Appendix 6: Sorting and Searching

Sorting and searching have a significant impact on the performance of a number of procedures. For those procedures, the methods used are identified here.

CROSSTABS

In the general mode, the table of cells is searched using an unordered open scatter table search and insertion algorithm similar to Knuth's Algorithm *L* (Knuth, 1973, p. 518). The scatter table contains only pointers to the actual cell contents and is twice as large as it need be (that is, if there is room for *m* cells, the scatter table has room for 2*m* pointers). This means it can never be more than half full. Collisions are resolved by sequential search from the initial location until an empty pointer is found.

The hash function used is given by the following algorithm:

```
Let
    $k$ be the table number
    $p$ be the dimension of the table
    $v_i (i = 1, \ldots, p)$ be the bit string used to represent the value of the $i$th variable
defining table $k$
    $m$ be the length of the scatter table
    $n$ be the resulting hash value, to be used as an index in the scatter table
    $j := k$
    for $i := 1$ to $p$
    $j := j$ rotated left 3 bits
    $j := j$ EXCLUSIVE OR $v_i$
    end
    $n := (j \text{ modulo } m) + 1$
```

When the tables have been completed, the cells are sorted by table numbers and the values of the defining variables using the algorithm described by Singleton (1969).

FREQUENCIES

FREQUENCIES uses the same search and sort algorithms as CROSSTABS, except that its hashing function is given by:

$$h = \left(\left(\left(k + 16807v\right)\text{modulo } 2^{31}\right)\text{modulo } m\right) + 1$$

where
h is the hash value, to be used as an index in the scatter table
k is the table number
v is the integer value of the bits representing the value to be tabulated
m is the length of the scatter table

NONPAR CORR and NPAR TESTS

Both use the method of Singleton to sort cases for computing ranks.

SURVIVAL

SURVIVAL uses a modified Quicksort similar to Knuth's algorithm *Q* (Knuth, 1973, p. 116) to sort cases.

References

Knuth, D. E. 1973. *The Art of Computer Programming*, volume 3: Sorting and Searching. Reading, Mass.: Addison-Wesley.

Singleton, R. C. 1969. Algorithm 347: Efficient sorting in minimal storage. *Communications of the ACM*, 12: 185–187.