)=sin(d)*sin(l)+cos(d)*cos(l)*cos(15(t-to))*

```
33:  P89        If X<=>F
  01:  51       X Loc sin(l)
  02:  4        <
  03:  0        F
  04:  30       Then Do
```

If sin(l) < 0 then set to zero

```
34:  P30        Z=F
  01:  0        F
  02:  0        Exponent of 10
  03:  51       Z Loc [:sin(l) ]

35:  P95        End

36:  P37        Z=X*F
  01:  51       X Loc sin(l)
  02:  1373     F
  03:  4        Z Loc [:So W m-2 ]
```

Calculate S_o

```
37: P37         Z=X*F
  01:  4        X Loc So W m-2
  02:  0.4      F
  03:  2 Z      Loc [:Threshold]
```

Calculate the threshold for sunshine

```
38:  P95        End

39:  P          End Table 3

*    A Mode     10 Memory Allocation


  01:  100      Input Locations
```

Input locations increased to allow extra workspace

```
02:  64         Intermediate Locations
  03:  0.0000   Final Storage Area 2

* C Mode        12 Security
     01:        0000 LOCK 1
     02:        0000 LOCK 2
     03:        0000 LOCK 3
```

Input Location Assignments (with comments):

```
Key:
T=Table Number
E=Entry Number
L=Location Number

T: E: L:
1: 1: 1: Loc [:W m-2 ]
3: 37: 2: Z Loc [:Threshold]
1: 5: 3: Z Loc [:Sun hrs ]
1: 12: 3: Z Loc [:Sun hrs ]
3: 36: 4: Z Loc [:So W m-2 ]
1: 14: 5: Z Loc [:MJ m-2 ]
3: 9: 40: Z Loc [:eq of tim]
3: 10: 40: F(X) Loc [:eq of tim]
3: 12: 40: F(X) Loc [:eq of tim]
3: 6: 41: Z Loc [:day/100 ]
3: 4: 42: Loc [:clndr day]
3: 5: 42: Z Loc [:clndr day]
3: 7: 43: F(X) Loc [:sin(d) ]
3: 14: 44: Z Loc [:cos(d) ]
3: 15: 44: Z Loc [:cos(d) ]
3: 16: 44: Z Loc [:cos(d) ]
3: 17: 44: Z Loc [:cos(d) ]
3: 18: 45: Z Loc [:sinl*sind]
3: 19: 45: Z Loc [:sinl*sind]
3: 20: 46: Z Loc [:cosl*cosd]
3: 21: 46: Z Loc [:cosl*cosd]
3: 22: 46: Z Loc [:cosl*cosd]
3: 2: 47: Z Loc [:latitude ] !!! User entry
3: 3: 48: Z Loc [:lngt. cor] !!! User entry
3: 23: 49: Loc [:t ]
3: 24: 49: Z Loc [:t ]
3: 25: 50: Z Loc [:t-to ]
3: 26: 50: Z Loc [:t-to ]
3: 27: 50: Z Loc [:t-to ]
3: 28: 51: Z Loc [:sin(l) ]
3: 29: 51: Z Loc [:sin(l) ]
3: 30: 51: Z Loc [:sin(l) ]
3: 31: 51: Z Loc [:sin(l) ]
3: 32: 51: Z Loc [:sin(l) ]
3: 34: 51: Z Loc [:sin(l) ]
```

Input Location Labels:

1:W m-2	18:_______	35:_______	52:_______
2:Threshold	19:_______	36:_______	53:_______
3:Sun hrs	20:_______	37:_______	54:_______
4:So W m-2	21:_______	38:_______	55:_______
5:MJ m-2	22:_______	39:_______	56:_______
6:_______	23:_______	40:eq of tim	57:_______
7:_______	24:_______	41:day/100	58:_______
8:_______	25:_______	42:clndr day	59:_______
9:_______	26:_______	43:sin(d)	60:_______
10:_______	27:_______	44:cos(d)	61:_______
11:_______	28:_______	45:sinl*sind	62:_______
12:_______	29:_______	46:cosl*cosd	63:_______
13:_______	30:_______	47:latitude	64:_______
14:_______	31:_______	48:lngt. cor	65:_______
15:_______	32:_______	49:t	66:_______
16:_______	33:_______	50:t-to	67:_______
17:_______	34:_______	51:sin(l)	68:_______