JOURNAL OF COMPUTATIONAL PHYSICS 85, 500–501 (1989)

Note

Updating Means and Variances

In [1], a method is given for updating means, variances, and cross-products of pairs of variables, when more data become available. There is a little-known variation of their formulae which is faster by avoiding the use of $i/(i+1)$ terms.

Using almost the same notation as in [1], if we define the sample mean and variance of the first i observations, $x_1, x_2, ..., x_i$ as

$$m_i = \sum_{r=1}^{i} x_r/i \tag{1}$$

$$s_i^2 = \sum_{r=1}^{i} (x_r - m_i)^2/(i-1). \tag{2}$$

The modified update formulae, which can easily be obtained from those given in [1], are

$$\delta = x_{i+1} - m_i \tag{3}$$

$$m_{i+1} = m_i + \delta/(i+1) \tag{4}$$

$$sx_{i+1} = sx_i + \delta \cdot (x_{i+1} - m_{i+1}), \tag{5}$$

where sx_i is the notation used in [1] for the sum of squares of deviations from the mean of the first i observations. The new sample variance, s_{i+1}^2, is obtained then by dividing sx_{i+1} by i.

This can be extended further to the case of weighted observations by simply replacing the $\delta/(i+1)$ in (4) with $w_{i+1} \cdot \delta/sw_{i+1}$, where w_r is the weight to be given to the rth case and sw_i is the sum of the first i weights. The term added on the right-hand side of (5) needs to be multiplied by the weight of the new case.

A review of some of the algorithms for updating sample variances is given in [2], although the variation above is not mentioned. The authors of [2] give their own algorithm which is based upon the binary representation of the sample size, i, which gives good accuracy when i is equal to a power of 2. Their algorithm requires very lengthy code. The present author learned of the above method from Eq. (16) on page 64 of [3].

The method above applies to updating sums of cross-products of variables, as well as to sums of squares, and is particularly efficient for the updating of matrices of sums of cross-products. A Fortran subroutine for performing such updates follows. Notice that the array *wk* is used to store the deviations from the updated

0021-9991/89 $3.00

mean while *dev* holds the deviation from the old mean. Only the lower triangle of array *sxx* is used. Users of other languages, such as BASIC, should have no difficulty in translating it.

```
      subroutine upsxy(x, k, y, n, xmean, ymean, sxx, sxy, syy, wk)
c
c     Progressively update means and sums of cross-products.
c
      integer k, n
      real x(k), y, xmean(k), ymean, sxx(k, k), sxy(k), syy, wk(k)
c
c     Local variables
c
      integer i, j
      real dev
c
c     dev = deviation of x- or y-value from the old mean.
c     wk(i) = deviation of x(i) from its updated mean.
c
      n = n + 1
      do 20 i = 1, k
        dev = x(i) - xmean(i)
        xmean(i) = xmean(i) + dev/n
        wk(i) = x(i) - xmean(i)
        do 10 j = 1, i
   10   sxx(i, j) = sxx(i, j) + dev * wk(j)
   20 continue
c
      dev = y - ymean
      ymean = ymean + dev/n
      do 30 i = 1, k
   30 sxy(i) = sxy(i) + dev * wk(i)
      syy = syy + dev * (y - ymean)
c
      return
      end
```

References

1. A. D. Booth and I. J. M. Booth, *J. Comput. Phys.* **77**, 537 (1988).
2. T. F. Chan, G. H. Golub, and R. J. LeVeque, *Amer. Statist.* **37**, 242 (1983).
3. R. I. Jennrich, in *Statistical Methods for Digital Computers*, edited by K. Enslein *et al.* (Wiley, New York, 1977), p. 58.

Received: October 11, 1988; revised: January 30, 1989

Alan J. Miller

CSIRO Division of Mathematics & Statistics, Private Bag 10, Clayton, Victoria 3168, Australia