Citation for published version

DOI

Link to record in KAR
http://kar.kent.ac.uk/21160/

Document Version
UNSPECIFIED

Copyright & reuse
Content in the Kent Academic Repository is made available for research purposes. Unless otherwise stated all content is protected by copyright and in the absence of an open licence (eg Creative Commons), permissions for further reuse of content should be sought from the publisher, author or other copyright holder.

Versions of research
The version in the Kent Academic Repository may differ from the final published version. Users are advised to check http://kar.kent.ac.uk for the status of the paper. Users should always cite the published version of record.

Enquiries
For any further enquiries regarding the licence status of this document, please contact: researchsupport@kent.ac.uk
If you believe this document infringes copyright then please contact the KAR admin team with the take-down information provided at http://kar.kent.ac.uk/contact.html
A Remark on Algorithm AS57 – Printing Multidimensional Tables
By Tim Hopkins and David Morse
Computing Laboratory, University of Kent,
Canterbury, Kent CT2 7NF, UK.

Keywords: Table; Printing; Multidimensional.

LANGUAGE
Fortran 77.

DESCRIPTION AND PURPOSE
This routine is a Fortran 77 version of that appearing in Griffiths and Hill (1985)
(originally Haberman (1973)). It uses character variables rather than storing
corresponding information in integers and reals which was the only way of implementing
such an algorithm in standard Fortran 66. Where possible data statements
have been replaced by parameter statements. The code has been restructured
and variable names have been altered to avoid clashes with Fortran 77 intrinsic
function names and the Fortran specifier name UNIT.

The algorithm prints one or more NVAR-dimensional parallel tables stored
in an array TABLE of length NTAB. For example, TABLE might contain
a table of observations, a table of fitted values and a table of residuals. To be
specific, suppose that a three-dimensional table \{n_{ijk}\} has been studied, where
1 \leq i \leq 4, 1 \leq j \leq 3, 1 \leq k \leq 2. From this investigation, a fitted table \{m_{ijk}\}
and a residual table \{r_{ijk}\} have been derived. For purposes of display it is
desired that the printed table have the format shown in Figure 1. In this case,
TABLE is divided into three subtables as follows.

The subtable corresponding to the observations \{n_{ijk}\} begins at TABLE(1),
the subtable for the fit \{m_{ijk}\} begins at TABLE(4 \times 3 \times 2 + 1 = 25) and the
subtable of residuals \{r_{ijk}\} begins at TABLE(49). Each subtable is arranged
in standard Fortran fashion; that is, TABLE(1) = n_{111}, TABLE(2) = n_{112},
TABLE(3) = n_{211}, etc. (Note: that standard Fortran 77 would allow TABLE
in the calling routine to be a four-dimensional array dimensioned (4,3,2,3) where
the last dimension denotes the number of tables.)

To permit proper printing of the table, the algorithm requires descriptive
information concerning the structure of the table, and information which indicates
how the table should be displayed. The basic structural information is
provided by an integer array DIM of length NVAR and an integer array LOC
of length COL. The array DIM gives the number of categories in each of the
Figure 1: Sample test.

<table>
<thead>
<tr>
<th>Var 3</th>
<th>Var 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cat. 31</td>
<td>Cat. 21</td>
</tr>
<tr>
<td>Res.</td>
<td>$n_{111}$</td>
</tr>
<tr>
<td>Fit.</td>
<td>$m_{111}$</td>
</tr>
<tr>
<td>Obs.</td>
<td>$r_{111}$</td>
</tr>
<tr>
<td>Cat. 31</td>
<td>Cat. 22</td>
</tr>
<tr>
<td>Res.</td>
<td>$n_{121}$</td>
</tr>
<tr>
<td>Fit.</td>
<td>$m_{121}$</td>
</tr>
<tr>
<td>Obs.</td>
<td>$r_{121}$</td>
</tr>
<tr>
<td>Cat. 31</td>
<td>Cat. 23</td>
</tr>
<tr>
<td>Res.</td>
<td>$n_{131}$</td>
</tr>
<tr>
<td>Fit.</td>
<td>$m_{131}$</td>
</tr>
<tr>
<td>Obs.</td>
<td>$r_{131}$</td>
</tr>
<tr>
<td>Cat. 32</td>
<td>Cat. 21</td>
</tr>
<tr>
<td>Res.</td>
<td>$n_{211}$</td>
</tr>
<tr>
<td>Fit.</td>
<td>$m_{211}$</td>
</tr>
<tr>
<td>Obs.</td>
<td>$r_{211}$</td>
</tr>
<tr>
<td>Cat. 32</td>
<td>Cat. 22</td>
</tr>
<tr>
<td>Res.</td>
<td>$n_{221}$</td>
</tr>
<tr>
<td>Fit.</td>
<td>$m_{221}$</td>
</tr>
<tr>
<td>Obs.</td>
<td>$r_{221}$</td>
</tr>
<tr>
<td>Cat. 32</td>
<td>Cat. 23</td>
</tr>
<tr>
<td>Res.</td>
<td>$n_{231}$</td>
</tr>
<tr>
<td>Fit.</td>
<td>$m_{231}$</td>
</tr>
<tr>
<td>Obs.</td>
<td>$r_{231}$</td>
</tr>
</tbody>
</table>

Table variables. Thus $DIM(1) = 4$, $DIM(2) = 3$ and $DIM(3) = 2$ in the example. The array $LOC$ gives the location in $TABLE$ of each of the $COL$ parallel tables. In the example, $COL = 3$, $LOC(1) = 1$, $LOC(2) = 25$ and $LOC(3) = 49$.

Label information is required for the table variables, the categories of each variable and the names of the parallel tables. In addition, the title for the complete display is needed. The character arrays $TITLE$, $VARNAM$, $CATNAM$ and $COLLAB$ provide these data. All label arrays are declared to be assumed size character arrays; the variables $TITLEN$ and $LABLEN$ control how much of each label is actually printed as well as the field width used for output of the labels. Thus the first $TITLEN$ characters of each of the $NT2$ lines of the title are output and the maximum length of any label is $LABLEN$ characters. The integer constant $GAP$ (see Adjustable constants below) controls the amount of space inserted between a label and the following data values. Of the $NVAR$ labels of $VARNAM$, the last $VERT$ of them are printed on the left-hand side of the page, while $NVAR - VERT$ labels are printed along the top. In the example, $VERT = 2$.

The body of the table is printed by use of $F$-conversion. The format for an entry from parallel table $J$ is $Fa.b$, where $a$ is $LABLEN + GAP$ and $b$ is $DEC(J)$. The integer array $DEC$ has length $COL$. 

2
When many tables are to be printed, it is sometimes useful to print several small tables on the same page. To permit this practice, the user may use the value of the line counter \texttt{LINE} as returned by a prior call to \texttt{TABWRIT} along with the argument \texttt{RESTOR}. If \texttt{RESTOR} is set \texttt{.FALSE.}, the table will be printed, space permitting, within the remainder of the page. If \texttt{RESTOR} is set \texttt{.TRUE.}, the new table will be printed at the top of a new page.
STRUCTURE

SUBROUTINE TABWRIT(TITLE, TITLEN, NT2, TABLE, NTAB, DIM, NVAR, + LOC, COL, DEC, VARNAM, LABELN, CATNAM, MAXCAT, COLLAB, + VERT, RESTOR, LINE, SKIP, PAGE, WIDTH, UNIT, IFAULT)

Formal parameters

TITLE Character(*)(*) array (NT2) input: the title of the table.

TITLEN Integer input: the number of characters per line of the title.

NT2 Integer input: the number of lines in the title.

TABLE Real array (NTAB) input: the tables to be printed.

NTAB Integer input: the number of elements in TABLE.

DIM Integer array (NVAR) input: the number of categories in each variable of the table.

NVAR Integer input: the number of variables (dimensions) in the table.

LOC Integer array (COL) input: the locations in TABLE of the subtables to be printed.

COL Integer input: the number of subtables to be printed.

DEC Integer array (COL) input: the number of places to the right of the decimal point for each subtable.

VARNAM Character(*)(*) array (NVAR) input: the variable names.
\textbf{LABELN} \hspace{1em} \text{Integer} \hspace{1em} \text{input:} \hspace{1em} \text{the maximum number of characters of each label to be printed.}

\textbf{CATNAM} \hspace{1em} \text{Character*\{\} array (\textit{MAXCAT})} \hspace{1em} \text{input:} \hspace{1em} \text{the names of the variable categories.}

\textbf{MAXCAT} \hspace{1em} \text{Integer} \hspace{1em} \text{input:} \hspace{1em} \text{the maximum number of categories per variable.}

\textbf{COLLAB} \hspace{1em} \text{Character*\{\} array (\textit{COL})} \hspace{1em} \text{input:} \hspace{1em} \text{the table names.}

\textbf{VERT} \hspace{1em} \text{Integer} \hspace{1em} \text{input:} \hspace{1em} \text{the number of labels to be printed on the left side of the page.}

\textbf{RESTOR} \hspace{1em} \text{Logical} \hspace{1em} \text{input:} \hspace{1em} \text{If \textit{.TRUE.} the table will be printed on a new page, if \textit{.FALSE.} then an attempt will be made to print the table on the same page (see \textit{Description and Purpose} for details).}

\textbf{LINE} \hspace{1em} \text{Integer} \hspace{1em} \text{input:} \hspace{1em} \text{and output:} \hspace{1em} \text{the line counter.}

\textbf{SKIP} \hspace{1em} \text{Integer} \hspace{1em} \text{input:} \hspace{1em} \text{the number of lines between tables on the same page.}

\textbf{PAGE} \hspace{1em} \text{Integer} \hspace{1em} \text{input:} \hspace{1em} \text{the number of lines per page.}

\textbf{WIDTH} \hspace{1em} \text{Integer} \hspace{1em} \text{input:} \hspace{1em} \text{the number of characters per line (excluding carriage control character).}
UNIT Integer input: the number designating the output device.

IF AULT Integer output: 1 if NVAR, COL, or DIM is incorrectly specified;
2 if table will not fit on the defined page;
3 if TITLEN > WIDTH or LABELN > WIDTH - GAP;
4 if RESTORE is .FALSE. and LINE < 0;
0 otherwise.

Adjustable constants
The following values are defined in PARAMETER statements:

MAXVAR Integer constant: maximum number of variables permitted in a table.

GAP Integer constant: number of characters between label and start of data values.

RESTRICTIONS
None.

PRECISION
It does not seem worth making this algorithm adjustable for double precision. If double precision data are to be used, values should first be copied into a single precision array.

REFERENCES