

## **NOTE P: ALTERNATE ALGORITHMS**

### **WELFORD'S FOR VARIANCE AND STANDARD DEVIATION**

Knuth (1998) Vol. 2, page 232 recommends Welford's algorithm (Welford 1962).

In Knuth's notation it is:

Initialize

$$M_1 = X_1$$

$$S_1 = 0$$

Loop over the data set

$$M_k = M_{k-1} + (X_k - M_{k-1}) / k$$

$$S_k = S_{k-1} + (X_k - M_{k-1}) * (X_k - M_k)$$

One form of the algorithm is given below:

Data X(1 to N)

A = X(1)

B = 0

FOR K = 2 to N

U = X(K) - A

A = A + U / CDBL(K)

B = B + U \* (X(K) - A)

NEXT K

AVERAGE = A

STDEV = SQR( B / CDBL(N - 1) )

Welford's algorithm is an exact solution, not an approximation.

Welford's algorithm however cannot be directly adapted to the specific autocorrelation function given by NIST or to covariances.

### **KAHAN'S ADDITION METHOD**

Knuth (1998) Vol. 2, page 244, exercise 19, shows that Kahan's method improves Welford's accuracy. However it may improve other addition methods. It is only effective when all the signs of the values to be summed are +.

In Knuth's notation it is:

Initialize

$$S_0 = 0$$

$$C_0 = 0$$

Loop over the data set

$$Y_k = X_k - C_{k-1}$$

$$S_k = S_{k-1} + Y_k$$

$$C_k = (S_k - S_{k-1}) - Y_k$$

$S_k$  is the resulting sum.

A vba sub would be:

Sub KAHAN( KIN, CIN, SIN, XIN ) 'XIN are the terms, SIN is the in/out sum

Dim y, x, c, s as double

If KIN = 0 then

$$SIN = XIN$$

$$CIN = 0$$

Else

$$Y = XIN - CIN$$

$$SOUT = SIN + Y$$

$$CIN = (SOUT - SIN) - Y$$

$$SIN = SOUT$$

End If

End Sub

## COMBINED ALGORITHM

The series of summations for variables A and B will introduce errors. An improvement in Welford's algorithm would be to include Kahan's algorithm to correct for summation errors.

Data X(1 to N)

$$A = X(1)$$

$$B = 0$$

FOR K = 2 to N

$$U = X(K) - A$$

$$A = A + U / CDBL(K)$$

$$B = B + U * (X(K) - A)$$

NEXT K

$$AVERAGE = A$$

$$STDEV = SQR( B / CDBL(N - 1) )$$

There are other forms with different summation control methods given in the ACM past publications. Which one is best, is a matter of opinion.

Note P shows the results of comparisons to the VAR and STDEV functions.

This would add to the code and slightly reduce the speed of execution. The improvement in accuracy is shown above for the uniform distribution

### **AUTOCORRELATION():**

This is the autocorrelation equation listed in NIST sets or uni-variate data. By adding this function, Excel can correctly reduce the NIST uni-variate data sets. This removes a current problem with all versions of Excel.

Welford's algorithm and Kahan's error correction are combined here into one routine. All variables except i and nvals are double. Variables cc1-cc5 hold the summation errors. Tv is a dummy variable. Nvals is the length of the range in the calling list.

```
mx0 = ax(1, 1)
```

```
my0 = ay(1, 1)
```

```
sx0 = 0#
```

```
sy0 = 0#
```

```
sxy0 = 0#
```

```
cc1 = 0#
```

```
cc2 = 0#
```

```
cc3 = 0#
```

```
cc4 = 0#
```

```
cc5 = 0#
```

```
For i = 2 To nvals
```

```
    c = CDBl(i)
```

```
    If method = 1 Then
```

```
        x = ax(i, 1)
```

```
        y = ay(i, 1)
```

```
    Else
```

```

        x = ax(1, i)
        y = ay(1, i)
    End If
    tv = ((x - mx0) / c) - cc1
    mx1 = mx0 + tv
    cc1 = (mx1 - mx0) - tv
    tv = (x - mx0) * (x - mx1) - cc2
    sx1 = sx0 + tv
    cc2 = (sx1 - sx0) - tv
    tv = ((y - my0) / c) - cc3
    my1 = my0 + tv
    cc3 = (my1 - my0) - tv
    tv = (y - my0) * (y - my1) - cc4
    sy1 = sy0 + tv
    cc4 = (sy1 - sy0) - tv
    tv = (x - mx1) * (y - my0) - cc5
    sxy1 = sxy0 + tv
    cc5 = (sxy1 - sxy0) - tv
    mx0 = mx1
    sx0 = sx1
    my0 = my1
    sy0 = sy1
    sxy0 = sxy1
Next i
NewCorrel = sxy1 / Sqr(sx1 * sy1)

```

Although it appears to hold a lot of potential, it does not have the accuracy of routines that first center and then do regular differences and sums of differences squared.

Kahan's error correction could improve the accuracy of long sums, providing they are all of numbers having the same sign (See Knuth).