TRYSYS^{TM} USER'S MANUAL

Robert B. Dean

and

Christopher Winters

Rev. August 2022

d^{2} Associates
720 W. Gordon Terrace #12C
Chicago, IL 60613

773.477.7830

robertbdean@gmail.com

Table of Contents

[1] The RLIST program is not yet available in this version of TRYSYS.

[2] The GYRO program is not yet available in this version of TRYSYS.

Introduction

(Draft Rev. 8-2022)

This *Manual* describes the use of the programs that are currently available in **TRYSYS**, the Fortran 77/90 version of the Tryon system of exploratory multivariate analysis. The original Tryon system was named "BCTRY" in the 1960s. It was then that the U.C., Berkeley psychologist Robert Tryon (1901-67) incorporated his innovative ideas on multivariate analysis in a large computer system of cluster analysis, factor analysis, and other statistical techniques. The development of this large computer system was made possible by substantial grants to Tryon from the National Institute of Mental Health. At the time of his death several dozen component programs had been fully developed and had been combined into an *integrated* statistical analysis package. All of the Tryon system programs, and their underlying statistical and psychometric procedures, are described in the Tryon and Bailey book *Cluster Analysis*, published in 1970. This volume brings together the statistical and psychometric ideas that Tryon originally formulated in the 1930s and later and developed further in papers published in the 1950s and 1960s (e.g., Tryon, 1957a, 1957b, 1958a, 1958b, 1959, 1966, 1967a, 1967b, 1967c, 1968a, 1968b).

Unfortunately, most expansion, conversion, and maintenance work on the Tryon system stopped after Tryon's death, and for many years the system was available only at a relatively small number of universities with CDC computers. However in 1985 d^2 Associates was awarded a Small Business Innovative Research (SBIR) grant by NIMH to begin conversion of the old BCTRY CDC Fortran 4 and Compass assembly language code to portable Fortran 77. This conversion effort (and other programming work) was continued under an NIMH SBIR Phase II grant for 1987-89.[1]

All of the programs needed to perform Tryon "V-analysis" (e.g., "key-cluster" factoring) were successfully converted. These included the programs DAP, COR2, COR3, DVP, CC5 and CSA2. In addition, the following programs were converted: (1) the spherical analysis program SPAN2; (2) the variable suppression program SLEP1 and the variable reinstatement program SLEP2; (3) the comparative analysis (factor matching) programs COMP1 and COMP2; (4) the programs for saving and retrieving analysis files, GIVE and TAKE; and (5) the DPRINT and COMMENT programs. The cluster/factor scoring program FACS and the object clustering program OYTPE were converted during the 1990s.

The old CDC version of the system could handle only 90 variables and 5,000 cases. Because of "parameterization" of the Fortran 77/90 code, the new system can handle as many variables or cases as needed. Currently TRYSYS is set to handle 600 variables and 120,000 cases.

None of the IBM 7090 "BIGNV" programs for performing unrestricted factoring were ever converted to run on the Berkeley CDC 6400 machine. The same is true of the factor rotation/matching programs SYDA and SYRN. The old IBM 7090 source code for all these programs appears to have been lost. But with the

[1] NIMH grant R43 MH38975-O1A2 during 1985. NIMH grant R44 MH38975-02 during 1987-89.

availability of very large physical and virtual memories on current machines, the need for the BIGNV programs no longer exists. Also, the grant-developed TRYCLUS program can handle correlation matrices of up to 1,000 x 1,000 in size.

Brief descriptions of the CDC Fortran 4 programs that were not converted are given in the "Component Programs of the Tryon System" document (along with descriptions of the converted programs).

The chief programmer on the Tryon system conversion project was John Bauer. The knowledge and experience he gained working on the BCTRY development project at Berkeley during the 1960s enabled him to complete the complex code conversion and reprogramming tasks in an efficient manner. His contribution to the production of this Fortran 77/90 version of the Tryon system has been invaluable.

The support of NIMH for the development of the current TRYSYS program is gratefully acknowledged.

Work on this *TRYSYS User's Manual* is an on-going project, and we would very much appreciate suggestions for improving it. Also, please bring to our attention any errors you find in the *Manual* -- factual, grammatical, typographical, and otherwise.

Following are the currently available TRYSYS modules and commands:

DAP	**Processes input data**
COR2	**Calculates correlations, for complete data**
COR3	**Calculates correlations, pairwise deletion of missing data**
DVP	**Calculates/selects initial communality values**
GIVE	**Writes the binary ISF file to a user-named file**
TAKE	**Retrieves a user-saved ISF file**
SELECT	**Selects a subset of variables for factoring**
SLEEP	**Suppresses a set of variables prior to factoring**
CC5	**Performs Tryon key-cluster, preset (multiple-group/ cluster), diagonal and salient/full centroid factoring**
GYRO	**Performs quartimax and varimax ortho. factor rotation**
RESTORE	**Restores all suppressed variables after factoring (extension analysis)**
CSA2	**Provides oblique cluster/psychometric statistics for dimensions defined by 1 or more variables**
SPAN2	**Shows factor structure (3 dimensions at a time) via variable positions on the surface of a unit sphere (spherical analysis)**
COMP1	**Writes to a file oblique or orthogonal factor coefficients found for a group**
COMP2	**Calculates various factor similarity coefficients for the results from two or more groups**
FAST	**Writes to the output and also to the ISF the residual matrix after *k* dimensions are extracted**
OTYPE	**Performs iterative-partitioning object cluster analysis**
RSCAT	**Produces scatterplots for both variables and cluster scores**

GIST **Allows manual input of runtitles, varnames, *r* matrices, ortho. factor loading matrices and so on**

Installing TRYSYS Files

A small number of Python language console applications provide an interface for the TRYSYS batch-processing (non-interactive) program. For ease of use we recommend that all of these apps be put into the same **c:\trysys\** directory as trysys.exe itself. This directory would contain the following files: trysys.exe, tryrun.py, trydap.py, trytake.py and the module files massagemod.ascii and massagemod.utf8, as well any language specific mods.

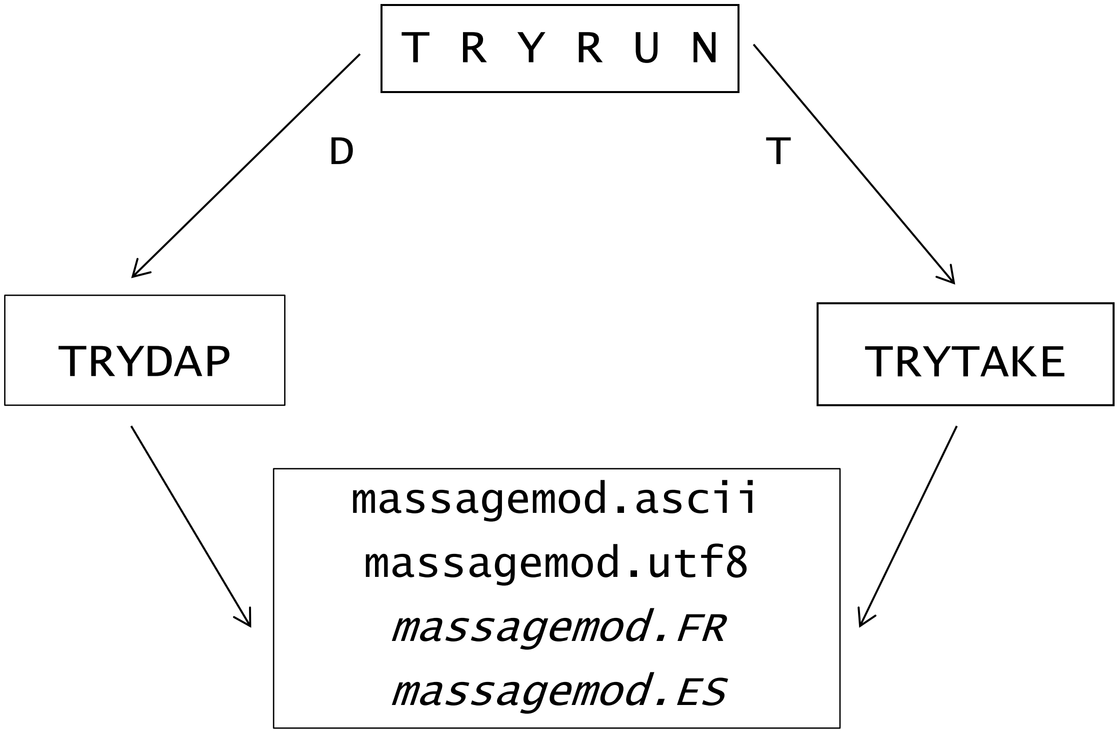
The massagemod.utf8 file is used when a TRYSYS command file has Unicode characters in Comment lines and/or in an Analysis Title. The trydap and trytake apps ask if your command file contains such characters (e.g., Chinese, Russian, Japanese, and so on). Enter Yes if so; Return otherwise. Also, a number of language mods are under development to convert certain output text into languages other than English.

Installing Python

A Python installation file can be downloaded from the website www.python.org . Click on *Download Latest Version for Windows*. Then under *Stable Releases/Download* click on *Windows x86/64 Executable Installer*.

Save the installer file to your Downloads directory. Click on the file and chose the *Custom Install* option. Under *Optional Features* leave all 5 options selected. Under *Advanced Options* leave all 3 default options selected. In particular make sure that the Add Python to Environmental Variables option is selected. This is important to do and permits Python applications to be run from any directory you are in.

Running TRYSYS



Note that tryrun.py is the *only* Python app that needs to be in every data directory you'll be working in. As indicated in the diagram on the previous page, it calls the prompting programs trydap.py or trytake.py. These in turn use one or more of the output file post-processing massagemod modules, as needed.

TRYRUN.py will ask if you want to process raw data or use an already existing correlation matrix. Enter **D** for the former & **T** for the latter.

TRYDAP.py will ask you for the name of a **command** file and the name of a **data** file. It will also ask if your command file contains Unicode characters.

TRYTAKE.py will ask for the name of your **command** file and the name of a **GIVE** file. GIVE files contain saved analysis results (e.g., *r* matrices, initial communalities, etc.) in compact binary form. Like TRYDAP, it also asks for the presence of Unicode characters (see the earlier paragraphs).

If you'd like some of the output text to be in (for example) French or Spanish, you should chose FR or ES when prompted at the end of a TRYDAP or TRYTAKE run. Other language options are under development.

www.trysys.info

TRYCMD: Program to Build Command Files

TRYCMD.PY is a Python console application that allows users to build TRYSYS command files. Place it in each data directory from which you'll be running TRYSYS. This application remains under development, so certain analysis options will require additional editing after an initial command file has been built. The text editing program Notepad++ is an excellent one to use, since it indicates the line number and the column location of the cursor. Note that many TRYSYS modules do not require the user to specify *any* analysis parameters (e.g., GIVE, TAKE, DPRINT, COR2, COR3, RESTORE, FAST, COMP1, ISFDMP). In addition the modules CSA2, SPAN2, and OTYPE currently have default parameters set.

Use of TRYCMD is quite simple. First decide on the analysis sequence you want (e.g., DAP, COR3, GIVE, DVP, CC5, CSA2, SPAN2). Then double click on it to start the program, enter a name for your file, and at each Command: prompt enter (in correct sequential order), the name of the component program that you wish to use. TRYCMD will prompt you if any additional information is needed for a particular component,

It is best to be somewhat familiar with the material in the *Manual* before using this program.

TRYCMD writes the following at the beginning of each command file:

```
*SET_NCLINE 118
*SET_NLPAGE 86
*START
*COMMENT
```

NCLINE sets the maximum number of characters in a TRYSYS output **line**. NLPAGE sets the maximum number of lines per output **page**. The user may edit these values somewhat, as desired. However, the default values should suffice for most purposes.[1]

www.trysys.info

[1] In BCTRY use on CDC mainframes, the maximum line length was set to 132 characters because of the then use of wide paper for output listings.

ISF: The Intermediate Storage File System

The ISF (Intermediate Storage File) system (mentioned in the Manual as "the ISF") is a TRYSYS mechanism for storing in compact binary form analysis results produced by the various component programs—which (usually) are needed by one or more following programs. For example: the raw data processing program DAP writes variable and case names, and variable sums, means, and standard deviations, to various ISF files; the COR programs write the correlations they produce to an ISF file named CORRM1; the DVP program writes diagonal values (usually initial communality estimates) to an ISF file named DIAGV1; and so on.

These ISF files are created and used during the course of an analysis but are lost after the analysis is completed, unless they are saved by a *GIVE command. GIVE saves all of the ISF files created prior to its call so that these files can be used in future analyses. A *TAKE command restores the saved files to the TRYSYS ISF system during a later run. Use of GIVE and TAKE can reduce substantially the time needed to run multiple correlational analyses on large data sets (e.g., with many hundreds of variables and tens of thousands of cases).

In the I/O section of the description of each program, the *ISF Input* section lists the ISF files the program *reads* from the ISF, and the *ISF Output* section lists the ISF files the program *writes* to the ISF.

A table showing the ISF I/O for each TRYSYS program is in preparation.

For details on the operation of the ISF in the original Berkeley CDC Cyber 6400 BCTRY program, see Tryon and Bailey (1970), chapter 3 and chapter 12 (pp. 269-273).[1] Also see the BCTRY section on the website www.trysys.info.

www.trysys.info

[1] In the book the ISF is called the "IST" (for Intermediate Storage Tape).

START, END, COMMENT

The ***START** command still must be used to initiate execution of the program. But see the TRYCMD section for what this app automatically writes at the beginning of each command file it is building.

The ***END** command still must be the last one in the command file. It is used to terminate execution of the program.

The ***COMMENT** command is used to insert explanatory comment lines into the program's output before (or after) output produced by any of the TRYSYS programs. COMMENT may be called as many times as desired, and as many comment lines as desiredmay be used. Blank lines may be used to separate blocks of the text.

Note that TRYSYS commands may now also be in lower case letters (see below).

Example

```
*START
*COMMENT

       A correlation matrix saved by a previous GIVE
       command is retrieved using the TAKE command.

*comment

       In this run SMC values are calculated as initial
       communality estimates (DVP), and a (default) Tryon key-
       cluster analysis (CC5) is carried out. These analysis
       results are saved by a *GIVE command.
*TAKE
*dvp
  10
*cc5
  blank line
  blank line
*give
*END
```

GIVE: Program to Save Analysis Results

Introduction

The *GIVE command is used to save analysis results that have been written to the ISF during the course of an analysis (see the ISF section). GIVE may be called at *any* point after the correlation matrix has been generated. When the GIVE and TAKE programs are used, correlations, communality estimates (diagonal values), and other results need be calculated only once. Note that GIVE does not have to be the last program called in a run; other programs may follow it. GIVE saves all files present on the ISF *at the time it is called*, including any that have been restored to the ISF by TAKE.

Use

The only command required is ***GIVE** .

Example

```
*START
*COMMENT

     Analysis sequence:  Process raw data (DAP), calculate
     correlations (COR3) & communality estimates (DVP), save
     these results (GIVE), and run a (default) Tryon
     key-cluster analysis (CC5).

*dap
  .
  .
  .
DATA
*cor3
  blank line
*dvp
  50
*GIVE
*cc5.
  blank line
  blank line
*END
```

- GIVE saves the correlations and diagonal values for future analyses.

I/O

Print Output: GIVE writes a list of the ISF files it is saving.

TAKE: Program to Restore Analysis Results

Introduction

The *TAKE command is used to restore to the ISF of a new run analysis results saved by a previous GIVE command (see the ISF section). Further analyses can then be carried out using the restored files (e.g., correlations and diagonal values). Except for COMMENT lines, TAKE must always be the first command (after START) when the ISF files are being restored. Note that all TRYSYS analyses must begin either with raw data processing (via DAP) or with retrieval of a saved correlation matrix, etc. (via TAKE).

Use

The only command required is ***TAKE** .

Example

```
*START
*comment

       Analysis sequence:  Restore a saved ISF (TAKE),
       calculate communality estimates (DVP),and run (default)
       key-cluster (CC5) & cluster structure (CSA2) analyses.

*TAKE
*dvp
  41
*cc5
  blank line
  blank line
*csa2
  blank line
  blank line
*GIVE
*END
```

- TAKE retrieves a previously saved correlation matrix, and CC5 & CSA2 analyses are carried out. These results are saved via a GIVE command.

I/O

Print Output: TAKE writes a list of the ISF files it is restoring.

www.trysys.info

DAP: Raw Data Processing Program

Introduction

DAP reads the raw data and processes it for use by one of the correlation programs or by the cluster/factor scoring program FACS. DAP allows you to name variables and to permanently change their sign and/or reorder them. It handles missing data, checks for non-numeric values in the data fields, and computes for each variable its mean, standard deviation, minimum and maximum values, and the number of missing values.

If you *incorrectly* indicate that you have complete data, TRYSYS terminates execution after DAP. In this case DAP will output the variable name and case name (or number) for each instance of missing data.

Use

The command ***DAP** calls the program.

❑ On the following line, enter an **analysis title** of up to 72 characters. This title is stored in the ISF and is used by many of the component programs.

❑ On a second line, specify the number of variables, indicate if you have missing data, and if you wish case names to be written to the output. If you don't specify the number of cases, DAP will determine the N.

cols.

1-4 Enter the number of variables (right justified).

5-9 Enter the number of cases (right justified). (*Optional*.)

12 *Blank* if any data are missing. **1** if the data are complete.

16 **1** if you wish case names to be written to the output listing file. *Blank* otherwise.

20 If you wish to specify a missing data **value**, enter it here. Use the numbers 0-9. For variables with a field width of 2 or greater, fill missing data fields with the number you select. Note that with this option you cannot also use blank fields to represent missing data. *Blank* otherwise.[1]

21-32 If you wish to provide a "root" for the automatic generation of variable names, enter this in cols. 21-32. A root name may be no more than 12 characters. *Blank* otherwise.[2]

[1] This option was added by John Bauer in 2004.

[2] See footnote 1.

- ❑ On a third line (more if needed), enter a **data format statement**. This statement must be enclosed in parentheses. If you need (or want) to continue the format statement on following lines, a comma or slash must be the last character on all lines except for the last one. A case identification field of up to 16 characters may be specified at any location in the statement (use the A format; see the example 1). In general, *fixed-field* data format specifications acceptable to SPSS and other such programs should be acceptable to TRYSYS.

- ❑ DAP has three ways to provide variables names. First, it can automatically produce variable names in the form VAR*nnn*, where *nnn* represents the ordinal number of each variable. This is the default option.

  Second, the user can specify a *root* name for automatically naming all variables.

  Third, the user can specify each variable name. To do this enter **VARIABLE NAMES** on the line after the data format statement. Then on following lines enter your names. Names may be up to 20 characters long and may include upper and lower case letters (including accented ones), numbers, and the following special characters: @ # $ % & * ( ) - + < > ? . There is no restriction on which character may occur as the first one in a name. Names must be separated by commas and should not be broken between lines. Also, the list of names must be terminated by a period. Finally, no spaces (blank characters) are permitted in the names.

- ❑ You can *permanently* reorder all of the variables immediately after the data have been read in. Enter **REORDER** on one line, and enter **all** variable names in their new order on following lines.

- ❑ You can also *permanently* change the sign of one or more variables. Enter **REFLECT** on one line, and enter the names of the variables to be reflected on following lines (see example 2).

DATA still must be the last subcommand in the DAP command sequence.

<u>Special note</u>:

Use the command ***PR_LNS** at any point in a command file to get an output list of regular and *accented* Latin letters permitted in variable names, including a list of permitted special symbols and characters (ASCII codes <256; see below).

Also, as a reminder, Unicode characters are now permitted in comment sections and in analysis titles. However, they require 2-3 times as much space (in bytes) as ASCII characters. Hence lines with such characters will need to be much shorter than lines with ASCII-only characters.

Ignore the unprintable text written to the *screen* as this command is processed.

Example *PR_LNS **printed** output:

```
Numerials: 0123456789
Upper case: ABCDEFGHIJKLMNOPQRSTUVWXYZ
Lower case: abcdefghijklmnopqrstuvwxyz
Symbols:  @#$%&*()-+=<>
Diacritics: ÀàÈèÌìÒòÙù
Diacritics: ÁáÉéÍíÓóÚúÝý
Diacritics: ÂâÊêÎîÔôÛû
Diacritics: ÃãÑñÕõ
Diacritics: ÄäËëÏïÖöÜüÿ
Diacritics: Åå
Diacritics: Ææ
Diacritics: Çç
Diacritics: Ðð
Diacritics: Øø
Diacritics: Þþ
Sharp S: ß
```

Example 1

```
....+....1....+....2..

*PR_LNS
*DAP
Tryon Key-Cluster Analysis of All CPI434 Items. N=9,800.
 434 9800          7 CPI434-
(200F1.0/200F1.0/34F1.0,A16)
DATA
```

- In this example there are 434 variables, 9,800 cases, and missing data.
- The missing data indicator is 7. Blank fields will be treated as data errors.
- Variables will be named using the root CPI434-.

Example 2

```
....+....1....+

*DAP
Testing Diagonal Factoring of the 1st 100 Items of the MMPI-2.
 100      1
(100F1.0)
VARIABLE NAMES
MMPI2-1,MMPI2-2,MMPI2-3,MMPI2-4,MMPI2-5,MMPI2-6,MMPI2-7,
MMPI2-8,MMPI2-9,MMPI2-10,MMPI2-12,MMPI2-13,MMPI2-14,MMPI2-15,
.
.
.
MMPI2-89,MMPI2-90,MMPI2-91,MMPI2-92,MMPI2-93,MMPI2-94,
MMPI2-95,MMPI2-96,MMPI2-97,MMPI2-98,MMPI2-99,MMPI2-100.
REFLECT
MMPI2-2,MMPI2-3,MMPI2-8,MMPI2-10,MMPI2-20,MMPI2-33,MMPI2-57,
MMPI2-75,MMPI2-83,MMPI2-90,MMPI2-91,MMPI2-95.
DATA
```

- In this example there are 100 variables, and no missing data.
- DAP will determine the number of cases.
- All variable names are specified by the user.
- Twelve of the variables are permanently reflected.

I/O

Print Output: DAP writes the following information to the output: analysis title; number of variables and cases; data format statement; variable names; and for each variable its mean, standard deviation, minimum and maximum value, and number of cases with missing data.

ISF Input: DAP reads no files from the ISF.

ISF Output: DAP writes the following files to the ISF:

IDFILE	analysis identification
VNAMS1	variable names
VSUMS1	sums
MEANS1	means
STDEV1	standard deviations
ONAMS1	case (subject, object) names
VAREN1	*sample sizes*[3]

DSF Output: DAP writes raw scores, missing data indicators, analysis title, and the number of variables to the DSF.

www.trysys.info

[3] Written only when there are missing data.

DPRINT: Program to Print the Raw Data

Introduction

DPRINT provides a complete listing of the raw data processed by DAP. It can be useful as a check on data accuracy. The program writes data for up to 10 variables and 50 cases on each output page. MISSING is written in the output to indicate missing data values. Note that DPRINT's output can be quite voluminous.. As an example, DPRINT would produce 200 pages of output for a data set with 100 variables and 1000 cases! If you are confident about the accuracy of your data, there is little reason to use DPRINT. Also, as an alternative to DPRINT, DAP may be used to obtain a compact listing of all missing data, case by case. You do this by setting the DAP parameter for missing data to No Missing.

Use

The only command required is ***DPRINT** .

DPRINT must be called in the same run as DAP, since it reads data from the DSF file produced by DAP.

I/O

ISF input: DPRINT reads the following files from the ISF:

IDFILE	analysis identification
VNAMS1	variable names
ONAMS1	case (subject, object) names

DSF input: Raw scores are read from the binary DSF system file produced by DAP. Note that the DSF file is never saved for future use.

ISF output: DPRINT writes no files to the ISF.

www.trysys.info

COR2: Correlation Program for *Complete* Data

Introduction

COR2 produces a matrix of Pearson product-moment correlation coefficients using all of the variables processed by DAP. COR2 should be used *only* if you have complete data. If you have any missing data, correlations based on such data will be incorrect. Any blank fields will be interpreted as valid zero values, and any missing-data indicators will be interpreted as valid data as well (see the DAP and DPRINT sections).

The correlation matrix produced by COR2 is stored in the ISF. When the GIVE and TAKE commands are used, the matrix needs to be calculated only once for a given data set (see the ISF section).

Use

The only command required is ***COR2** .

I/O

Print Output: COR2 writes the correlation matrix in lower triangular form.

ISF Input: COR2 follows DAP and reads the following files from the ISF:

IDFILE	analysis identification
VNAMS1	variable names
VSUMS1	sums
MEANS1	means
STDEV1	standard deviations

DSF Input: Raw scores are read from the binary DSF system file produced by DAP.

ISF Output: COR2 writes the following files to the ISF:

CORRM1	correlations

www.trysys.info

COR3: Correlation Program for *Missing* Data

Introduction

COR3 is the program you should use if you have *any* missing data. Pearson product-moment correlations are computed using pairwise matched observations. The program provides sample moment estimates of the means and standard deviations. Because it must handle missing data, COR3 is slower than COR2. For large data sets we suggest that you use the GIVE and TAKE programs to avoid having to calculate your correlation matrix more than once.

You should use caution in analyzing data that have more than just a small number of missing values. If you have a great deal of missing data, or if there are distinct patterns of it, you could end up with spurious analytical results. Various data imputation techniques been proposed over the years, and you should consider preprocessing your data using one of these procedures before using TRYSYS. See Enders (2010) for a comprehensive treatment of this topic.[1]

Method

The formulas for the calculation of means and standard deviations are the standard ones for sample moment unbiased estimates.

Use

COR3 currently has its analysis options **preset** to do the following:

- Calculate the correlation matrix and write it to the ISF.
- Write the correlation matrix to the output in lower triangular form.

The command ***COR3** calls the program. Add a blank line after *COR3.

[1] Enders, C.K. (2010). *Applied missing data*. New York: Guilford Press. Also check out his companion website appliedmissingdata.com .

Example

```
*COR3
  blank line
```

- This is the current COR3 default. The correlation matrix is written to the output, and correlations (and matched Ns, means & standard deviations if calculated) are written to the ISF.

I/O

Print Output: COR3 writes the correlation matrix in lower triangular form.

ISF Input: COR3 follows DAP and reads the following files from the ISF:

IDFILE	analysis identification
VNAMS1	variable names
VSUMS	*sums*[1]
VAREN1	*sample sizes*[2]
MEANS1	means
STDEV1	standard deviations

DSF Input: Raw scores are read from the binary DSF system file produced by DAP.

ISF Output: COR writes the following files to the ISF:

2SENS1	*matched Ns*[3]
2MEAN1	*matched means*[3]
2STDV1	*matched standard deviations*[3]
CORRM1	correlations

www.trysys.info

[1] Read only if data are complete.
[2] Read only if there are missing data.
[3] Written only if there are missing data.

DVP: Diagonal Values Program

Introduction

DVP computes or selects a set of values to be inserted in the principal diagonal of the correlation matrix. The values usually desired are estimates of the communalities of the variables, although other values may be used for special applications (e.g., unities are used in principal components analysis). The basic definition of the communality of a variable is "that proportion of its variance that can be accounted for by the common factors" (Gorsuch, 1983, p. 29). DVP calculates communality estimates using a variety of methods, many not available in other programs (but which are now rarely if ever used).

There is still some difference of opinion on which method provides the best *initial* estimate of a variable's communality. However, as Gorsuch points out in his discussion of this and related issues (pp. 102-111 and 121-126), different methods for communality estimation tend to give similar *final* communality results when the number of variables is reasonably large (e.g., 40-50 or greater). Also, nearly all factoring procedures iterate communalities at least several times, or until some convergence criterion is reached.[1]

Method

All of the methods available in DVP, except for the squared multiple correlation method (SMC), are described in Tryon and Bailey (1970) *Cluster Analysis* (chapter 5 and pages 282-288). Based on some empirical evidence, Tryon recommends his "modified approximation B" method for most applications. The MODB estimate is the average of (a) the highest absolute correlation of a given variable with the other NV – 1 variables and (b) a cluster composite of the three variables most collinear with it. Gorsuch indicates that using the highest absolute correlation (method HIGHR), with multiple factoring iterations, can also produce good results.

If you wish to see two or more initial communality estimates in a single run, make multiple calls to DVP. But note that only the method you use last will have its values stored in the ISF and be used in later analyses.

[1] See the CC5 section for the iteration options available in TRYSYS.

Use

The command ***DVP** calls the program.

❑ On the following line (or lines) specify the DVP method you wish to use and chose options for certain of the methods:

cols.

3-4 Enter the **number** of the DVP method you wish to use:

10	*Squared multiple correlation (SMC)*[2]
30	*Quadratic formula (QF[i])*
40	*Approximation B (B)*
41	*Modified approximation B (MODB) (recommended)*
50	*Highest absolute correlation (HIGHR)*
51	*Average absolute correlation*
52	*Triads*
70	*Zero (0)*
71	Unity (1)

7-8 Enter a value *n* to produce a list of the *n* highest correlations for each variable. *Blank* if you don't want this information.[3]

15-16 *If you select method QF,* you need to enter another number to indicate a preliminary method (HIGHR is recommended).

19-20 *Used by method QF only.* Enter the number of predictor variables to be used with each variable to compute the communality estimates (10 is the maximum). If left *blank*, the number of predictors will be either 10, or less than half the number of variables in the matrix, whichever is smaller.

21-24 *Used by method QF only.* Enter a number which is the convergence criterion for QF (.010 is a reasonable value). The decimal point must be entered.

28 *Used by method QF only.* Enter a 1 to obtain a listing of the successive estimates of communality for each iteration. *Blank* otherwise.

[2] This method was added by James Fleming (in 2004).
[3] This revised option was added by John Bauer (in 2004).

Example 1

```
....+....1

*DVP
  41   6
```

- The Tryon Modified Approximation B (MODB) method is requested.
- The 6 highest correlations for each variable are also requested.

Example 2

```
*DVP
  51
*DVP
  50
*DVP
  10
*DVP
  41
```

- Four methods are requested: Average *r*, HIGHR, SMC, and MODB. Only the MODB estimates will remain stored in the ISF.

Example 3

```
....+....1....+....2....+....3

*DVP
  30          50   8.005   1
```

- The Quadratic Formula (QF) method is requested, and HIGHR is chosen as the preliminary method.
- The number of predictor variables selected is 8, and the convergence criterion chosen is .005.
- Successive communality estimates are listed for each iteration.

I/O

Output Listing: DVP writes a description of the methods and options chosen, a list of the diagonal values, and, if requested, the *n* highest correlations for each variable, in their rank order.

ISF Input: DVP follows COR and reads the following files from the ISF:

IDFILE	analysis identification
CORRM1	correlations

ISF Output: DVP writes the following file to the ISF:

DIAGV1	diagonal values

www.trysys.info

CC5: Key-Cluster, Multiple-Group, Centroid and Diagonal Analysis Program

Introduction

This versatile factoring program is one of the two key programs of TRYSYS (CSA2 is the other). CC5 permits either empirical or rational ("preset") selection of a **subset of variables** as the defining variables of each dimension, and computes factor coefficients and other statistics for all variables on each dimension. In its primary empirical mode (Tryon's method of "cumulative communality key-cluster analysis"), CC5 successively selects as dimension-definers variables that are (usually) tightly collinear, extracts these cluster dimensions from the correlation matrix, and continues this factoring process until the default or a user-specified communality exhaustion criterion is reached. Key-cluster analysis is fast, provides a good estimate of the optimal number of interpretable dimensions, and in effect provides a direct oblique solution. Besides Tryon key-cluster analysis, CC5 may be used to perform other types of analysis such as diagonal (square root, or pivot variable) factoring, and various forms of multiple-group analysis (including centroid factoring).

Method

Key-cluster analysis is discussed in some detail in *Cluster Analysis* (chapter 6 and pp. 288-296). See Gorsuch (1983, chapter 5) for a discussion of diagonal and multiple-group analysis. These methods are also discussed in greater mathematical detail in Harman (1976, pp. 101-102 and 234-243). Also see Comrey and Lee (1992, chapter 3) and Harman (pp. 166-173) for a discussion of centroid analysis as a variation on multiple-group factoring.

Use

The command ***CC5** calls the program.

❑ On the following line, specify the computation, printing, and other options you wish.

col.

4 Indicate how you wish the **number of dimensions**, *k*, to be determined:

Blank (or **1**), by the communality exhaustion criterion.

2, by the sum of squared correlations exhaustion criterion.

3, by the communality increment criterion.

4, to be preset by the user (in cols. 19-20); or by pivot variable or salient centroid analysis (see col. 12).

8 Indicate how you wish **pivot variables** to be selected:

Blank (or **1**), select the variable with the highest variance of squared correlations.

2, the key variable selection method chosen (in col. 12) also determines the pivot variable.

12 Chose one of the following methods for selecting the **defining variables** of each dimension:

Key-Cluster Analysis (collinear subset):

Blank (or **2**), most mutually collinear subset of variables.

1, variables most collinear with a single pivot variable.

Preset:

3, the user selects all of the definers.

Pivot Variable Analysis (one definer per dimension)[1]:

5, the single pivot variable having the highest residual communality.

6, the single pivot variable having the highest sum of squared correlations.

7, the single pivot variable having the highest variance of squared correlations.

Salient Centroid Analysis:

When the total number of variables exceeds 20, the selected set of definers of each dimension consists of:

8, the 20 variables having the highest variance of squared correlations.

9, the 20 variables having the highest sum of squared correlations.

19-20 *If the number of dimensions k is preset* (**4** in col. 4 and/or **3** in col. 12), enter *k* here. *Blank* otherwise.

23-24 *If you wish to specify a fixed number of iterations*, enter the number here. If left *blank,* the number of iterations is 4. Enter a **0** to have iterations stopped by satisfaction of the communality difference criterion.[2]

31-32 Enter the maximum number of successive pivot variables to be tested as pivots of variable subsets that meet the P^2 criterion.[3] The default number is **4** (*blank*).

68 *Blank* to suppress printing of the residual matrix after each dimension has been extracted. **1** to print these matrices. Note that this output can be very voluminous.

72 *Blank* to suppress printing of the factoring results for each successive iteration. **1** to print these results. This output, too, can be quite voluminous.

[1] Diagonal analysis.

[2] See the following command line, col. 24.

[3] See the following command line, cols. 25-32.

❑ On a second line, specify criterion values for terminating factoring and iteration, and for accepting definers of dimensions. If an option is not being used, and its criterion is therefore not applicable, just leave the relevant field blank.[4]

col.

1-8 *If you have chosen one of the collinear subset (key-cluster) methods of factoring*[5], enter an index of proportionality (P^2) criterion for accepting more than four definers of a dimension. The default value of P^2 is **.81** (*blank*).

9-16 *If you have chosen one of the exhaustion methods to determine the number of dimensions* $\underline{k}$[6], enter an exhaustion criterion to be met by the last (k^{th}) dimension. The default value is **.92** (*blank*).

17-24 *If you have not specified a fixed number of iterations*[7], enter a communality difference criterion for terminating iteration. The default value is **.01** (*blank*).

25-32 *If you have chosen one of the collinear subset methods,* enter a P^2 criterion for accepting the second, third, and fourth definers of a dimension. The default value is **.40** (*blank*).

❑ *If you are performing a **preset analysis***, you need to enter next the *ordinal* numbers of the definers of your *k* preset dimensions. These ordinal numbers must be entered in 20I4/20I4 format (right justified within each field). If you have less than 21 definers per dimension, be sure to include a blank line after each set of definers (see example 3). Also, do *not* use a negative sign before any number.

If you have used SLEP1 to suppress some variables, and have not restored them with SLEP2, then be sure to use the *reordered* ordinal numbers as printed by SLEP1.

[4] This command line is very often left entirely blank (see the examples).

[5] See the first command line, col. 12.

[6] See the first command line, col. 4.

[7] See the first command line, col. 23-24.

Example 1:

```
*CC5
  blank line
  blank line
```

- *All* the default options are chosen. A standard key-cluster analysis is performed.

Example 2:

```
....+....1....+....2....+

*CC5
   4   2   5       9
  blank line
```

- The pivot variable analysis method of factoring is used.
- Pivot variables having the highest residual communality will be selected as the single definers of dimensions.
- The number of dimensions is preset to 9.

Example 3:

```
....+....1....+....2....+....3....+....4

*CC5
   4   2   3       6
  blank line
   2   3   4   5   6   8   9  10  11  14
  blank line
  15  17  18  19  21  22  25  27  28
  blank line
  30  33  35  37  38  40  42
  blank line
  35  47  51  54  57  59
  blank line
  65  67  69  71  72
  blank line
  90  93  95  99 101 103
  blank line
```

- A preset (multiple-group) analysis is requested.
- The number of preset dimensions is 6.

I/O

Output Listing: CC5's output can be quite voluminous and will vary depending on the analysis options chosen. In general, the output is reasonably well-labeled and is relatively self-explanatory. However, you may wish to consult *Cluster Analysis* for the definition of some of the specialized statistical terms used in the listing.

ISF Input: CC5 follows DVP and reads the following files from the ISF:

IDFILE	analysis identification
CORRM1	correlations
DIAGV1	diagonal values

ISF Output: CC5 writes the following files to the ISF:

DIAGV1	diagonal values
UFACT1	unrotated factor coefficients
CLUST1	cluster indicators
REFLX1	reflection indicators

www.trysys.info

CSA2: Cluster Structure Analysis Program

Introduction

CSA2 provides a comprehensive statistical and psychometric description of the correlational properties of oblique cluster dimensions. Variables defining these dimensions are most often empirically derived by CC5, but may also be chosen on theoretical or other rational grounds. A particularly helpful feature of this program is its calculation of the reliability for the initial set of collinear variables defining each dimension. CSA2 also automatically shows how cluster reliability can be improved through the inclusion of additional collinear variables.

Method

The cluster structure statistics produced by CSA2 are discussed in detail in *Cluster Analysis* (chapter 7 and pages 270-281).

Use

The command ***CSA2** calls the program.

- ❑ On the following line, specify the computation, printing and other options you wish.

col.

4 *Blank* (or **0**) to analyze the matrix of observed correlations among variables. **1** to analyze a matrix of correlations reproduced from the orthogonal factor coefficients.

8 *Blank* (or **2**) to suppress the formation of a "clustered" correlation matrix. **0** to print this matrix.

12 *Blank* (or **0**) to suppress the calculation of *augmented* oblique factor coefficients. **1** to calculate and print these coefficients.

16 *Blank* (or **0**) to suppress the calculation of an augmented correlation matrix. **1** to calculate and print this matrix.

24 *Blank* (or **0**) to suppress the calculation of correlations of raw cluster scores with cluster domains. **1** to calculate and print these correlations.

28 1 to print variable "titles" if available. U

- ❑ On a second line, enter in columns 1-8 a communality "lower limit" value. Variables whose communalities are below this limit are excluded from the "expanded cluster structure" section of the output.[1] If this line is left *blank*, the default value of **.20** is used.

Note that sometimes a variable with a low communality is selected empirically as a dimension definer. Such low-communality variables invariably have low factor loadings and should rarely be retained as dimension definers.

[1] Where variables that are not "definers" are assigned to the oblique dimension on which they have their highest factor loading.

Example 1:

```
*CSA2
  blank line
  blank line
```

- *All* the default options are chosen.

Example 2:

```
....+....1....+....2....+

*CSA2
   1   0   1   1       1
.125
```

- In this example reproduced correlations are analyzed and a clustered correlation matrix is printed. All three of the special calculations are performed, and the results are printed.
- The communality lower limit is set to .125.

I/O

Output Listing: CSA2 always writes the following information: defining variables of the clusters; correlations among the cluster composites (and among the cluster domains); Spearman-Brown reliability coefficients, and domain validities of the cluster composites; the oblique factor coefficients; the generality of each cluster; and reliability information for expanded cluster composites. In addition, a "clustered" correlation matrix, an augmented correlation matrix, augmented oblique factor coefficients, and correlations of raw cluster scores with cluster domains are printed as requested.

ISF Input: Depending on the analysis options chosen, CSA2 reads the following files from the ISF:

IDFILE	analysis identification
VNAMS1	variable names
REFLX1	reflection indicators
CORRM1	*correlations*[2]
DIAGV1	*diagonal values*[2]
UFACT1	*unrotated factor coefficients (orthogonal)*[3]

ISF Output: CSA2 writes the following files to the ISF:

RFACT1	rotated factor coefficients (oblique)
BASIS1	correlations among the oblique factors

www.trysys.info

[2] Read when observed correlations are to be analyzed.

[3] Read when reproduced correlations are to be analyzed.

SPAN2: Spherical Analysis Program

Introduction

The spherical analysis program SPAN2 produces 3-space graphical representations of cluster/factor structure, depicting variables as points on (or very near) the surface of unit spheres. Variables close together on each spherical surface will have very similar augmented factor loadings on the three dimensions being represented. Each spherical configuration is projected to a 2-space plane, for printing as part of the output. For analyses with four or more dimensions, SPAN2 produces a parsimonious set of 3-space spherical plots, in order to best represent the common factor variance in the total hyperspherical configuration. Spherical analysis was originally discussed by Thurstone, Guilford, Tryon, and others during the 1930s, but only Tryon implemented the technique for computers (in the 1960s).

Note that the term "subspace" is used below to denote a space of *any* dimensionality.Additional material is in preparation.

Method

See *Cluster Analysis* (pages 94-100 and 296-303), for the procedures involved in spherical analysis. This technique is especially useful for visualizing oblique factor structure. There are numerous examples of SPAN figures throughout the book.

Use

The command ***SPAN2** calls the program.

❑ On the following line specify the program options you wish.

col.

4 Indicate how you wish the subspaces to be selected:

Blank (or **1**), SPAN2 selects all the subspaces (note that it selects only **3**-dimensional spaces).

3, the program selects subspaces, as above, but the user also designates additional subspaces.

2, the *user* selects all of the subspaces.[1]

8 Enter the source of the orthogonal factor coefficients to be analyzed:

Blank (or **1**), for unrotated coefficients from CC5.

11-12 Enter $k_{min.}$, the *lowest* dimensional subspace desired (you may specify from 1 to 20). SPAN2 calculates tables for all subspaces from $k_{min.}$ up to either (a) the number of dimensions obtained during factoring or (b) the $k_{max.}$ value you specify below (whichever is less). The default value is **3** (*blank*).

15-16 Enter $k_{max.}$, the *highest* dimensional subspace desired (from 1 to 20). Do not specify a value of $k_{max.}$ greater than the number of dimensions found during factoring. The default value is **3** (*blank*).

19-20 Enter the minimal number of "new" variables (i.e., variables not included in previous subspaces) that a subspace must have for it to be printed. Subspaces defined by less than this minimal number of variables are deleted. The default number is **3** (*blank*).

[1] This option will override certain other options that may be chosen later (below).

21-28 Enter the lowest proportion of communality of a variable that will admit it to a subspace. A variable is "projected" into a subspace only if the proportion of its communality exhausted by the subspace is greater than or equal to the value you specify. The default value is **.80** (*blank*).

29-35 Enter $h^2_{min.}$, a communality lower bound, below which a variable is marked for potential exclusion from a SPAN plot. Variables with communalities below $h^2_{min.}$ are marked by an asterisk in the "spotter" table that follows each plot. The default value is **.10** (*blank*).

40 Indicate how you wish the spherical plots (3-spaces) to be centered ("physically rotated"):

Blank (or **0**), SPAN2 centers the configuration on the centroid of *all* salient variables (i.e., variables marked by either a '+' or a '-' sign in the SPAN plots and in the table of augmented coordinates for the 3-space.

1, SPAN2 centers the configuration on the centroid of just the *dimension-definers* among the salient variables.

❑ *If you will be selecting some or all of the subspaces yourself,* do so on one or more following lines. For each subspace desired, enter on a line the ordinal numbers for the dimensions that define it (in I4 format). Dimensions can be given in any order desired. For 3-spaces the first, second, and third numbers (dimensions) specified will be for the Z, X, and Y dimensions respectively. Enter a *blank line* to indicate the end of the list of desired subspaces (see example 3).

Example 1:

```
*SPAN2
  blank line
```

- All the default options are chosen. Only program-selected 3-dimensional subspaces are calculated and printed.

Example 2:

```
....+....1....+....2....+....3....+....4

*SPAN2
                   5.75     .15        1
```

- For subspaces to be printed, they must be defined by at least 5 "new" variables.
- Variables are projected into a subspace when at least 75% of their communality is exhausted by that subspace.
- The communality lower bound parameter is set to .15.
- The spherical plots are centered using dimension-definers only.

Example 3:

```
....+....1....+....2

*SPAN2
   3               4
   1   3   5
   6   4   2
  blank line
```

- Three user-specified subspaces will be printed in addition to those selected by SPAN2 itself.
- For SPAN2-selected subspaces to be printed, they must be defined by at least 4 "new" variables.

I/O

Print Output: SPAN2 always writes the following for each analysis: a title page, including the analysis title; a list of the parameter values chosen; a list of subspaces deleted (because of too few "new" salient variables); the SPAN plots; the percent of communality, and partial communalities, for all variables in every subspace selected (in summary tables at the end of the output listing). In addition, a table of augmented factor coefficients (coordinates) is printed for every subspace, regardless of its dimensionality. In these tables, salient variables are indicated by either a '+' or a '-' sign (for reflected variables).

For 3-dimensional subspaces (the default), variables are plotted on the surface of a unit sphere, using the coordinates printed in the tables. The entire configuration is rigidly rotated to center the variables on the spherical surface. The SPAN diagrams are the result of projecting the 3-dimensional spherical surface onto the 2-dimensional surface of a page. The name and line location of each salient variable is given in the "spotter" tables that follow the diagrams. Note that SPAN diagrams are printed only for 3-dimensional subspaces.

ISF Input: SPAN2 reads the following files from the ISF:

IDFILE	analysis identification
VNAMS1	variable names
UFACT1	unrotated factor coefficients

ISF Output: SPAN2 writes no files to the ISF.

www.trysys.info

SLEEP: Program to Suppress Variables

Introduction

Two very useful TRYSYS programs are SLEEP, used to suppress selected variables, and RESTORE, used to reactivate these variables at a later point in an analysis. These programs are used *after* the full correlation matrix of all variables has been calculated by COR2 or COR3.

There are various reasons why you might wish to suppress certain variables before factoring (and before communalities are calculated by DVP). One of the more common reasons is the need to delete linearly (or experimentally) dependent variables. Such variables are likely to generate communalities greater than one during factoring (so-called "Heywood cases") and lead to spurious results (e.g., see Gorsuch (2015) and Comrey and Lee (2015)).

These programs can also be used to perform what is known in the factor analysis literature as "extension analysis" (see Gorsuch, 1983, pages 236-238 and 300-302). For example, data sets often contain some demographic, exogenous, criterion, or other to-be-predicted variables in addition to the main set of variables. It is usually desirable to exclude these variables from the factoring process. This is easily done with SLEEP. RESTORE can then be used to reactivate the suppressed variables after factoring and to compute all factor statistics for these reactivated variables.

Method

The full correlation matrix is reordered by shifting the suppressed variables to the end rows and columns of the matrix. Only the submatrix containing the unsuppressed variables is made available to the factoring programs.

Use

The command *****SLEEP** calls the program.

❑ On the following line (or lines) specify the ordinal numbers or names of the variables to be suppressed. Variable numbers/names must be separated by commas, with a period after the last one. Do not break a name between lines and do not go beyond column 80.

I/O

Output Listing: SLEEP prints a list of the new and old locations of the variables in the correlation matrix. It also prints the suppressed and non-suppressed portions of the matrix.

ISF Input: SLEEP reads the following files from the ISF:

IDFILE	analysis identification
VNAMS1	variable names
CORRM1	correlations

ISF Output: SLEEP writes the following files to the ISF:

VNAMS1	reordered variable names
CORRM1	correlations among non-suppressed variables
REORD1	list of new variable locations
CORRM2	correlations among *all* variables

www.trysys.info

RESTORE: Program to Reactivate Suppressed Variables

A discussion of the analytical uses of the SLEEP and RESTORE programs is outlined in the SLEEP section. Note that you are not *required* to reactivate variables that have been suppressed. Variables may remain suppressed until there is a need to reactivate them. SLEEP and RESTORE may be used as many times as needed during a series of analyses (e.g., if you wish to conduct a series of exploratory analyses using different subsets of variables). However, if you need to change your list of suppressed variables, you must first reactivate all variables by calling RESTORE and then specify your new list of variables to be suppressed by calling SLEEP again.

Method

Once factoring has been performed on the submatrix produced by SLEEP, RESTORE can be used to calculate factor coefficients and communalities for all of the suppressed variables. When RESTORE is called, it reconstitutes the full correlation matrix in its originally ordered form and writes this to the ISF file CORRM1.

Use

The only command required is ***RESTORE** .

I/O

Print Output: RESTORE prints the factor coefficients, communalities, and other statistics of all variables using their SLEEP *reordered* positions. The factor coefficients for the previously suppressed variables are printed at the bottom of the orthogonal, unrotated factor loading matrix. It is thus easy to see what portion of their common factor variance is accounted for by the dimensions derived from factoring of the unsuppressed variables.

ISF Input: RESTORE reads the following files from the ISF (as they are available):

IDFILE	analysis identification
VNAMS1	reordered variable names
CORRM1	correlations among non-suppressed variables
REORD1	list of new variable locations
CORRM2	correlations among *all* variables
UFACT1	unrotated factor coefficients
CLUST1	*cluster indicators*
REFLX1	*reflection indicators*

ISF Output: RESTORE writes the following files to the ISF:

VNAMS1	variable names in their original order
CORRM1	correlations in their original order
DIAGV1	diagonal values for all variables
UFACT1	unrotated factor coefficients for all variables
CLUST1	*cluster indicators based on original order*
REFLX1	*reflection indicators based on original order*

www.trysys.info

COMP1: Program to Write Factor Matrices

Introduction

In preparation.

Method

In preparation.

Use

The only command required is ***COMP1**.

I/O

Printed output: COMP1 produces no printed output. Its only output is a file of factor coefficients and other information arranged in a format readable by the COMP2 program.

ISF input: COMP1 follows CSA2 and reads the following files from the ISF:

IDFILE	analysis identification
VNAMS1	variable names
RFACT1	rotated factor coefficients
CLUST1	cluster indicators

ISF output: COMP2 writes no files to the ISF.

www.trysys.info

COMP2: Program to Calculate Factor Structure Similarities

Introduction

In preparation.

Method

In preparation.

Use

The command ***COMP2** calls the program.

❑ On the following line enter an analysis title of up to 72 characters. This title is stored in the ISF and is printed by many of the component programs.

❑ On a second line, specify the number of groups of data to be analyzed, the variables (adjoined matrix rows) you wish to use, the dimensions (adjoined matrix columns) you wish to use, and the similarity index desired.

cols.

3-4 Enter the number of groups of data to be analyzed.

8 Indicate how you wish the variables (rows) of the adjoined matrix to be selected:

Blank (or **2**), use those rows corresponding to the defining variables of the key clusters.

0, use *all* the rows.

1, use the rows specified below on one or more following lines (see example 2).

12 Indicate how you wish the dimensions (columns) of the adjoined matrix to be selected.

Blank (or **0**), use *all* the columns.

1, use the columns specified on one or more following lines (see below, and example 2)

16 Select the similarity index to be used in the comparison:

Blank (or **0**), use the cosine method.

1, use the signed P^2 method.

2, use the correlation coefficient method.

- If you wish to select specific variables (rows) to be used in the analysis (1 in col. 8), enter the variable names on one or more lines here. These names must be separated by commas and may not be broken between lines. Also, the list of names must be terminated by a period.

- If you wish to select specific dimensions (columns) to be used in the analysis (1 in col. 12), enter the dimension names on one or more lines. These dimension names must be of the form G*nn*D*kk* (e.g., G03D08 indicates the eighth dimension of the third group). Use the same rules for listing the dimension names that are used for listing variable names.

- Use your editor to insert here the COMP1 files to be used in your comparative analysis. If you select specific dimensions (columns) to be used, be sure to insert the COMP1 files in the appropriate order.

Example 1:

```
....+....1

*START
*COMP2
Comparison of SCL-90 Factor Structure for Male vs. Female Students.
   2
DATA FILE PREPARED BY COMP1 FOR COMP2
Preset SCL-90 Factoring Results for Male Students.
.
.
.
.
.
DATA FILE PREPARED BY COMP1 FOR COMP2
Preset SCL-90 Factoring Results for Female Students.
.
.
.
.
.
*GIVE
*END
```

- Inter-factor similarities (cosines) for *two* groups are calculated and saved.
- All the default COMP2 options are chosen.

ISFDMP: Program to Print ISF File Contents

Introduction

ISFDMP provides a readable listing of the contents of ISF system files. To save space, the information is provided in a condensed format. Each file in the ISF (e.g., variable names, correlations, diagonal values, factor loadings, etc.) is identified by its six-character name (see the ISF section).

Most users will probably never have occasion to use ISFDMP. However, the program would be of value if you were uncertain of the contents of an ISF file saved by an earlier *GIVE command. In general, ISFDMP is most usefully called at the end of a run or just after a *TAKE command.

Use

The only command required is ***ISFDMP** .

Example:

```
*START
*TAKE
*ISFDMP
*END
```

▪ A previously saved ISF file is retrieved by TAKE, and its contents are printed by ISFDMP.

I/O

ISF Input: ISFDMP reads all files that exist on the ISF at the time it is called.

ISF Output: ISFDMP writes no files to the ISF.

www.trysys.info

RLIST: Program to Print Rearranged COR3 Statistics

Introduction

RLIST provides a rearranged printout of values computed by COR3. Although COR3 provides a printout of values it computes, each type of statistic is printed in a separate matrix. As a convenience to the user with missing data, RLIST prints on one line all statistics for each pair of variables. Printed along with each correlation coefficient are: the names of the variables, the matched N, the matched mean, and the matched standard deviation. You should use caution in calling RLIST because its output can run to many dozens of pages if the number of variables is large.

Use

The only command required is ***RLIST** .

I/O

ISF input: RLIST follows COR3 and reads the following files from the ISF (as they are available):

IDFILE	analysis identification
VNAMS1	variables names
VAREN1	sample sizes (for each variable)
CORRM1	correlations
2SENS1	matched Ns
2MEAN1	matched means
2STDV1	matched standard deviations

ISF output: RLIST writes no files to the ISF.

GYRO: Factor Rotation Program[1]

Introduction

GYRO performs two commonly available techniques of *orthogonal* factor rotation, **Varimax** and **Quartimax**. There are advantages and disadvantages to the use of each of these rotational methods (see Gorsuch, 1983, pages 183-186). In the absence of definitive guidelines, you should consider performing *both* rotations after factoring, calling SPAN2 after each, and choosing the rotational method that produces the most visually interpretable spherical analysis results.

In *Cluster Analysis*, Tryon and Bailey (pages 126-133) illustrate how the structural configuration of the 24 Holzinger ability variables remains essentially invariant across different methods of factoring and different methods of rotation (see also Gorsuch, 1983, pages 197-206).

Method

The Quartimax and Varimax programs of GYRO are modified versions of the published programs of Cooley and Lohnes (1962).

Use

The command ***GYRO** calls the program.

- ❑ On the following line, enter **QRTMAX** in cols. 1-6 for a Quartimax rotation and **VARMAX** for a Varimax rotation.
- ❑ On a second line, enter in cols. 3-4 the maximum number of iterations to be performed. If this line is left *blank*, the maximum number of iterations is 10.
- ❑ On a third line, enter in cols. 1-8 the convergence criterion for iteration termination. If this line is left *blank*, the default values are used (.001 for Quartimax and .0000001 for Varimax).

Note that you do not need to repeat factoring each time you want to use GYRO. The GIST procedure writes the unrotated factor coefficients to the ISF, and these values remain there, unaltered.

Example 1:

```
*GYRO
VARMAX
  blank line
  blank line
```

- The Varimax method of rotation is requested.
- Default values for the number of iterations and for the convergence criterion are utilized.

Example 2:

```
*GYRO
QRTMAX
   8
.005
```

- The Quartimax method is requested.
- The maximum number of iterations desired is 8, and the convergence criterion chosen is .005.

I/O

Printed Output: GYRO prints a description of the method and the parameters chosen, a summary of the iteration process, and tables of results.

ISF Input: GYRO reads the following files from the ISF:

IDFILE	analysis identification
UFACT1	unrotated factor coefficients

ISF Output: GYRO writes the following files to the ISF:

RFACT1	rotated factor coefficients (orthogonal)
BASIS1	identity matrix

References

Bailey, D.E., and Cartwright, D.S. (1979). The BCTRY integrated statistical computer package. *Behavior Research Methods and Instrumentation*, **11**, 407-413.

Cooley, W., and Lohnes, P. (1962). *Multivariate procedures for the behavioral sciences.* New York: John Wiley.

Gorsuch, R.L. (1983). *Factor analysis*. Hillsdale, N.J.: Lawrence Erlbaum.

Tryon, R.C. (1957a). Reliability and behavior domain validity: Reformulation and historical critique. *Psychological Bulletin*, **54**, 229-249.

Tryon, R.C. (1957b). Communality of a variable: Formulation from cluster analysis. *Psychometrika*, **22**, 241-259.

Tryon, R.C. (1958a). Cumulative communality cluster analysis. *Educational and Psychological Measurement*, **18**, 3-35.

Tryon, R.C. (1958b). General dimensions of individual differences: Cluster analysis versus factor analysis. *Educational and Psychological Measurement*, **18**, 477-495.

Tryon, R.C. (1959). Domain sampling formulation of cluster and factor analysis. *Psychometrika*, **24**, 113-135.

Tryon, R.C. (1966). Unrestricted cluster and factor analysis with applications to the MMPI and Holzinger-Harmon problems. *Multivariate Behavioral Research*, **1**, 229-244.

Tryon, R.C. (1967a). Person-clusters on intellectual abilities and MMPI attributes. *Multivariate Behavioral Research*, **2**, 5-34.

Tryon, R.C. (1967b). Predicting individual differences in cluster analysis: Holzinger abilities and MMPI attributes. *Multivariate Behavioral Research*, **2**, 325-348.

Tryon, R.C. (1967c). Predicting group differences in cluster analysis: The social area problem. *Multivariate Behavioral Research*, **2**, 453-475.

Tryon, R.C. (1968a). Comparative cluster analysis of variables and individuals: Holzinger abilities and the MMPI. *Multivariate Behavioral Research*, **3**, 115-144.

Tryon, R.C. (1968b). Comparative cluster analysis of groups: The social area problem. *Multivariate Behavioral Research*, **3**, 213-282.

Tryon, R.C. (1973a). Basic unpredictability of individual responses to discrete stimulus presentations. *Multivariate Behavioral Research*, **8**, 275-295.

Tryon, R.C. (1973b). Generality and reliability of individual differences in discrete stimulus-item situations. *Multivariate Behavioral Research*, **8**, 297-311.

Tryon, R.C., and Bailey, D.E. (1966). The BCTRY computer system of cluster and factor analysis. *Multivariate Behavioral Research*, **1**, 95-111.

Tryon, R.C., and Bailey, D.E. (1970). *Cluster analysis*. New York: McGraw-Hill