Maximum Difference Scaling Method in the `MaxDiff` R Package

Tomasz Bartłomowicz and Andrzej Bąk

Abstract In microeconomics, measurement of consumer preferences is one of the most important elements of marketing research. Accurate measurement of preferences allows to gain an understanding of likes and dislikes of consumers. Using some statistical methods (like e.g. conjoint analysis and discrete choice models) it is possible to quantify preferences and answer the questions: What product will a consumer choose? What attribute of the product is most important? Consumer choice models attempt to answer these questions. This article describes the R package `MaxDiff` for the Maximum Difference Scaling method to assess consumer preferences from consumer choice experiments. Because practical applications of this method depend on the availability of computer software, this paper describes an implementation of the Maximum Difference Scaling method in the R package `MaxDiff`. Functions of the `MaxDiff` R package can be used for the measurement of consumer preferences. `MaxDiff` supports the design of the experiment (e.g. to build a list of features), encode the alternatives, estimate the models, etc. Some functions of

Tomasz Bartłomowicz
Wrocław University of Economics, Department of Econometrics and Computer Science, Nowowiejska 3, 58-500 Jelenia Góra, Poland,
✉ tomasz.bartlomowicz@ue.wroc.pl

Andrzej Bąk
Wrocław University of Economics, Department of Econometrics and Computer Science, Nowowiejska 3, 58-500 Jelenia Góra, Poland
✉ andrzej.bak@ue.wroc.pl

ARCHIVES OF DATA SCIENCE, SERIES A
KIT SCIENTIFIC PUBLISHING
Vol. 1, No. 1, S. 89–101, 2016

DOI 10.5445/KSP/1000058747/06
ISSN 2363-9881

the `MaxDiff` R package are presented with examples of applications in the empirical analysis of consumer preferences.

1 Introduction

Maximum Difference Scaling (MaxDiff) is a relatively new approach for measuring the importance of preferences for multiple items like product features, job-related benefits, advertising claims, product packaging, etc. Although Maximum Difference Scaling has much in common with conjoint analysis and discrete choice methods, the method is easier to use for researchers, respondents and clients. We can say, that the MaxDiff method combines the best features of traditional conjoint analysis and discrete choice methods. That is why Maximum Difference Scaling is also known as Best-Worst Scaling or Best-Worst Conjoint (Louviere, 1991). Comparison of the most popular preferenc measurement methods is presented in Table 1.

Maximum Difference Scaling (originally Best-Worst Scaling) is a preference measurement method developed by Louviere and and other researchers (see Louviere and Woodworth, 1983; Louviere, 1991; Finn and Louviere, 1992; Marley and Louviere, 2005).

With the MaxDiff method, respondents are shown subsets of the possible items in the experiment and are asked to indicate (among these subsets) the most and the least preferred (best and worst) items. Respondents typically evaluate a dozen sets where each set contains a different subset of items. The combinations of items are designed very carefully. Each item is shown with an equal number of pairs of items an equal number of times. Each respondent typically sees each item two or even more times across the MaxDiff sets. Compared to the rankings which is usually limited to a small number if items and to the scaled ratings, MaxDiff choosing is usually selective enough. Let us consider a set in which a respondent evaluates five items: A, B, C, D and E. If the respondent says that A is the best and E is the worst, these two responses inform us on seven of ten possible implied paired comparisons: A>B, A>C, A>D, A>E, B>E, C>E, D>E. In the opinion of some authors, humans are much better at judging items at extremes than in discriminating among items of middling importance of preference. Maximum Difference Scaling experiments focus on estimating preference or importance scores for typically about 15 to 30 attributes (Sawtooth Software, 2013).

Table 1 Comparison of the most popular methods of preferences measurement

Specification	Traditional conjoint analysis	Discrete choice method	Maximum Difference Scaling
Number of variables (attributes)	Max 6 attributes	Max 9 attributes	15-30 attributes
Method of profiling	Full factorial design, fractional factorial design	Blocking factorial design	Subsets of items
Method of data collection	Ranking (rating) of all profiles	Choosing of the most preferred item or any one	Choosing of the most and least preferred (best and worst) items
Model	Multiple regression model	Multinomial, conditional, mixed logit model	Multinomial logit model
Estimation method	OLS regression	Maximum likelihood method, Expectation-Maximization (EM) algorithm	Maximum likelihood method
Commercial software	SPSS, STATISTICA, Sawtooth Software	SAS/STAT, STATISTICA, S-PLUS	Sawtooth Software
Free software (GNU GPL license)	conjoint R package	DiscreteChoice R package, mlogit R package	MaxDiff R package

In MaxDiff models estimation of the utility function is typically performed using multinomial discrete choice models, in particular multinomial logit models. Several algorithms could be used in this estimation process, including maximum likelihood, neural networks and the hierarchical Bayes method. In the `MaxDiff` R package a multinomial logit model with maximum likelihood estimation method is used. Additional information about the MaxDiff method can be found in (Cohen, 2003; Louviere, 1991; Sawtooth Software, 2013).

2 The `MaxDiff` R package functions

The `MaxDiff` package is an implementation of the Maximum Difference Scaling method for R (Bartłomowicz and Bąk, 2013). The package is available under the GNU General Public License with free access to source code. The current version of the `MaxDiff` package is 1.12. It is possible to download the

package from the CRAN packages repository[1] and the home WWW page of the Department of Econometrics and Computer Science of the Wrocław University of Economics[2]. To use the package it is necessary to install the base R computer program (R Development Core Team, 2013) and two other packages: `mlogit` (Croissant, 2012) for estimation of the logit models and `AlgDesign` (Wheeler, 2004) for generating fractional factorial designs.

The current version of the `MaxDiff` package (v. 1.12) has thirteen functions. All of them (with their arguments and short description) are presented in Table 2 (in order of the MaxDiff procedure).

The first two functions are used to make fractional factorial design with the suggested number of profiles and alternatives in each block of profiles using vector (or matrix) of alternatives' names. The function `mdBinaryDesign()` returns a binary fractional factorial design while the function `mdAggregate-Design()` returns an aggregate fractional factorial design. If we want to make a design with alternatives' names, it is necessary to use next the `mdDesign-Names()` function which replaces the binary or aggregate fractional factorial design in the design with the alternatives' names.

The next two functions: `mdAggregateToBinaryDesign()` and `mdBinaryToAggregateDesign()` convert binary designs to aggregate designs or aggregate designs to binary ones. These functions are complements of each other, because for some of the functions (`mdRankData()`, `mdLogitData()`, `mdLogitIndividualCounts()`, `mdLogit-Individual-Ranks()`, `mdMeanRanks()`, `mdLogitModel()`, `mdLogitRanks()` and `mdMeanIndividualCounts()`) it is necessary to convert an aggregate design to a binary design.

In the group of data set functions there are two functions, namely the function `mdRankData()` which converts a basic data set into a rank data set and the function `mdLogitData()` which converts a rank data set into a logit data set. In the first case the basic data set from questionnaires is converted into a special data set for almost all functions of the `MaxDiff` package except the function `mdLogitModel()`. For this last function, it is necessary to use the `mdLogitData()` function which converts rank data into a special data set for the logit model which is estimated with the `mlogit` R package.

In the next group of functions we find the function `mdMeanIndividual-Counts()` and the function `mdMeanRanks()`. The first of them computes

[1] http://cran.r-project.org/web/packages/MaxDiff

[2] http://keii.ue.wroc.pl/MaxDiff/

Table 2 Functions of MaxDiff R package with required arguments

Function header and description
mdBinaryDesign(profiles.number, alternatives.per.profile.number, alternatives.names) function makes binary fractional factorial design with suggested number of profiles and alternatives in each profile using vector (or matrix) of alternatives' names
mdAggregateDesign(profiles.number, alternatives.per.profile.number, alternatives.names) function makes aggregate fractional factorial design with suggested number of profiles and alternatives in each profile using vector (or matrix) of alternatives' names
mdDesignNames(binary.or.aggregate.design, alternatives.names) function replaces binary or aggregate fractional factorial design in design with alternatives' names
mdAggregateToBinaryDesign(aggregate.design, alternatives.names) function converts aggregate design to binary design with alternatives' names
mdBinaryToAggregateDesign(binary.design) function converts binary design to aggregate design
mdRankData(basic.data, binary.design) function converts basic data set into rank data set for functions: *mdIndividualCounts()*, *mdLogitData()*, *mdLogitRanks()*, *mdLogitIndividualCounts()*, *mdLogitIndividualRanks()*, *mdMeanRanks()*
mdLogitData(rank.data, binary.design, alternatives.names) function converts rank data set into logit data set for *mdLogitModel()* function
mdMeanIndividualCounts(rank.data, binary.design) function computes the individual-level counts for each respondents
mdMeanRanks(rank.data, binary.design) function computes the overall counts for the whole sample using the arithmetic mean
mdLogitModel(logit.data, binary.design, alternatives.names) function estimates an aggregate logit model
mdLogitRanks(rank.data, binary.design, alternatives.names) function computes the overall counts and ranks for the whole sample using the logit model
mdLogitIndividualCounts(rank.data, binary.design, alternatives.names) function computes the individual-level counts for each respondent using the logit model
mdLogitIndividualRanks(rank.data, binary.design, alternatives.names) function computes the individual-level ranks for each respondent using the logit model

Arguments of functions	
profiles.number	Number of profiles in every block
alternatives.per.profile.number	Number of alternatives in every block of profiles
alternatives.names	Vector (or matrix) with alternatives' names
binary.or.aggregate.design	Binary or aggregate fractional factorial design
aggregate.design	Aggregate fractional factorial design
binary.design	Binary fractional factorial design
basic.data	Data set from questionnaires
rank.data	Data set with ranks
logit.data	Data set for logit model

the individual level counts for each respondent, while the second one computes the overall counts for the whole sample using the arithmetic mean.

The last group of functions is linked to the logit model. In this group there are four functions. The first function – `mdLogitModel()` estimates an aggregate logit model. The second one – `mdLogitRanks()` computes the overall counts and ranks for the whole sample using the logit model. The third one, the function `mdLogitIndividualCounts()`, computes the individual level counts for each respondent using the logit model and the fourth function `mdLogitIndividualRanks()` computes the individual-level ranks for each respondent using the logit model.

The detailed description and more examples of the use of all functions are available in the documentation of the `MaxDiff` R package (Bartłomowicz and Bąk, 2013).

3 The `MaxDiff` R package application

In the application example of the `MaxDiff` R package the identification and analysis of the preferences of respondents using some forms of job benefits is proposed. The main aim was to determine the most and least important features of the following job benefits: phone (mobile), laptop, company car, voucher, house subsidy and food subsidy. The data set of job benefits choice data allows to illustrate the use of the `MaxDiff` R package.

In the example, the job benefits experiment has 6 choice options. This means that in the outcome it was necessary to build a fractional factorial design with at least 5 profiles of job benefits. In the `MaxDiff` R package it is possible to generate a binary design (for the rest of calculations) and an aggregate design (used in the questionnaires) with profiles as a fractional factorial design. In the following example 5 profiles with 3 (from 6) attributes in each profile were generated[3]:

[3] The same fractional factorial design is saved as matrix `X` in the `Job_benefits` sample data set for `MaxDiff` R package.

```
> library(MaxDiff)
> Z=c("Phone","Laptop","Company_car","Voucher","House_subsidy",
      "Food_subsidy")
> X=mdBinaryDesign(5, 3, Z)
> print(X)
                 Profile1 Profile2 Profile3 Profile4 Profile5
Phone                1        0        0        1        1
Laptop               1        1        0        0        0
Company_car          0        1        0        1        0
Voucher              0        1        1        0        1
House_subsidy        0        0        1        1        1
Food_subsidy         1        0        1        0        0
```

The binary design can also be converted into an aggregate design with the help of the function `mdBinaryToAggregateDesign()` function[4]:

```
> X.aggregate=mdBinaryToAggregateDesign(X)
> print(X.aggregate)

  Profile1 Profile2 Profile3 Profile4 Profile5
1        1        2        4        1        1
2        2        3        5        3        4
3        6        4        6        5        5
```

Besides that, it is possible to create an aggregate design immediately with the function `mdAggregateDesign()`. To see the design in the form of a questionnaire we should replace the numbers with attributes' names using the function `mdDesignNames()`. It does not matter if we use the binary or the aggregate design as the primary parameter:

```
> survey.design=mdDesignNames(X.aggregate, Z)
> print(survey.design)

  Profile1     Profile2    Profile3      Profile4      Profile5
1 Phone        Laptop      Voucher       Phone         Phone
2 Laptop       Company_car House_subsidy Company_car   Voucher
3 Food_subsidy Voucher     Food_subsidy  House_subsidy House_subsidy
```

To present and check the `MaxDiff` R package there should be used some data. In the example we use an artificial data set for 10 respondents. The data set contains the choice of the best and worst attribute in each profile for each respondent (three attributes) and is shown below. For example, for the first respondent in the first profile the best attribute is phone (coded as 1) and the worst attribute is food subsidy (coded as 6).

[4] It is also possible to convert the prepared aggregate design into a binary design with `mdAggregateToBinaryDesign()` function.

```
> library(MaxDiff)
> data(Job_benefits)
> print(Y)
   Id Profile Best Worst
1   1       1    1     6
2   1       2    3     2
3   1       3    6     4
4   1       4    3     5
5   1       5    1     4
6   2       1    1     6
7   2       2    2     4
8   2       3    5     6
9   2       4    1     5
10  2       5    1     5
...
```

To calculate the next functions it was necessary to convert the data matrix `Y` shown above into a rank data set:

```
> rank.data=mdRankData(basic.data=Y, binary.design=X)
> print(rank.data)
         Phone Laptop Company_car Voucher House_subsidy Food_subsidy
Profile1     1      0          NA      NA            NA           -1
Profile2    NA     -1           1       0            NA           NA
Profile3    NA     NA          NA      -1             0            1
Profile4     0     NA           1      NA            -1           NA
Profile5     1     NA          NA      -1             0           NA
Profile1     1      0          NA      NA            NA           -1
...
```

A rank data set represents each alternative as a variable, with missing value codes (`NA`) used when alternatives are not shown, a `1` is used to denote an alternative that is chosen as best, `-1` for worst and `0` for alternatives shown but neither best nor worst (not chosen). This structure of data is a very useful way of setting up results in statistical programs. When data is structured in this way the counts can be, for example, computed using sums and arithmetic mean[5]:

```
> mean.ranks=mdMeanRanks(rank.data, binary.design=X)
> mean.ranks
                  Counts Ranks
Phone          0.4000000     1
Laptop         0.3500000     2
Company_car    0.3000000     3
Voucher       -0.2333333     4
House_subsidy -0.4000000     6
Food_subsidy  -0.3000000     5
```

[5] If we want to compute the individual-level counts for each respondents we should use `mdMeanIndividualCounts()` function.

With the mdMeanRanks() function it is possible to rank the attributes. According to this ranking, the most attractive are the following job-benefits: phone, then laptop and company car. The least attractive are: voucher, food subsidy and house subsidy. But because the design can be unbalanced (some attributes can be shown more times than others) it is rather a very simple way to rank the attributes. That is, why a logit model should rather be used to count and rank the attributes.

First of all, the rank data should be converted into a logit data set[6] (Food_subsidy abbreviated to Food...):

```
> logit.data=mdLogitData(rank.data, binary.design=X,
        alternatives.names=Z)
> print(head(logit.data))
  ID Set Choice Phone Laptop Company_car Voucher House_subsidy Food...
1  1   1      1     1      0           0       0             0       0
2  1   1      0     0      1           0       0             0       0
3  1   1      0     0      0           0       0             0       1
4  1   2      0    -1      0           0       0             0       0
5  1   2      0     0     -1           0       0             0       0
6  1   2      1     0      0           0       0             0      -1
...
```

The data set logit.data allows to use the mlogit R package in the MaxDiff R package to estimate the logit model:

```
> mdLogitModel(logit.data, binary.design=X, alternatives.names=Z)

Call:
mlogit(formula = formula, data = logit.data, alt.levels =
    paste(1:alternatives.per.profile.number),
    shape = "long", method = "nr", print.level = 0)

Frequencies of alternatives:
   1    2    3
0.33 0.30 0.37

nr method
4 iterations, 0h:0m:0s
g'(-H)^-1g = 0.00139
successive function values within tolerance limits

Coefficients :
               Estimate Std. Error t-value  Pr(>|t|)
Laptop         0.167594   0.436420  0.3840 0.7009635
Company_car    0.032706   0.433899  0.0754 0.9399142
Voucher       -1.413223   0.408347 -3.4608 0.0005385 ***
House_subsidy -1.752940   0.402960 -4.3502  1.36e-05 ***
```

[6] Based on: http://surveyanalysis.org/wiki/Analyzing_Max-Diff_Using_-Standard_Logit_Models_Using_R.

```
Food_subsidy  -1.528994   0.455613 -3.3559 0.0007911 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log-Likelihood: -90.622
```

The logit model estimates all parameter values relative to the first alternative (attribute phone), where the alternative has a parameter of 0. It means that 2 attributes (laptop and company car) are more attractive than phone, and 3 attributes (voucher, food subsidy and house subsidy) are not so attractive as a phone.

Similar results can be reached with the function `mdLogitRanks()`. This function computes the overall counts and ranks for the whole sample using a logit model:

```
> logit.ranks=mdLogitRanks(rank.data, binary.design=X,
         alternatives.names=Z)
> print(logit.ranks)
              Counts Rank
Phone           15.5    3
Laptop          41.4    1
Company_car     32.3    2
Voucher          1.1    6
House_subsidy    3.4    5
Food_subsidy     6.3    4

> print(logit.ranks[order(logit.ranks[, 2]), ])
              Counts Rank
Laptop          41.4    1
Company_car     32.3    2
Phone           15.5    3
Food_subsidy     6.3    4
House_subsidy    3.4    5
Voucher          1.1    6

> sum(logit.ranks[, 1])
[1] 100
```

Thus, between the attributes presented in the example there is the following MaxDiff relationship: laptop>company car>phone>food subsidy>house subsidy> voucher. This result is achieved for the whole sample (10 respondents). It is also possible to compute the individual counts and ranks for individual respondents using the logit model. The function `mdLogitIndividual-Ranks()` computes individual ranks for each respondent:

```
> mdLogitIndividualRanks(rank.data, binary.design=X,
        alternatives.names=Z)
      Phone Laptop Company_car Voucher House_subsidy Food_subsidy
 [1,]     2      5           1       6             4            3
 [2,]     1      2           3       5             4            6
 [3,]     2      1           4       3             6            5
 [4,]     1      5           2       4             3            6
 [5,]     6      3           1       5             2            4
 [6,]     2      1           5       3             6            4
 [7,]     2      3           1       6             5            4
 [8,]     4      3           2       5             6            1
 [9,]     2      1           5       3             6            4
[10,]     4      1           2       5             3            6
```

For most respondents, the phone is attractive or very attractive but most attractive is the laptop. Some respondents prefer other job benefits. For example, for 5th respondent, the phone is the least attractive and the company car is the best option. The food subsidy is the most attractive option for the 8th respondent, but only for him, not for the whole sample. That is why, for the whole sample these 3 attributes (food subsidy, house subsidy, and voucher) are the worst.

4 Conclusions

The MaxDiff package presented in this article is a new package for R designed mostly for statisticians, econometricians, economists and students of economics who are interested in the research of stated consumers preferences. As the R environment and many other packages for R, the package is available for free (under the GNU General Public License with free access to source code). Nevertheless, it is as useful as commercial specialized computer software packages.

The MaxDiff R package implements the Maximum Difference Scaling method supporting all steps of the method. It is possible to design the experiment, encode the alternatives, estimate the models, etc. within the same environment – the MaxDiff R package. By building a binary or aggregate fractional factorial design with the suggested number of profiles and alternatives it is also possible to generate a useful questionnaire for respondents in the package. In the authors' opinion, the MaxDiff R package provides an integrated support of the process of a Maximum Difference Scaling experiment for the researchers.

Table 3 R packages for measurement of stated preferences from Department of Econometrics and Computer Science Wrocław University of Economics

Package name	Implemented method	Authors	Download site
`conjoint`	Traditional *conjoint analysis* (Bąk and Bartłomowicz, 2013a)	Andrzej Bąk, Tomasz Bartłomowicz	CRAN: `http://cran.r-project.org/web/packages/conjoint/` Homepage: `http://keii.ue.wroc.pl/conjoint/`
`DiscreteChoice`	Discrete choice method (Bąk and Bartłomowicz, 2013b)	Andrzej Bąk, Tomasz Bartłomowicz	CRAN: `http://cran.r-project.org/web/packages/DiscreteChoice/` Homepage: `http://keii.ue.wroc.pl/DiscreteChoice/`
`MaxDiff`	Maximum Difference Scaling	Tomasz Bartłomowicz, Andrzej Bąk	CRAN: `http://cran.r-project.org/web/packages/MaxDiff/` Homepage: `http://keii.ue.wroc.pl/MaxDiff/`

In the current version the MaxDiff R package contains a mix of own functions and of functions of packages maintained by others to implement the Maximum Difference Scaling method. It means that to use the package it is necessary to install, in addition to the base R computer program, the `mlogit` and `AlgDesign` packages. This makes the MaxDiff R package dependent on the `mlogit` and the `AlgDesign` package. Because it is not the first package for measurement of stated preferences implemented by members of the Department of Econometrics and Computer Science of the Wroclaw University of Economics (see Table 3), the authors have some experience as maintainers of the packages: In the experience of the authors a new software version of the `mlogit` and the `AlgDesign` package often require also a software update in the MaxDiff R package. And because of this, it is desirable that in the future the MaxDiff R package does not depend on any external packages.

References

Bartłomowicz T, Bąk A (2013) Maximum Difference Scaling – package MaxDiff. URL http://keii.ue.wroc.pl/MaxDiff/

Bąk A, Bartłomowicz T (2013a) Conjoint analysis – package conjoint. URL http://cran.r-project.org/web/packages/conjoint

Bąk A, Bartłomowicz T (2013b) Discrete choice methods – package DiscreteChoice. URL http://keii.ue.wroc.pl/DiscreteChoice/

Cohen SH (2003) Maximum difference scaling: Improved measures of importance and preference for segmentation. Tech. rep., Sawtooth Software Conference Proceedings, URL http://www.sawtoothsoftware.com/download/techpap/maxdiff.pdf

Croissant Y (2012) Multinomial logit model – package mlogit. URL http://cran.r-project.org/web/packages/mlogit

Finn A, Louviere JJ (1992) Determining the appropriate response to evidence of public concern: The case of food safety. Journal of Public Policy & Marketing 11(2):12–25

Louviere JJ (1991) Best-worst scaling: A model for the largest difference judgments, Working Paper, University of Alberta

Louviere JJ, Woodworth G (1983) Design and analysis of simulated consumer choice or allocation experiments: An approach based on aggregate data. Journal of Marketing Research 20(4):350–367

Marley A, Louviere J (2005) Some probabilistic models of best, worst, and best–worst choices. Journal of Mathematical Psychology 49(6):464–480, DOI 10.1016/j.jmp.2005.05.003

R Development Core Team (2013) R: A language and environment for statistical computing, R foundation for statistical computing. URL http://cran.r-project.org/

Sawtooth Software (2013) What is MaxDiff? URL http://www.sawtoothsoftware.com/products/maxdiff-software/93-support/sales-support/238-maxdiff-method

Wheeler RE (2004) eval.design. AlgDesign. The R project for statistical computing, URL http://www.r-project.org/