Max-Planck-Institut für demografische Forschung
Max Planck Institute for Demographic Research
Konrad-Zuse-Strasse 1 · D-18057 Rostock · GERMANY
Tel +49 (0) 3 81 20 81 - 0; Fax +49 (0) 3 81 20 81 - 202;
http://www.demogr.mpg.de

MPIDR TECHNICAL REPORT 2012-003
JULY 2012

compareFinRaw.r – an R program to measure the difference between datasets

Rainer Walke (walke@demogr.mpg.de)
Andreas Müller (amueller@demogr.mpg.de)

For additional material see www.demogr.mpg.de/tr/

This technical report has been approved for release by: Mikko Myrskylä (myrskyla@demogr.mpg.de), Head of the Max Planck Research Group on Lifecourse Dynamics and Demographic Change.

© Copyright is held by the authors.

Technical reports of the Max Planck Institute for Demographic Research receive only limited review. Views or opinions expressed in technical reports are attributable to the authors and do not necessarily reflect those of the Institute.

compareFinRaw.r – an R program to measure the difference between datasets

Rainer Walke and Andreas Müller
Max Planck Institute for Demographic Research, Rostock

2012-Jul-18

Abstract

In every data-related research it is essential to have knowledge about potential disparity of the data in use. There might be differences between modification stages of a single dataset or between distinct datasets. Either way the researcher has to be aware of these differences in order to draw proper conclusions that might be affected by different data properties. This report describes an adaptable solution to cope with that problem by using the statistical software `R` [R 2011]. The program `compareFinRaw.r` is a suitable automatic tool to measure differences of two datasets by computing distances for all relevant variable (column) pairs of the datasets on two levels. Two excerpts of the `R`-internal dataset `Seatbelts` [R 2011, Harvey 1986] serve as an illustrative data example.

Keywords: data analysis, data comparability, data evaluation, data processing, software

Background

The program which has also been successfully tested on large datasets automatically applies our algorithm to the example datasets. It mainly compares all nonconstant variables of one dataset with all nonconstant variables of the other dataset. This pairing is done case by case, so only the values belonging to the same case (observation) are being compared. Therefore two measures of distance are processed: on the one hand the difference in quantity of the unique elements for both variables against the number of their unique mappings ("mapping distance") and on the other hand case-wise Levenshtein distances [LevDis 2012]. For the latter, a 5-parts quantile section serves as means of classification. After having run the program, five comma-separated values (CSV) files and an additional `RData` file are returned to present the results. The `RData` file is also used as data source for an automatically generated report using `Sweave` [Leisch 2002]. A functioning `Sweave` installation and a Sweave file (`compareFinRaw_results.rnw`, see Appendix B) are required, a control file (`compareFinRaw_prep_results.r`) is optional. All outputs are in folder "Output_ready".

How to use it

The only restrictions concerning the structure of the two datasets are

- equal number of cases and
- a common variable serving as unique case identifier.

For reasons of simplicity, the datasets are called `Fin` (final) and `Raw` (raw or original). A new folder (“Results”) holding the resulting CSV and `RData` files is created in the current working directory. After import, the variable names of `Fin` get the suffix “F” and those of `Raw` “R” for better distinction. Afterwards both datasets are sorted to the unique case identifier (stating file names and definition of the identifier are the only necessary manual adaptations to handle arbitrary datasets). Constant variables are selected and excluded from further comparisons. The other variables are reduced to the number of unique values (levels).

	Fin.c.no	Fin.c.nm	Fin.c.ls	Raw.c.no	Raw.c.nm	Raw.c.ls	⋯
1	1	monthF	12	1	monthR	12	⋯
2	1	monthF	12	2	DriversKilledR	10	⋯
3	1	monthF	12	3	driversR	12	⋯
4	1	monthF	12	4	frontR	12	⋯
5	1	monthF	12	5	rearR	11	⋯
6	1	monthF	12	6	kmsR	12	⋯
7	1	monthF	12	7	PetrolPriceR	11	⋯
8	1	monthF	12	8	VanKilledR	8	⋯
9	2	DriversKilledF	12	1	monthR	12	⋯
10	2	DriversKilledF	12	2	DriversKilledR	10	⋯
11	2	DriversKilledF	12	3	driversR	12	⋯
12	2	DriversKilledF	12	4	frontR	12	⋯
13	2	DriversKilledF	12	5	rearR	11	⋯
14	2	DriversKilledF	12	6	kmsR	12	⋯
15	2	DriversKilledF	12	7	PetrolPriceR	11	⋯
16	2	DriversKilledF	12	8	VanKilledR	8	⋯
17	3	driversF	12	1	monthR	12	⋯
18	3	driversF	12	2	DriversKilledR	10	⋯
19	3	driversF	12	3	driversR	12	⋯
20	3	driversF	12	4	frontR	12	⋯
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋯

Table 1: Example output as in `results` and `compareFinRaw_results.csv`

Table 1 shows the first 20 rows of the left side of the full result table as given in the resulting data frame `results` within the `RData` file `compareFinRaw.RData` or in the CSV file `compareFinRaw_results.csv` (both to be found in folder “Results” after having run the program) with the columns

- consecutive row number of the output,
- consecutive numbers of the evaluated variables from both datasets (`Fin.c.no`, `Raw.c.no`),
- corresponding variable names (`Fin.c.nm`, `Raw.c.nm`), and
- numbers of unique values (levels) per variable (`Fin.c.ls`, `Raw.c.ls`).

The distances for every pair of nonconstant variables are measured by distinct program lines. In lines 82-83, the case-wise Levenshtein distances are calculated. Another one sums up those distances per variable pair (86) and one calculates their quantiles (89). Their type is the first one given by `R` [R 2011] and the underlying publication [Hyndman 1996]. The code chunk's last line assigns the mapping distance (92). For the full coding please look up Appendix A.

```
81  # compute case-wise Levenshtein distances
82  coldist <- function(x, c1, c2) adist(x[c1], x[c2])
83  ed.di   <- apply(x, 1, coldist, c1 = "a", c2 = "b")
84
85  # sum up Levenshtein distances per colum pair
86  ed.di.sum <- sum(ed.di)
87
88  # quantiles (inverse of empirical distribution function -> type 1)
89  ed.di.qua <- quantile(ed.di, probs = seq(0, 1, 0.25), na.rm = TRUE, type = 1)
90
91  # difference in number of unique values between the mapping and the variables
92  map.di <- 2 * uni.len["map"] - uni.len["a"] - uni.len["b"]
```

	···	map.di	ed.di.sum	ed.di.min	ed.di.25	ed.di.med	ed.di.75	ed.di.max
1	···	0	0	0	0	0	0	0
2	···	2	29	2	2	2	3	3
3	···	0	43	3	3	4	4	4
4	···	0	33	2	2	3	3	4
5	···	1	32	2	2	3	3	3
6	···	0	52	3	4	4	5	5
7	···	1	193	15	16	16	16	17
8	···	4	17	1	1	1	2	2
9	···	0	22	1	1	2	2	3
10	···	2	26	1	2	2	3	3
11	···	0	41	3	3	3	4	4
12	···	0	31	1	2	3	3	4
13	···	1	32	2	2	3	3	3
14	···	0	47	3	3	4	4	5
15	···	1	183	14	14	15	16	16
16	···	4	18	1	1	1	2	2
17	···	0	43	3	3	4	4	4
18	···	2	34	2	2	3	3	4
19	···	0	37	2	3	3	3	4
20	···	0	37	2	3	3	3	4
⋮	···	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 2: Example output as in `results` and `compareFinRaw_results.csv` (cont.)

Table 2 shows the first 20 rows of the right side of the full result table as given in the data frame `results` within `compareFinRaw.RData` or `compareFinRaw_results.csv`:

- mapping distance (`map.di`),
- sum of all case-wise Levenshtein distances per variable pair (`ed.di.sum`),
- their quantiles (`ed.di.min`, `ed.di.25`, `ed.di.med`, `ed.di.75`, `ed.di.max`).

The column `map.di` shows the occurrence of bijections (if equal to 0). An identity between two variables is shown by column `ed.di.sum` (if equal to 0).

References

[Harvey 1986] Harvey, A. C. and Durbin, J. (1986). The Effects of Seat Belt Legislation on British Road Casualties: A Case Study in Structural Time Series Modelling. J R Stat Soc Ser A, 149(3), 187–227. Blackwell Publishing for the Royal Statistical Society. http://www.jstor.org/stable/2981553.

[Hyndman 1996] Hyndman, R. J. and Fan, Y. (1996). Sample Quantiles in Statistical Packages. Am Stat, 50(4), 361–365. American Statistical Association. http://www.jstor.org/stable/2684934.

[Leisch 2002] Leisch, F. (2002). Sweave: Dynamic Generation of Statistical Reports Using Literate Data Analysis. In W. Härdle and B. Rönz, editors, Compstat 2002 — Proceedings in Computational Statistics, 575–580. Physika Verlag, Heidelberg, Germany, ISBN 3-7908-1517-9. http://www.stat.uni-muenchen.de/~leisch/Sweave.

[LevDis 2012] Wikipedia (2012). Levenshtein distance — Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/wiki/Levenshtein_distance (accessed 2012-Jul-18).

[R 2011] R Development Core Team (2011). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, http://www.R-project.org/.

Appendix A: compareFinRaw.r

```r
# Id: compareFinRaw.r 51 2012-07-17 11:05:59Z amueller
# Rainer Walke and Andreas Mueller, MPIDR Rostock
# Measure the difference between two datasets

################################################################################
# obtain and prepare data

# read R-data Seatbelts as final dataset "Fin" (rows 181:192, i.e. 1984)
Fin <- cbind(month = 1:12, year = 1984, Seatbelts[181:192, ])

# read R-data Seatbelts as raw dataset "Raw" (rows 1:12, i.e. 1969)
Raw <- cbind(month = 1:12, year = 1969, Seatbelts[1:12, ])

# modify names to denote the source file type
FinNames      <- colnames(Fin)
RawNames      <- colnames(Raw)
colnames(Fin) <- paste(FinNames, "F", sep = "")
colnames(Raw) <- paste(RawNames, "R", sep = "")

# sort all cases by the unique case identifier
Fin <- Fin[order(Fin[, "monthF"]), ]
Raw <- Raw[order(Raw[, "monthR"]), ]

################################################################################
# drop constant variables

# check for constant variables
FnoV <- apply(Fin, 2, unique, drop = FALSE)
RnoV <- apply(Raw, 2, unique, drop = FALSE)

Fnv <- lapply(FnoV, length) == 1
Rnv <- lapply(RnoV, length) == 1

# eventually drop constant variables, keep others for further calculations
Fvar.mat <- Fin[, setdiff(colnames(Fin), colnames(Fin)[Fnv]), drop = FALSE]
Rvar.mat <- Raw[, setdiff(colnames(Raw), colnames(Raw)[Rnv]), drop = FALSE]

################################################################################
# define data frame "results" to store the results

# save column length (number of cases) for both datasets and "results"
DFvar   <- dim(Fvar.mat)[2]
DRvar   <- dim(Rvar.mat)[2]
col.len <- DFvar * DRvar

results <- data.frame(Fin.c.no  = rep(0, col.len),
                      Fin.c.nm  = rep(colnames(Fvar.mat), each = DRvar),
                      Fin.c.ls  = rep(0, col.len),
                      Raw.c.no  = rep(0, col.len),
                      Raw.c.nm  = rep(colnames(Rvar.mat), times = DFvar),
                      Raw.c.ls  = rep(0, col.len),
                      map.di    = rep(0, col.len),
                      ed.di.sum = rep(0, col.len),
                      ed.di.min = rep(0, col.len),
                      ed.di.25  = rep(0, col.len),
                      ed.di.med = rep(0, col.len),
                      ed.di.75  = rep(0, col.len),
                      ed.di.max = rep(0, col.len))

################################################################################
# store measures for all pairs of nonconstant Fin and Raw variables

i <- 1; j <- 1
for (i in seq(DFvar)) {
  for (j in seq(DRvar)) {

    a <- as.character(Fvar.mat[, i])
    rNum <- (i - 1) * DRvar + j
    b <- as.character(Rvar.mat[, j])

    # mapping function per case-wise value pair (join by blank)
    map <- paste(a, b, sep = "␣")
    table <- as.data.frame(cbind(a, b, map))

    # compute the number of unique values per variable and mapping
    uni.len <- apply(table, 2, function(x) length(unique(x)))

    # keep only rows with unique mapping values
    x <- table[which(!duplicated(table$map)), ]

    # compute case-wise Levenshtein distances
    coldist <- function(x, c1, c2) adist(x[c1], x[c2])
    ed.di   <- apply(x, 1, coldist, c1 = "a", c2 = "b")

    # sum up Levenshtein distances per colum pair
    ed.di.sum <- sum(ed.di)

    # quantiles (inverse of empirical distribution function -> type 1)
    ed.di.qua <- quantile(ed.di, probs = seq(0, 1, 0.25), na.rm = TRUE, type = 1)

    # difference in number of unique values between the mapping and the variables
    map.di <- 2 * uni.len["map"] - uni.len["a"] - uni.len["b"]

    # fill the data frame with distances and quantiles
    results[rNum, c(-2, -5)] <- c(i, uni.len["a"], j, uni.len["b"],
                                  map.di, ed.di.sum, ed.di.qua)
  }}

################################################################################
# save special results as R variables

# constant variables
Fnv.cols <- Fin[, colnames(Fin)[Fnv], drop = FALSE]
Rnv.cols <- Raw[, colnames(Raw)[Rnv], drop = FALSE]

Fnv.tab <- t(Fnv.cols[1,, drop = FALSE])
Rnv.tab <- t(Rnv.cols[1,, drop = FALSE])

colnames(Fnv.tab) <- "Fin.no.var"
colnames(Rnv.tab) <- "Raw.no.var"

# bijections between variables (mapping distance = 0)
results.bijec <- results[results$map.di == 0,, drop = FALSE]

# identical variables (sum of all Levenshtein distances = 0)
results.ident <- results[results$ed.di.sum == 0,, drop = FALSE]

################################################################################
# create folder to save all results
dir.create("Results")

################################################################################
# save all R variables into one data file
save.image("Results/compareFinRaw.RData")

################################################################################
# export results as comma-separated values (CSV) files

# constant variables (separately for Fin and Raw)
write.csv(Fnv.tab, "Results/compareFinRaw_FinNoVar.csv")
write.csv(Rnv.tab, "Results/compareFinRaw_RawNoVar.csv")

# bijections beween variable pairs
write.csv(results.bijec, "Results/compareFinRaw_bijective.csv")

# pairs of identical variables (overall Levenshtein distance = 0)
write.csv(results.ident, "Results/compareFinRaw_identical.csv")

# data frame "results" (all nonconstant variables compared)
write.csv(results, "Results/compareFinRaw_results.csv")
```

Appendix B: compareFinRaw_results.rnw

```
% Id: compareFinRaw_results.rnw 49 2012-07-17 09:53:16Z amueller
\documentclass[DIV15, a4paper, 12pt, final]{scrartcl}
\usepackage[T1]{fontenc}
\usepackage[latin1]{inputenc}
\usepackage[english]{babel}
\usepackage{Sweave}

\begin{document}

\title{compareFinRaw.r -- results report}
\maketitle

<<echo = false, results = hide>>=
library(xtable)
load("Results/compareFinRaw.RData")
@

\noindent The \texttt{R} program \texttt{compareFinRaw.r} has evaluated two datasets (\texttt{Fin} and \texttt{Raw}). \texttt{Fin} has \Sexpr{dim(Fin)[2]}~variables and \Sexpr{dim(Fin)[1]}~cases and dataset \texttt{Raw} \Sexpr{dim(Raw)[2]}~variables and \Sexpr{dim(Raw)[1]}~cases. \Sexpr{dim(Fnv.tab)[1]}~variable(s) of dataset \texttt{Fin} and \Sexpr{dim(Rnv.tab)[1]} of dataset \texttt{Raw} are constant. The \Sexpr{dim(Fvar.mat)[2]}~nonconstant variable(s) in \texttt{Fin} have been compared with the \Sexpr{dim(Rvar.mat)[2]}~nonconstant variable(s) in \texttt{Raw}. \\


\section*{Summary of results}

\noindent \Sexpr{dim(results.ident)[1]} of these variable pairs consist(s) of 2~identical variables. The number of bijective variable mappings is \Sexpr{dim(results.bijec)[1]} (first~5 pairs are shown in Table~\ref{tab:bijec}). Some main numbers are shown in Figure~\ref{fig:numb}.\\

<<echo = false, results = tex>>=
numbers <- c(Fin.variables = dim(Fin)[2], Raw.variables = dim(Raw)[2], bijections = dim(results.bijec)[1], identities = dim(results.ident)[1])
xtable(results.bijec[1:5, c(2:3, 5:8)], digits = 0,
caption = "Bijective variable mappings as obtained by compareFinRaw.r (excerpt)",
label = "tab:bijec")
@
\begin{figure}
\centering
<<echo = false, fig = true, width = 12, height = 4>>=
barplot(numbers, width = 0.2, space = 1, cex.axis = 2, cex.names = 2)
@
\caption{Visualization of some numbers of the compareFinRaw.r results}
\label{fig:numb}
\end{figure}
\end{document}
```