Res. Lett. Inf. Math. Sci., 2005, Vol.**8**, pp143-154
Available online at http://iims.massey.ac.nz/research/letters/

Leaving useful traces when working with matrices

Kimmo Vehkalahti

Department of Mathematics and Statistics
University of Helsinki, Finland

We consider the documentation of the working process in the context of matrix computations and multivariate statistical analyses. Our focus is on the way of working: how well does the working process get documented dynamically, or what sort of *traces* are left behind. These questions are relevant in any area of research. Leaving useful traces while working may save a considerable amount of time, and provide better possibilities for other researchers to comprehend the points of a study. These principles are demonstrated with examples using Survo software and its matrix interpreter.

1 Introduction

Documentation is a critical part of the working process in any area of research. The results of a study are not enough. We should also consider and answer the following questions: How did we achieve the results? How did we justify the analyses? How did we correct the errors? Why did we take a particular step? How do we repeat the steps?

Errors in the data or in the working process can not be avoided, since they will occur in each research project. Therefore it should be a routine to correct any error when detected, document the correction, make necessary updates, and continue working without disturbing the thinking. A good test for any researcher and any software environment would be to find out how easily the errors are detected, and how quickly the results are reproduced after the errors have been corrected.

In general, we should pay more attention to the quality of the working process: How does the work proceed? How well does it get documented? Since our interest is mainly in the *dynamic documentation*, we would like to ask: What sort of *traces* are left behind while working? Can we retrace our steps efficiently, as we take steps forwards and backwards, perhaps following some side tracks?

2 Leaving traces—what do we get?

The traces represent saved ideas and documented thoughts that are processed while working. Leaving useful traces may save a considerable amount of time. It supports the researcher in backtracking immediately when certain steps of the process have to be repeated. Useful traces help to avoid re-inventing the wheel, and to hold the process together, especially when managing multiple projects simultaneously. They may also provide better possibilities for other researchers to comprehend the points of a study.

Gentleman and Temple Lang [7] have introduced the term *compendium* to refer to documents that are self-contained mixtures of code, text, and data. Publishing compendiums instead of traditional scientific papers allows the readers to verify what exactly have been done and how. A proper documentation of the working process published together with accompanying software tools may encourage *reproducible research*, meaning that the reader can directly reproduce the results employing the methods that are presented in [7].

Regardless of whether the results are published as a compendium or as a traditional scientific publication, it is important for the researcher to leave useful traces when working with research problems. For example, it may be quite demanding to trace one's own thoughts and ideas in the middle of a review process of a paper, perhaps several months after submitting the manuscript to a journal. With a good documentation it is easier to get back on the track.

3 Software environment—what do we need?

The working process depends heavily on the software environment, which may consist of several different software packages with varying user interfaces. The support from the software environment is needed in the working process including the documentation. This applies for large scale tasks, but also for small details, which tend to appear repeatedly during the working process.

In general, we should have flexible tools for saving the ideas and thoughts while working. It is preferable to have possibilities for writing free notes and comments nearby the actual operations that are committed. In this sense, the menu-driven environments usually do not provide enough means for dynamic documentation.

There should be detailed mechanisms to access the data in various situations. Interactive display of the data is useful for browsing and searching, but the errors should be corrected in a way that also documents the corrections. Otherwise the corrections could be forgotten, when certain steps of the process have to be repeated. A typical situation might be that the original data are modified or even replaced, and the operations using the data are repeated. Manual corrections could be difficult and frustrating to repeat. They would slow down the working process and probably also introduce new errors.

Automatic documentation means that the software environment leaves useful

traces without a continuous need to document the working process manually. For example, it is necessary to have good means of naming different objects, such as variables in the data, or row and column labels in the matrices. However, cooperation between the environment and the user is needed, since it is also required that the user actually gives describing names. If the environment provides the means, and the user takes advantage of them, it is possible to achieve automatic documentation in subsequent operations.

Another way to support automatic documentation is that the operations of the environment are self-documenting as such. Writing too many comments or notes to explain the steps of the working process is not reasonable. It could even be worse than no comments, especially if the code or data are modified but the comments are not updated accordingly.

4 Introduction to Survo environment

In this paper, we display examples of working processes and documentation in *Survo* environment, created in [18]. The approach of Survo differs considerably from the main-stream software packages. Despite of its long history beginning from the 1960s, Survo—at least its recent development—is not widely known. Therefore we first review the fundamental features of Survo.

The current version of Survo has emerged in several phases from a statistical programming language SURVO 66 [1] leading to the creation of SURVO 76, one of the first interactive statistical programs [10] and further, via SURVO 84 [13] to SURVO MM, a general environment for creating and processing text and numerical data [21].

Survo supports the documentation of the working process by its *editorial interface* [11, 12, 14], which replaced the menu-based interface of SURVO 76 in 1979. Editorial interface is a distinguishing construct, in which a special text editor takes care of both input and output of operations related to statistical analysis, matrix and other computations, according to the needs of the user. The operations, expressions, and links activated within the text, possibly producing output back in the editor, can be commented inherently in free format. The outputs in turn can be edited, formatted, highlighted, commented, printed, and used as input for other operations.

The `PRINT` operation of Survo takes advantage of the editorial interface by user-definable keywords, control characters, and so-called shadow characters, and gives its output in PostScript format with CMYK colors [15]. The same technique of printing multipage documents on paper is used to create documents in any structural language, such as LaTeX or HTML. The editorial interface of Survo also enables fruitful cooperation with other programs, such as R [23]. There is even a specialized, user-friendly frontend, which combines the best properties of R and Survo.

In addition to the editorial interface, SURVO MM includes selected elements of the graphical user interface. For example, the statistical and other graphs are

by default plotted as metafiles and displayed in their own graphics windows. The rotations in factor analysis can be performed in an interactive graphics window as a sequence of two-dimensional rotations.

The matrix operations of Survo are carried out by a matrix interpreter, which is closely integrated in the statistical, graphical, and other operations of the system through the editorial interface and the data and matrix structures [18, pp. 360–392]. Features of the matrix interpreter, such as automatically inherited row and column labels of the matrix objects, support the documentation of the working process.

Mustonen [20] has demonstrated matrix computations in Survo, beginning from elementary operations and extending to advanced applications and programming. The data sets and *sucros* (or Survo macros [16]) utilized in the paper are included in the distribution package of Survo. Obviously, this idea is equivalent to the principles that have been recently termed reproducible research [7], since the accompanying data sets and sucros allow the user to directly (or even automatically) reproduce the results and employ the methods that are presented in the paper. The user may also freely modify the premises and make alternative analyses and simulations using the ready-made working schemes provided in the saved Survo jobs.

Reproducibility does not depend on whether we are working with matrices or not, as the common factor of all activities in Survo is the editorial interface. Working in Survo typically results in saved jobs or documents that not only give detailed descriptions of the working processes but also facilitate repeating the processes more or less automatically. Hence, it can be said that the idea of reproducible research has been a regular way of working in Survo since the invention of the editorial interface in 1979 [11].

The core structures of Survo have not been modified since 1985, when SURVO 84C, the first version of Survo programmed in C language, was released. Its programming libraries, including a set of matrix functions have been described in [17] with the details of the data and matrix structures. The libraries are updated synchronously along the system development, and they are freely available. Survo is a modular system which can be extended without limits [17].

The interaction between the community of Survo users and the developers has been rewarding both for the users and the developers [18, p. iii–iv]. The developers appreciate the feedback and suggestions, while the users appreciate the continuous development process. New versions of SURVO MM are released frequently, bringing new functions and operations available. However, because of the continuous development process, the concept of version is not very significant. According to one of the major development principles of Survo, new functions are added to the system in a way that the existing ones are not changed, since that could potentially cause harm for any user-defined applications relying on a particular function or its output. By adding new options, the functions can still be extended without limits and without disturbing existing usage.

Applying the above principle has guaranteed a full backward compatibility between the versions of Survo based on the editorial interface [21]. Therefore, the user

never needs to re-learn how to use Survo. The interface is the same, although the technical environment, such as the operating system has changed and might change again. In our terms, Survo is an environment where the traces left behind can be retraced also in the future.

A traditional research paper can not give a true picture of any dynamic contents, but hopefully the following examples will provide an idea of the documentation possibilities in the Survo environment.

5 Factor analysis and false assumptions

In the following we demonstrate some practices of leaving useful traces when working with matrices and certain multivariate statistical analyses using Survo, especially its matrix interpreter.

5.1 Background of misconceptions

We are going to revisit one of the experiments on factor analysis carried out in [5, 6] during the 1970s. Francis claimed factor analysis to be "the most misunderstood and misapplied statistical procedure" [6, p. 9]. Francis' findings are summarized by Seber in [24, pp. 222–235] in his classic book *Multivariate Observations*. Before turning to a particular example, we briefly review some historical background and facts on factor analysis.

Factor analysis was originally invented in 1904 by Spearman [25], and developed further both by psychologists and statisticians. It has been said that "factor analysis was born before its time, and had to mark time until the technology caught up" [3, pp. 216–217]. At the time of Francis' experiments, factor analysis was commonly criticized by statisticians, although Anderson and Rubin [2] had established the statistical properties of the method, and Lawley and Maxwell [8] showed factor analysis to be a statistical method with a proper maximum likelihood estimation.

The mistrust in factor analysis has persistently been present in the statistical literature. In their introductory book of multivariate analysis, Chatfield and Collins [4, pp. 88–89] have given a list of drawbacks of the method recommending that "factor analysis should not be used in most practical situations". However, this mistrust is mainly caused by misconceptions.

Mustonen [19, pp. 106–112] has examined Francis' experiments and discussed Seber's conclusions, one of them being: "In conclusion, it must be stated that if factor analysis is carried out, then the results must be interpreted with extreme caution." [24, p. 235]. But, as Mustonen [19, p. 106] reminds, this applies similarly to results of *any* statistical method.

Seber [24, p. 235] continues: "Even if the postulated model is true—and this is a very strong assumption—the chance of its recovery by present methods does not seem very great." This conclusion leaves room for further thoughts. The problem is that the conclusion is based on false assumptions: Francis had believed that

the given factor pattern would represent a *simple structure* [28] which should be reproduced by factor analysis. However, the structure is not simple in this sense, which makes it quite impossible to reproduce by standard methods [19, p. 110].

5.2 Francis' experiment revisited

One of the original factor patterns of Francis [5] is given here as a matrix consisting of loadings of three common factors `F1`, `F2`, and `F3`, and the standard deviations of the unique factors `PSI`:

```
MATRIX FRANCISV
Factor matrix from Francis (1973), quoted by Seber (1984).
MODEL_V
///   F1    F2    F3   PSI
X1    10     7     4    15
X2    10     7     4    15
X3    10     7     4    15
X4    10     7     4    15
X5    10     7     0    15
X6    10     7     0    20
X7    10     7     0    20
X8    10     0     0    20
X9    10     0     0    20
X10   10     0     0    20
```

The matrix `FRANCISV` is written in the Survo editor. The meta information and documentation around the numerical elements consist of 1) general comments, including the internal name of the matrix, 2) row labels, and 3) column labels. The structure would also allow rowwise comments, which are typically used in vectors giving some scalar results e.g.in regression analysis. All information is included in the matrix object when it is saved as a matrix file `FRANCISV.MAT` in the current working directory. In subsequent operations, the matrix is referred to by using this external name. The internal name of the matrix will be updated automatically by each matrix operation.

By the following commands the matrix `FRANCISV` is saved, and two sub-matrices (`F0` and `PSI`) are extracted. Comments may be written all around the commands, even on the same line when separated with an isolated slash. They are ignored by the interpreter (the `MAT` commands), but they might be valuable for the user in retracing the steps of the work in the future.

```
MAT SAVE FRANCISV
MAT F0!=FRANCISV(*,F1:F3)    / F0 = original factor matrix (the loadings)
MAT PSI!=DV(FRANCISV(*,PSI)) / Diagonal matrix of the column Vector PSI
```

Next, the extracted matrices are used to form the variance–covariance matrix of the variables according to the basic equation of factor analysis. Scaling by the standard deviations forms the corresponding correlation matrix and the scaled factor matrix:

```
MAT S=F0*F0'+PSI^2           / S = covariance matrix
MAT D=DIAG(S)^(-0.5)
MAT R=D*S*D                  / R = correlation matrix
MAT F=D*F0                   / F = scaled factor matrix
```

We take a look at the resulting factor matrix using the sucro `SUM2`, which computes the sums of squares by rows and columns of the matrix given as the first parameter. The second parameter `##.###` gives the desired precision of the output:

```
/SUM2 F ##.###
```

```
MATRIX SUM2
F_with_sums_of_squares_by_rows_and_columns
///          F1     F2     F3 Sumsqr
X1        0.506  0.354  0.203  0.423
X2        0.506  0.354  0.203  0.423
X3        0.506  0.354  0.203  0.423
X4        0.506  0.354  0.203  0.423
X5        0.517  0.362  0.000  0.398
X6        0.427  0.299  0.000  0.271
X7        0.427  0.299  0.000  0.271
X8        0.447  0.000  0.000  0.200
X9        0.447  0.000  0.000  0.200
X10       0.447  0.000  0.000  0.200
Sumsqr    2.257  0.812  0.164  3.234
```

The columnwise sums of the squares of the factor loadings reveal that the three-dimensional factor structure is questionable. At least the third factor is very weak. The communalities (the rowwise sums of the squares of the loadings) are also poor. All the largest loadings are on the first factor.

It can be inferred that Francis' factor pattern does not conform to the requirements of the simple structure [5, 28], and thus it is a bit useless to continue further. However, for the sake of demonstration, we continue and follow the steps of Francis [5] trying to reproduce the result by factor analysis.

First, we examine the correlation matrix `R` by loading it in the edit field. The parameter `CUR+1` gives the beginning line for the output, relative to the line that is activated:

```
MAT LOAD R #.### CUR+1
MATRIX R
DIAG(F0*F0'+PSI^2)^(-0.5)*(F0*F0'+PSI^2)*DIAG(F0*F0'+PSI^2)^(-0.5)
```

```
///         X1    X2    X3    X4    X5    X6    X7    X8    X9   X10
X1       1.000 0.423 0.423 0.423 0.390 0.322 0.322 0.226 0.226 0.226
X2       0.423 1.000 0.423 0.423 0.390 0.322 0.322 0.226 0.226 0.226
X3       0.423 0.423 1.000 0.423 0.390 0.322 0.322 0.226 0.226 0.226
X4       0.423 0.423 0.423 1.000 0.390 0.322 0.322 0.226 0.226 0.226
X5       0.390 0.390 0.390 0.390 1.000 0.329 0.329 0.231 0.231 0.231
X6       0.322 0.322 0.322 0.322 0.329 1.000 0.271 0.191 0.191 0.191
X7       0.322 0.322 0.322 0.322 0.329 0.271 1.000 0.191 0.191 0.191
X8       0.226 0.226 0.226 0.226 0.231 0.191 0.191 1.000 0.200 0.200
X9       0.226 0.226 0.226 0.226 0.231 0.191 0.191 0.200 1.000 0.200
X10      0.226 0.226 0.226 0.226 0.231 0.191 0.191 0.200 0.200 1.000
```

The internal name of the matrix tells about the history of the matrix. Here, it shows exactly how the matrix `R` was created from `F0` and `PSI`. As a side result of those operations, the row labels of the factor matrix are inherited to both row and column labels of the correlation matrix.

We compute the spectral decomposition of the matrix `R` to investigate the eigenvalues. They are saved as a column vector `L`, which we load in the edit field in transposed form:

```
MAT SPECTRAL DECOMPOSITION OF R TO S,L / (S includes the eigenvectors)
MAT LOAD L' #.### CUR+1
MATRIX L'
L(CORR)'
///        ev1   ev2   ev3   ev4   ev5   ev6   ev7   ev8   ev9  ev10
eigenval 3.632 0.956 0.800 0.800 0.732 0.729 0.622 0.577 0.577 0.577
```

The matrix `R` is evidently positive definite, but the dimensionality is difficult to infer, although the first eigenvalue is clearly greater than the others. Following the experiments of Francis [5], we compute the maximum likelihood factor analysis of three factors, using the above correlation matrix, which corresponds to using an infinite sample size.

The matrix `R` is used as input to the `FACTA` module, which gives its output only in the matrix file `FACT.M`, since no line is given the command. The second parameter gives the number of factors. We check the output by `/SUM2`, and notice again that the only reasonable factor is the first one:

```
FACTA R,3 / default method is maximum likelihood
/SUM2 FACT.M ##.###

MATRIX SUM2
FACT.M_with_sums_of_squares_by_rows_and_columns
///          F1     F2     F3 Sumsqr
X1        0.643 -0.080 -0.051  0.423
X2        0.643 -0.080 -0.051  0.423
```

```
X3        0.643 -0.080 -0.051  0.423
X4        0.643 -0.080 -0.051  0.423
X5        0.618  0.010  0.129  0.398
X6        0.510  0.008  0.106  0.271
X7        0.510  0.008  0.106  0.271
X8        0.378  0.236 -0.037  0.200
X9        0.378  0.236 -0.037  0.200
X10       0.378  0.236 -0.037  0.200
Sumsqr    2.987  0.192  0.054  3.234
```

The factor structure may seem as more complicated than the original, but a suitable rotation to make the structures identical could be found by using *symmetric transformation analysis* in [9], a practical method for comparison of factor structures based on singular value decomposition ([19, pp. 95–105]). Francis' experiments have also been studied by Mustonen and Vehkalahti [22], with the focus on simulation and rotation methods.

5.3 Assessing structural validity

Finally, we would like to introduce an additional point of view to the same problem. It is based on working with measurement models and measurement scales.

A general framework for modeling the measurement and assessing the quality of multivariate measurement scales was introduced in [27, 29, 26]. In the following, we examine the *structural validity* of the factor model in Francis' [5] experiments.

We begin by giving the matrices names that correspond with the concepts of the measurement framework:

```
MAT B=FACT.M       / new name for the ML factor matrix computed above
MAT NAME B AS ML   / change also the internal name to "ML"
MAT A=B            / factor images (scales corresponding to the factors)
MAT NAME R AS CORR / shorten the name (wiping out the previous history)
```

Now, the matrix `B` represents $\boldsymbol{B} \in \mathbb{R}^{10\times 3}$, which specifies the relationship between the measured variables $\boldsymbol{x} = (x_1, \ldots, x_{10})'$ and the *true scores* (or factors) $\boldsymbol{\tau} = (\tau_1, \tau_2, \tau_3)'$ [27, Eq. 2.1].

Correspondingly, the matrix `A` represents $\boldsymbol{A} \in \mathbb{R}^{10\times 3}$, which gives the weights for the measurement scales $\boldsymbol{u} = \boldsymbol{A}'\boldsymbol{x}$ [27, Eq. 2.3]. The particular scales with $\boldsymbol{A} = \boldsymbol{B}$ are called *factor images* [27, Dfn. 2].

The structural validity of the model can be assessed by the *reliabilities* of the factor images. In general form, the reliability matrix is obtained by $\boldsymbol{\rho_u} = diag(\boldsymbol{A}'\boldsymbol{B}\boldsymbol{\Phi}\boldsymbol{B}'\boldsymbol{A})\times [diag(\boldsymbol{A}'\boldsymbol{\Sigma}\boldsymbol{A})]^{-1}$, where $\boldsymbol{\Phi} = \mathrm{Cov}(\boldsymbol{\tau})$ and $\boldsymbol{\Sigma} = \mathrm{Cov}(\boldsymbol{x})$ [27, Eq. 2.11]. Here, we have scaled the variables and use the correlations instead of covariances. As the factors are orthogonal, $\boldsymbol{\Phi} = \boldsymbol{I}$, an identity matrix. We are now ready to compute the reliability matrix:

```
MAT RHO=DIAG(A'*B*B'*A)*INV(DIAG(A'*R*A)) / reliability matrix (diagonal)
MAT RHO=VD(RHO)' / make a row vector of the diagonal elements
MAT LOAD RHO ##.### CUR+2

MATRIX RHO
VD(DIAG(ML'*ML*ML'*ML)*INV(DIAG(ML'*CORR*ML)))'
///           F1     F2     F3
diag       0.824  0.225  0.081
```

Not surprisingly, the reliabilities of the factor images `F2` and `F3` indicate a serious lack of the structural validity of the model. The factors `F2` and `F3` are totally artificial, and the correct number of factors would obviously be one.

Again the row and column labels are inherited and the internal name of the matrix updated automatically according to the preceding matrices and operations. Hence, the matrix interpreter of Survo leaves useful traces that help the user to keep on the track.

Acknowledgements

I would like to thank George Seber for a useful discussion on factor analysis during the 14th IWMS in Auckland, New Zealand. I am grateful to Seppo Mustonen and Simo Puntanen for their valuable comments and suggestions.

References

[1] Alanko, T., Mustonen, S., and Tienari, M. (1968). A statistical programming language SURVO 66. *BIT*, 8:69–85.

[2] Anderson, T. W. and Rubin, H. (1956). Statistical inference in factor analysis. In Neyman, J., editor, *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability*, volume V, pages 111–150, Berkeley and Los Angeles. University of California Press.

[3] Bartholomew, D. J. (1995). Spearman and the origin and development of factor analysis. *The British Journal of Mathematical and Statistical Psychology*, 48:211–220.

[4] Chatfield, C. and Collins, A. J. (1980). *Introduction to Multivariate Analysis*. Chapman and Hall, London.

[5] Francis, I. (1973). Factor analysis: Its purpose, practice, and packaged programs. Invited paper, American Statistical Association, New York.

[6] Francis, I. (1974). Factor analysis: Fact or fabrication. *Mathematical Chronicle*, 3:9–44.

[7] Gentleman, R. and Lang, D. T. (2004). Statistical analyses and reproducible research. Bioconductor Project Working Papers, Working Paper 2, `http://www.bepress.com/bioconductor/paper2`.

[8] Lawley, D. N. and Maxwell, A. E. (1971). *Factor Analysis as a Statistical Method*. Butterworth, London, second edition.

[9] Mustonen, S. (1966). Symmetrinen transformaatioanalyysi [Symmetric transformation analysis, in Finnish]. Report 24, Social Research Institute of Alcohol Studies, Helsinki.

[10] Mustonen, S. (1977). SURVO 76, a statistical data processing system. Research Report 6, Department of Statistics, University of Helsinki.

[11] Mustonen, S. (1980). SURVO 76 EDITOR, a new tool for interactive statistical computing, text and data management. Research Report 19, Department of Statistics, University of Helsinki.

[12] Mustonen, S. (1982). Statistical computing based on text editing. In Caussinus, H., Ettinger, P., and Tomassone, R., editors, *Proceedings in Computational Statistics*, pages 353–358, Wien. Physica-Verlag.

[13] Mustonen, S. (1984). SURVO 84 – interactive system for statistical computing, graphics and text processing. Research Report 51, Department of Statistics, University of Helsinki.

[14] Mustonen, S. (1987). Editorial approach in statistical computing. In Pukkila, T. and Puntanen, S., editors, *Proceedings of the Second International Tampere Conference in Statistics*, pages 205–224, Tampere, Finland. Department of Mathematical Sciences, University of Tampere.

[15] Mustonen, S. (1988a). Postscript printing in SURVO 84C. SURVO 84C Contributions 1, Department of Statistics, University of Helsinki.

[16] Mustonen, S. (1988b). Sucros in SURVO 84C. SURVO 84C Contributions 2, Department of Statistics, University of Helsinki.

[17] Mustonen, S. (1989). Programming SURVO 84 in C. SURVO 84C Contributions 3, Department of Statistics, University of Helsinki. `http://www.helsinki.fi/survo/c/`.

[18] Mustonen, S. (1992). *Survo, An Integrated Environment for Statistical Computing and Related Areas*. Survo Systems, Helsinki. ISBN 951-96634-0-1.

[19] Mustonen, S. (1995). *Tilastolliset monimuuttujamenetelmät* [Statistical Multivariate Methods, in Finnish]. Survo Systems, Helsinki.

[20] Mustonen, S. (1999). Matrix computations in Survo. Invited paper, The Eighth International Workshop on Matrices and Statistics, Tampere, Finland. http://www.helsinki.fi/survo/matrix99.html.

[21] Mustonen, S. (2001). New Windows version of Survo. http://www.survo.fi/mm/english.html.

[22] Mustonen, S. and Vehkalahti, K. (1997). Survo as an environment for statistical research and teaching. In *Bulletin of ISI*, number 2 in Proceedings, pages 69–72, Istanbul. International Statistical Institute. http://www.helsinki.fi/survo/isi97.html.

[23] R Development Core Team (2004). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

[24] Seber, G. A. F. (1984). *Multivariate Observations*. Wiley, New York.

[25] Spearman, C. (1904). General intelligence objectively determined and measured. *American Journal of Psychology*, 15:201–293.

[26] Tarkkonen, L. (1987). *On Reliability of Composite Scales: an Essay on the Structure of Measurement and the Properties of the Coefficients of Reliability – a Unified Approach*. Number 7 in Statistical Studies. Finnish Statistical Society, Helsinki. Doctoral dissertation, Department of Statistics, University of Helsinki.

[27] Tarkkonen, L. and Vehkalahti, K. (forthcoming). Measurement errors in multivariate measurement scales. *Journal of Multivariate Analysis*. Available online Nov 11 2004 at http://dx.doi.org/10.1016/j.jmva.2004.09.007.

[28] Thurstone, L. L. (1947). *Multiple-Factor Analysis*. The University of Chicago Press, Chicago.

[29] Vehkalahti, K. (2000). *Reliability of Measurement Scales: Tarkkonen's General Method Supersedes Cronbach's Alpha*. Number 17 in Statistical Research Reports. Finnish Statistical Society, Helsinki. Doctoral dissertation, Department of Statistics, University of Helsinki, http://ethesis.helsinki.fi/julkaisut/val/tilas/vk/vehkalahti/.