OPEN ACCESS
OPUS AUGSBURG

Universität Augsburg
Universitätsbibliothek

Fechnerian Scaling in R: The Package fechner

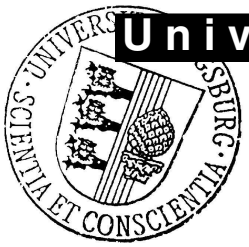
Ali Ünlü, Thomas Kiefer, Ehtibar N. Dzhafarov

Angaben zur Veröffentlichung / Publication details:

Ünlü, Ali, Thomas Kiefer, and Ehtibar N. Dzhafarov. 2009. "Fechnerian Scaling in R: The Package fechner." Augsburg: Universität Augsburg.

Nutzungsbedingungen / Terms of use: **licgercopyright**

Dieses Dokument wird unter folgenden Bedingungen zur Verfügung gestellt: / This document is made available under these conditions:
Deutsches Urheberrecht
Weitere Informationen finden Sie unter: / For more information see:
https://www.uni-augsburg.de/de/organisation/bibliothek/publizieren-zitieren-archivieren/publiz/



Universität Augsburg

Institut für
Mathematik

Ali Ünlü, Thomas Kiefer, Ehtibar N. Dzhafarov

Fechnerian Scaling in R: The Package fechner

Preprint Nr. 07/2009 — 17. April 2009
Institut für Mathematik, Universitätsstraße, D-86135 Augsburg http://www.math.uni-augsburg.de/

Impressum:

Herausgeber:
Institut für Mathematik
Universität Augsburg
86135 Augsburg
`http://www.math.uni-augsburg.de/forschung/preprint/`

ViSdP:
Ali Ünlü
Institut für Mathematik
Universität Augsburg
86135 Augsburg

Preprint: Sämtliche Rechte verbleiben den Autoren © 2009

Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *http://www.jstatsoft.org/*

Fechnerian Scaling in R: The Package fechner

Ali Ünlü
University of Augsburg

Thomas Kiefer
University of Augsburg

Ehtibar N. Dzhafarov
Purdue University

Abstract

Fechnerian scaling is a procedure for constructing a metric on a set of objects (e.g., colors, symbols, X-ray films, or even statistical models) to represent dissimilarities among the objects "from the point of view" of a system (e.g., person, technical device, or even computational algorithm) "perceiving" these objects. This metric, called Fechnerian, is computed from a data matrix of pairwise discrimination probabilities or any other pairwise measure which can be interpreted as the degree with which two objects within the set are discriminated from each other. This paper presents the package **fechner** for performing Fechnerian scaling of object sets in R. We describe the functions of the package. Fechnerian scaling then is demonstrated on real data sets accompanying the package.

Keywords: Fechnerian scaling, psychophysics, metric, subjective dissimilarity, R.

1. Introduction

This paper discusses the R (`http://www.r-project.org/`, R Development Core Team 2006) package **fechner** for Fechnerian scaling (FS) of object (or stimulus) sets. It is available on CRAN (`http://cran.r-project.org/`). FS provides a theoretical framework for deriving so-called Fechnerian distances among objects from the discrimination probabilities or other measures showing the degree with which objects are discriminated from each other by what is generically referred to as a perceiving system. In addition to the Fechnerian distances, FS also computes geodesic pathways and loops, points of subjective equality, and the generalized Shepardian dissimilarity index. (These concepts are explained in detail in Section 2.)

This paper provides a brief and by necessity schematic overview of the main concepts of FS. For detailed discussions of the various developments in this field refer to the following literature. The latest and most general version of FS is the dissimilarity cumulation theory (Dzhafarov and Colonius 2007; Dzhafarov 2008a,b). This theory extends the previously proposed theories of FS in continuous (Dzhafarov and Colonius 2005a) and discrete and discrete-continuous (Dzhafarov and Colonius 2005b) stimulus spaces. For historical background and the relation

of FS to traditional issues of psychophysics—for instance, Fechner (1860)'s original theory and its experimental and theoretical critiques—see Dzhafarov (2001, 2002a,b) and Dzhafarov and Colonius (1999, 2001). The finite, discrete version of FS, by far the most important for practical applications, is discussed in detail in Dzhafarov and Colonius (2006a). As any data set is necessarily finite, this is the version implemented in the package **fechner** and described in the present paper.

Currently available software for FS includes **FSCAMDS**, which runs on MATLAB and uses **MS Excel** for input-output functions, and a MATLAB toolbox. This software can be downloaded from, in respective order, `http://www.psych.purdue.edu/~ehtibar/` and `http://www.psychologie.uni-oldenburg.de/stefan.rach/`.

The paper is structured as follows. In Section 2, we briefly review the theory of FS. In Section 3, we present the package **fechner** and describe the functions therein. In Section 4, we demonstrate FS by applying the package's functions to real data sets accompanying the package.

2. Fechnerian scaling of object sets

Let $\{x_1, \ldots, x_n\}$ be a set of objects endowed with a discrimination function $\psi(x_i, x_j)$. The primary meaning of $\psi(x_i, x_j)$ in FS is the probability with which x_i is judged to be different from (not the same as) x_j. For example, a pair of colors (x_i, x_j) may be repeatedly presented to an observer (or a group of observers), and $\psi(x_i, x_j)$ may be estimated by the frequency of responses "they are different." Or (x_i, x_j) may be a pair of categories, and $\psi(x_i, x_j)$ the frequency of times a randomly chosen exemplar of category x_i and a randomly chosen exemplar of category x_j are judged by a person to belong to one and the same category. Or (x_i, x_j) may be a pair of statistical models, and $\psi(x_i, x_j)$ the probability with which model x_j fails to fit (by some statistical criterion) a randomly chosen data set generated by model x_i. Possible examples are numerous, and more can be found in Dzhafarov and Colonius (2006b). Moreover, ψ need not be a probability, it can be, say, the median of numerical estimates of "how dissimilar x_i and x_j are."

It is a well-established empirical fact that $\psi(x_i, x_j)$, however obtained, is not a metric:

(1) $\psi(x_i, x_i)$ is not always zero;

(2) moreover, $\psi(x_i, x_i)$ and $\psi(x_j, x_j)$ for $i \neq j$ are not generally the same;

(3) $\psi(x_i, x_j)$ is generally different from $\psi(x_j, x_i)$;

(4) and the triangle inequality is not generally satisfied either, $\psi(x_i, x_j) + \psi(x_j, x_k)$ may very well be less than $\psi(x_i, x_k)$.

The only data-analytic procedure other than FS which is aimed at imposing a metric on $\{x_1, \ldots, x_n\}$ based on ψ is nonmetric multidimensional scaling (MDS; see, e.g., Kruskal and Wish 1978). In its common version MDS assumes that $\psi(x_i, x_j)$ is some unknown monotone transformation of a "true" distance $d(x_i, x_j)$, and the MDS procedure searches for this transformation. No transformation, however, can deal with points (2) and (3) above, so the data have to be modified to make MDS applicable: e.g., $\psi(x_i, x_j)$ and $\psi(x_j, x_i)$ are replaced with their averages, and all "diagonal" values $\psi(x_i, x_i)$ are averaged over as well. Even then

MDS may not succeed in finding the transformation, especially since the class of allowable metrics in MDS is usually a priori restricted to Euclidean (or so-called Minkowskian) metrics in low-dimensional spaces of real-component vectors.

By contrast, FS deals directly with ψ-data subject to points 1–4 above, and it imposes no a priori restrictions on the class of metrics d computed from ψ. The only property of the ψ-data which is required by FS is regular minimality (RM). This property can be formulated in three statements:

(A) for every x_i there is one and only one x_j such that $\psi(x_i, x_j) < \psi(x_i, x_k)$ for all $k \neq j$ (this x_j is called the Point of Subjective Equality, or PSE, of x_i);

(B) for every x_j there is one and only one x_i such that $\psi(x_i, x_j) < \psi(x_k, x_j)$ for all $k \neq i$ (this x_i is called the PSE of x_j);

(C) and x_j is the PSE of x_i if and only if x_i is the PSE of x_j.

Every data matrix in which the diagonal entry $\psi(x_i, x_i)$ is smaller than all entries $\psi(x_i, x_k)$ in its row ($k \neq i$) and all entries $\psi(x_k, x_i)$ in its column ($k \neq i$) satisfies RM in the simplest (so-called canonical) form. In this simplest case every object x_i is the PSE of x_i. (Note that regular maximality can be defined analogously, replacing "minimal" with "maximal." This is required when the ψ-data represent closeness values rather than differences; e.g., $\psi(x_i, x_j)$ may be the percent of times x_i is judged to be the same as x_j.)

It need not always be the case, however, that every x_i is the PSE of x_i. What makes any (x_i, x_j) an ordered pair, different from (x_j, x_i), and what makes (x_i, x_i) a pair rather than a single object, is the fact that x_i and x_j (in particular, x_i and x_i), when being compared, necessarily differ in some property which "does not count" for the comparison. For example, if x_i and x_j are two colors, they must occupy two different spatial locations, or one of them may be presented first and the other second in time. This difference in spatial or temporal locations (generically referred to as the difference between two observation areas) does not enter in the comparison, but it may affect the way people perceive colors, and this in turn may lead to $\psi(x_i, x_i)$ being larger than $\psi(x_i, x_j)$ for some distinct i and j (in the same way as it may lead to $\psi(x_i, x_j) \neq \psi(x_j, x_i)$). The matrix of ψ-data

$$\begin{bmatrix} & x_1 & x_2 & x_3 \\ x_1 & 0.2 & 0.1 & 0.5 \\ x_2 & 0.7 & 0.3 & 0.2 \\ x_3 & 0.1 & 0.6 & 0.3 \end{bmatrix}$$

satisfies RM, with (x_1, x_2), (x_2, x_3), and (x_3, x_1) being pairs of mutual PSEs. Here, the first symbol in every pair refers to a row object (all row objects belonging to one, the "first," observation area) and the second symbol refers to a column object (in the "second" observation area).

Generalizing, given a matrix of $\psi(x_i, x_j)$-values with the rows and columns labeled by the objects $\{x_1, \ldots, x_n\}$, if (and only if) RM is satisfied, the row objects and column objects can be presented in pairs of PSEs $(x_1, x_{k_1}), (x_2, x_{k_2}), \ldots, (x_n, x_{k_n})$, where $(k_1, k_2, \ldots, k_n)$ is a permutation of $(1, 2, \ldots, n)$. The FS procedure identifies and lists these PSE pairs and then relabels them so that two members of the same pair receive one and the same label:

$$(x_1, x_{k_1}) \mapsto (a_1, a_1), (x_2, x_{k_2}) \mapsto (a_2, a_2), \ldots, (x_n, x_{k_n}) \mapsto (a_n, a_n).$$

Thus, the matrix in the example above becomes

$$\begin{bmatrix} & a_3 & a_1 & a_2 \\ a_1 & 0.2 & 0.1 & 0.5 \\ a_2 & 0.7 & 0.3 & 0.2 \\ a_3 & 0.1 & 0.6 & 0.3 \end{bmatrix} = \begin{bmatrix} & a_1 & a_2 & a_3 \\ a_1 & 0.1 & 0.5 & 0.2 \\ a_2 & 0.3 & 0.2 & 0.7 \\ a_3 & 0.6 & 0.3 & 0.1 \end{bmatrix}$$

in which each diagonal entry is minimal in its row and in its column. After this relabeling the original function $\psi(x_i, x_j)$ is redefined. We present it as $p_{ij} = \hat{\psi}(a_i, a_j)$ according to the rule: if $(x_i, x_{k_i}) \mapsto (a_i, a_i)$ and $\left(x_j, x_{k_j}\right) \mapsto (a_j, a_j)$, then $p_{ij} = \psi\left(x_i, x_{k_j}\right)$ (and $p_{ji} = \psi(x_j, x_{k_i})$). Note that p_{ij} is subject to the same properties 1–4 which were stipulated above for ψ. Of course, in the simplest case (canonical form), when each x_i is its own PSE, no relabeling of objects is necessary, and p_{ij} coincides with $\psi(x_i, x_j)$. (In the package **fechner** the pairs of PSEs are assigned identical labels leaving intact the labeling of the rows and relabeling the columns with their corresponding PSEs. This is referred to as canonical relabeling.)

FS imposes a metric G on the set $\{a_1, \ldots, a_n\}$ in such a way that, if x_i and $x_{i'}$ are each other's PSEs relabeled into a_i and x_j and $x_{j'}$ are each other's PSEs relabeled into a_j, then $G(x_i, x_j) = G(x_{i'}, x_{j'}) = G(a_i, a_j)$. Here is how it is done. For every pair of objects (a_i, a_j) we consider all possible chains of objects $(a_i, a_{k_1}, \ldots, a_{k_r}, a_j)$, where $(a_{k_1}, \ldots, a_{k_r})$ is a sequence chosen from $\{a_1, \ldots, a_n\}$ (and r may be 0, in which case the chain inserted between a_i and a_j is empty). For each such a chain we compute what is called its psychometric length (of the first kind) as

$$L^{(1)}(a_i, a_{k_1}, \ldots, a_{k_r}, a_j) = \sum_{m=0}^{m=r} \left(p_{k_m k_{m+1}} - p_{k_m k_m}\right),$$

where we put $a_i = a_{k_0}$ and $a_j = a_{k_{r+1}}$. (The quantities $p_{k_m k_{m+1}} - p_{k_m k_m}$ are referred to as psychometric increments of the first kind.) Then we find a chain (which need not be unique) with the minimal value of $L^{(1)}$, and take this minimal value of $L^{(1)}$ for the quasidistance $G_{ij}^{(1)}$ from a_i to a_j (referred to as the oriented Fechnerian distance of the first kind). Quasidistance (quasimetric, or oriented distance) is a pairwise measure which satisfies all metric properties except for symmetry: $G_{ij}^{(1)} = 0$ if and only if $i = j$, and $G_{ij}^{(1)} + G_{jk}^{(1)} \geq G_{ik}^{(1)}$, but $G_{ij}^{(1)}$ need not equal $G_{ji}^{(1)}$. In FS we symmetrize this quasimetric and transform it into a metric by computing $G_{ij}^{(1)} + G_{ji}^{(1)}$ and taking it for the "true" or "overall" Fechnerian distance (of the first kind) G_{ij} between a_i and a_j. Any chain $(a_i, a_{k_1}, \ldots, a_{k_r}, a_j)$ with $L^{(1)}(a_i, a_{k_1}, \ldots, a_{k_r}, a_j) = G_{ij}^{(1)}$ is called a geodesic chain (of the first kind). Then the overall Fechnerian distance G_{ij} is the psychometric length (of the first kind) of a geodesic loop $(a_i, a_{k_1}, \ldots, a_{k_r}, a_j, a_{l_1}, \ldots, a_{l_s}, a_i)$, or equivalently $(a_j, a_{l_1}, \ldots, a_{l_s}, a_i, a_{k_1}, \ldots, a_{k_r}, a_j)$.

Although this is not, strictly speaking, necessary for computations, it is worth noting that we can also compute the psychometric length (of the second kind) of an arbitrary chain $(a_i, a_{k_1}, \ldots, a_{k_r}, a_j)$ as

$$L^{(2)}(a_i, a_{k_1}, \ldots, a_{k_r}, a_j) = \sum_{m=0}^{m=r} \left(p_{k_{m+1} k_m} - p_{k_m k_m}\right)$$

(where $p_{k_{m+1} k_m} - p_{k_m k_m}$ are called psychometric increments of the second kind), and then define the quasidistance (the oriented Fechnerian distance of the second kind) $G_{ij}^{(2)}$ from a_i

to a_j as the minimal value of $L^{(2)}$ across all chains inserted between a_i and a_j. It makes, however, no difference for the final computation of the overall Fechnerian distance G_{ij} (of the first or second kind), because it can be shown that

$$G_{ij} = G_{ij}^{(1)} + G_{ji}^{(1)} = G_{ij}^{(2)} + G_{ji}^{(2)}.$$

It also holds that any geodesic loop $(a_i, a_{k_1}, \ldots, a_{k_r}, a_j, a_{l_1}, \ldots, a_{l_s}, a_i)$ of the first kind, if traversed in the opposite direction, as $(a_i, a_{l_s}, \ldots, a_{l_1}, a_j, a_{k_r}, \ldots, a_{k_1}, a_i)$, becomes a geodesic loop of the second kind. The $L^{(1)}$-length of $(a_i, a_{k_1}, \ldots, a_{k_r}, a_j, a_{l_1}, \ldots, a_{l_s}, a_i)$ equals the $L^{(2)}$-length of $(a_i, a_{l_s}, \ldots, a_{l_1}, a_j, a_{k_r}, \ldots, a_{k_1}, a_i)$; the result is G_{ij} in both cases.

The package **fechner** computes, among other quantities (Section 3), the value of G_{ij} (referred to as G in the package) and identifies a geodesic loop (perhaps one of several possible) for any pair of (relabeled) objects (a_i, a_j). It also compares the value of G_{ij} to what we call a Shepardian index of dissimilarity S_{ij} (after R. N. Shepard who used a similar transformation of probabilities in his work): $S_{ij} = p_{ij} + p_{ji} - p_{ii} - p_{jj}$. Note that $G_{ij} \leq S_{ij}$ in all cases. If the geodesic loop for (a_i, a_j) contains no other objects, i.e., if it is (a_i, a_j, a_i), then $G_{ij} = S_{ij}$.

3. The R package fechner

The package **fechner** is implemented based on the S3 system. It comes with a namespace and consists of three external functions (functions the package exports); they are described below. The package also contains internal functions (functions not exported by the package), which basically are `plot`, `print`, and `summary` methods for objects of the class `fechner`. There are two real and two artificial data sets accompanying the package **fechner**. The package's functionality and output closely follow that of the software **FSCAMDS** (Section 1). It was tested on real and artificial data and yielded the same results as obtained with **FSCAMDS**.

In the following the main functions of the package are described.

1. `check.data`

   Description: `check.data` is used to check whether the data are of required format.

   Usage:

   ```
   check.data(X, format = c("probability.different", "percent.same",
                            "general"))
   ```

   Arguments:

   `X`: a required square matrix or data frame of numeric data. No `NA`, `NaN`, `Inf`, or `-Inf` values are allowed.

   `format`: an optional character string giving the data format to be checked. This must be one of `"probability.different"`, `"percent.same"`, or `"general"`, with default `"probability.different"`, and may be abbreviated to a unique prefix.

   Value: If the data are of required format, `check.data` returns a matrix of the data with rows and columns labeled.

   Details: The data must be a matrix or data frame, have the same number of rows and columns, and be numeric consisting of real numbers. In particular, no infinite,

undefined, or missing values are allowed. This is the general data format. The probability-different and percent-same formats, in addition, require that the data lie in the intervals $[0, 1]$ and $[0, 100]$, respectively. If all of the requirements for a data format are satisfied, the data are returned as a matrix with rows and columns labeled; otherwise the function stops with respective error messages. The labeling is as follows.

The data are presented without any labeling of the rows and columns: The function does the labeling automatically, as $a1, b1, \ldots, z1$, $a2, b2, \ldots, z2$, etc., up to $a9, b9, \ldots, z9$ if the data are as large as 234×234, or if the data are larger than 234×234, the labeling is $v1, v2, \ldots, vN$, where $N \times N$ is the dimension of the data (and $N > 234$).

The data are presented with either row or column labeling: In that case, the row or column labels are assigned to the columns or rows, respectively.

The data are presented with row and column labeling: Since the labeling of both the rows and columns is now provided by the user manually, the same labeling must be used. If that is the case, the initial labeling is adopted. Otherwise the function stops with an error message.

2. `check.regular`

Description: `check.regular` is used to check whether the data satisfy regular minimality/maximality (Section 2).

Usage:

```
check.regular(X, type = c("probability.different", "percent.same",
                          "reg.minimal", "reg.maximal"))
```

Arguments:

`X`: a required square matrix or data frame of numeric data. No `NA`, `NaN`, `Inf`, or `-Inf` values are allowed.

`type`: an optional character string giving the type of check to be performed. This must be one of `"probability.different"`, `"percent.same"`, `"reg.minimal"`, or `"reg.maximal"`, with default `"probability.different"`, and may be abbreviated to a unique prefix.

Value: If the data do satisfy regular minimality/maximality, `check.regular` returns a named list consisting of the following four components:

`canonical.representation`: a matrix giving the representation of `X` in which regular minimality/maximality is satisfied in the canonical form. That is, the single minimal/maximal entries of the rows and columns lie on the main diagonal (of the canonical representation). In addition, the rows and columns are canonically relabeled.

`canonical.transformation`: a data frame giving the permutation of the columns of `X` used to produce the canonical representation of `X`. The first and second variables of this data frame, `observation.area.1` and `observation.area.2`, respectively, represent the pairs of PSEs. The third variable, `common.label`, lists the identical labels assigned to the pairs of PSEs.

`check`: a character string giving the check that was performed. This is either `"regular minimality"` or `"regular maximality"`.

`in.canonical.form`: logical. If `TRUE`, the permutation of the columns used to obtain the canonical representation of `X` is the identity; that is, the original data `X` already are in the canonical form.

Details: The argument `type` specifies whether regular minimality or regular maximality is to be checked. The values `"probability.different"` and `"percent.same"` are for data sets in the probability-different and percent-same formats, and imply regular minimality and regular maximality checks, respectively. The values `"reg.minimal"` and `"reg.maximal"` can be specified to force checking regular minimality and regular maximality, respectively, independent of the used data set. In particular, `"reg.minimal"` and `"reg.maximal"` are to be specified for data sets that are properly in the general format. The function `check.regular` calls `check.data`. In particular, the rows and columns of the canonical representation matrix are canonically relabeled based on the labeling provided by `check.data`. That is, using the `check.data` labeling, the pairs of PSEs are assigned identical labels leaving intact the labeling of the rows and relabeling the columns with their corresponding PSEs. If the data `X` do not satisfy regular minimality/maximality, `check.regular` stops with respective error messages. The error messages give information about parts of `X` violating that condition.

3. `fechner`

Description: `fechner` provides all the Fechnerian scaling computations (Section 2). It is the main function of this package.

Usage:

```
fechner(X,
        format=c("probability.different","percent.same","general"),
        compute.all = FALSE, check.computation = FALSE)
```

Arguments:

`X`: a required square matrix or data frame of numeric data. No `NA`, `NaN`, `Inf`, or `-Inf` values are allowed.

`format`: an optional character string giving the data format that is used. This must be one of `"probability.different"`, `"percent.same"`, or `"general"`, with default `"probability.different"`, and may be abbreviated to a unique prefix.

`compute.all`: an optional logical. The default value `FALSE` corresponds to short computation, which yields the main Fechnerian scaling computations. The value `TRUE` corresponds to long computation, which additionally yields intermediate results and also allows for a check of computations if `check.computation` is set `TRUE`.

`check.computation`: an optional logical. If `TRUE`, the check for whether the overall Fechnerian distance of the first kind (in the first observation area) is equal to the overall Fechnerian distance of the second kind (in the second observation area) is performed. The check requires `compute.all` to be set `TRUE`.

Value: If the arguments `X`, `format`, `compute.all`, and `check.computation` are of required types, the function `fechner` returns a named list, of the class `fechner` and with the attribute `computation`, which consists of 6 or 18 components, depending on whether short computation (`computation` is then set `short`) or long computation (`computation` is then set `long`) was performed, respectively.

(a) The short computation list contains the following 6 components:

`points.of.subjective.equality`: a data frame giving the permutation of the columns of `X` used to produce the canonical representation of `X`. The first variable and the second variable of this data frame, `observation.area.1` and `observation.area.2`, respectively, represent the pairs of PSEs. The third variable, `common.label`, lists the identical labels assigned to the pairs of PSEs.

`canonical.representation`: a matrix giving the representation of `X` in which regular minimality/maximality is satisfied in the canonical form. That is, the single minimal/maximal entries of the rows and columns lie on the main diagonal (of the canonical representation). In addition, the rows and columns are canonically relabeled.

`overall.Fechnerian.distances`: a matrix of the overall Fechnerian distances (of the first kind); by theory, invariant from observation area.

`geodesic.loops`: a data frame of the geodesic loops of the first kind; must be read from left to right for the first kind, and from right to left for the second kind.

`graph.lengths.of.geodesic.loops`: a matrix of the graph-theoretic (edge-based) lengths of the geodesic loops (of the first kind).

`S.index`: a matrix of the generalized Shepardian dissimilarity (or S–index) values.

(b) The long computation list contains the following 18 components:

`points.of.subjective.equality`: the same as in the case of short computation.

`canonical.representation`: the same as in the case of short computation.

`psychometric.increments.1`: a matrix of the psychometric increments of the first kind.

`psychometric.increments.2`: a matrix of the psychometric increments of the second kind.

`oriented.Fechnerian.distances.1`: a matrix of the oriented Fechnerian distances of the first kind.

`overall.Fechnerian.distances.1`: a matrix of the overall Fechnerian distances of the first kind.

`oriented.Fechnerian.distances.2`: a matrix of the oriented Fechnerian distances of the second kind.

`overall.Fechnerian.distances.2`: a matrix of the overall Fechnerian distances of the second kind.

`check`: if `check.computation = TRUE`, a list of two components: `difference` and `are.nearly.equal`. The component `difference` is a matrix of the differences of the overall Fechnerian distances of the first and second kind; ought to be a zero matrix. The component `are.nearly.equal` is a logical indicating whether this matrix of differences is equal to the zero matrix up to machine precision. If `check.computation = FALSE`, a character string saying "computation check was not requested."

`geodesic.chains.1`: a data frame of the geodesic chains of the first kind.

`geodesic.loops.1`: a data frame of the geodesic loops of the first kind.

`graph.lengths.of.geodesic.chains.1`: a matrix of the graph-theoretic (edge-based) lengths of the geodesic chains of the first kind.

`graph.lengths.of.geodesic.loops.1`: a matrix of the graph-theoretic (edge-based) lengths of the geodesic loops of the first kind.

`geodesic.chains.2`: a data frame of the geodesic chains of the second kind.

`geodesic.loops.2`: a data frame of the geodesic loops of the second kind.

`graph.lengths.of.geodesic.chains.2`: a matrix of the graph-theoretic (edge-based) lengths of the geodesic chains of the second kind.

`graph.lengths.of.geodesic.loops.2`: a matrix of the graph-theoretic (edge-based) lengths of the geodesic loops of the second kind.

`S.index`: the same as in the case of short computation.

Details: The argument `format` specifies the data format that is used. The values `"probability.different"` and `"percent.same"` are for data sets in the probability-different and percent-same formats. In the latter case, the data are automatically transformed prior to the analysis using the transformation $(100-X)/100$. The value `"general"` is to be used for data sets that are properly in the general data format. Note that for `"percent.same"` the data must satisfy regular maximality, for `"probability.different"` and `"general"` regular minimality. In particular, data in the general format may possibly need to be transformed manually prior to calling the function `fechner`.

If `compute.all = TRUE` and `check.computation = TRUE`, the performed check computes the difference "overall Fechnerian distance of the first kind minus overall Fechnerian distance of the second kind." By theory, this difference is zero. The function `fechner` calculates that difference and checks for equality of the distances up to machine precision. The function `fechner` calls `check.regular`, which in turn calls `check.data`. In particular, the specified data format and regular minimality/maximality are checked, and the rows and columns of the canonical representation matrix are canonically relabeled based on the labeling provided by `check.data`.

The function `fechner` returns an object of the class `fechner`, for which `plot`, `print`, and `summary` methods are provided. Moreover, objects of the class `fechner` are set the specific named attribute `computation`, which is assumed to have the value `short` or `long` indicating whether short computation (`compute.all = FALSE`) or long computation (`compute.all = TRUE`) was performed.

Next the S3 methods to plot, print, and summarize objects of the class `fechner` are described.

4. `plot`

Description: S3 method to plot objects of the class `fechner`.

Usage:

```
plot(x, level = 1)
```

Arguments:

`x`: a required object of class `fechner`, obtained from a call to the function `fechner`.

`level`: an optional numeric, integer-valued and greater-equal 1, giving the level of comparison of the S–index and the overall Fechnerian distance G.

Value: If the arguments `x` and `level` are of required types, and if there are (off-diagonal) pairs of stimuli with geodesic loops containing at least `level` $-$ 1 edges, the `plot` method produces a plot, and invisibly returns `NULL`.

Details: The `plot` method graphs the results obtained from Fechnerian scaling analyses. It produces a scatterplot of the overall Fechnerian distance G versus the S–index, with rugs added to the axes and jittered (`amount = 0.01` of noise) to accommodate ties in the S–index and G values. The added diagonal line $y = x$ is for visual inspection of the deviations of the two types of values. The `level` of comparison refers to the minimum number of edges in geodesic loops plus 1. That is, choosing `level` n means that comparison involves only those S–index and G values that have geodesic loops containing not less than $n - 1$ edges. If there are no (off-diagonal) pairs of stimuli with geodesic loops containing at least `level` $-$ 1 edges (in this case a plot is not possible), the `plot` method stops with an error message.

5. `print`

Description: S3 method to print objects of the class `fechner`.

Usage:

```
print(x)
```

Arguments:

`x`: a required object of class `fechner`, obtained from a call to the function `fechner`.

Value: If the argument `x` is of required type, the `print` method prints the main results obtained from Fechnerian scaling analyses, which are the overall Fechnerian distances and the geodesic loops, and invisibly returns `x`.

6. `summary`

Description: S3 method to summarize objects of the class `fechner`.

Usage:

```
summary(object, level = 1)
```

Arguments:

`object`: a required object of class `fechner`, obtained from a call to the function `fechner`.

`level`: an optional numeric, integer-valued and greater-equal 1, giving the level of comparison of the S–index and the overall Fechnerian distance G.

Value: If the arguments `object` and `level` are of required types, and if there are (off-diagonal) pairs of stimuli with geodesic loops containing at least `level` $-$ 1 edges, the `summary` method returns a named list (of the class `summary.fechner`, which is then `print`ed) consisting of the following four components:

`pairs.used.for.comparison`: a data frame giving the pairs of stimuli (first variable `stimuli.pairs`) and corresponding S–index (second variable `S.index`) and G (third variable `Fechnerian.distance.G`) values used for comparison.

`Pearson.correlation`: a numeric giving the value of the Pearson correlation coefficient if it exists, or a character string saying "Pearson's correlation coefficient is not defined" if it does not exist.

`C.index`: a numeric giving the value of the C–index (see below).

`comparison.level`: a numeric giving the `level` of comparison used.

Details: The `summary` method outlines the results obtained from Fechnerian scaling analyses. It computes the Pearson correlation coefficient and the C–index

$$C = \frac{2\sum(S-G)^2}{\sum S^2 + \sum G^2}$$

for specific (controlled by the argument `level`) stimuli pairs with corresponding S–index and G values. The `level` of comparison refers to the minimum number of edges in geodesic loops plus 1. If there are no (off-diagonal) pairs of stimuli with geodesic loops containing at least `level` $-$ 1 edges (in this case a summary is not possible), the `summary` method stops with an error message. Individual summary information details such as individual stimuli pairs with corresponding S–index and G values can be accessed through assignment. (Note that, more precisely, the `summary` method returns an object of the class `summary.fechner`, for which a `print` method is provided.)

4. Examples

The package **fechner** contains two real (`morse` and `wish`) and two artificial (`regMin` and `noRegMin`) data sets. We use these data sets to demonstrate the functions of the package. They are briefly described first.

4.1. The data sets

`morse`—Rothkopf (1957)'s Morse code data of discrimination probabilities among 36 auditory Morse code signals for the letters $A, B, \ldots, Z$ and the digits $0, 1, \ldots, 9$. The `morse` data frame consists of 36 rows and 36 columns, representing the Morse code signals for the letters and digits $A, B, \ldots, Z, 0, 1, \ldots, 9$ presented first and second, respectively. Each number, an integer, in the data frame gives the percentage of subjects who responded "same" to the row signal followed by the column signal. Each signal consists of a sequence of dots and dashes. A chart of the Morse code letters and digits can be found at `http://en.wikipedia.org/wiki/Morse_code`. This data set gives the same-different judgements of 598 subjects in response to the 36×36 auditorily presented pairs of Morse codes. Subjects who were not familiar with Morse code listened to a pair of signals constructed mechanically and separated by a pause of approximately 1.4 seconds. Each subject was required to state whether the two signals presented were the same or different. Each number in the `morse` data frame is the percentage of roughly 150 subjects.

`wish`—Wish (1967)'s Morse-code-like data of discrimination probabilities among 32 auditory Morse-code-like signals. The `wish` data frame consists of 32 rows and 32 columns, representing the Morse-code-like signals presented first and second, respectively. Each number, a numeric, in the data frame gives the relative frequency of subjects who responded "different" to the row signal followed by the column signal. This data set gives the same-different judgements of subjects in response to the 32×32 auditorily presented pairs of codes. The 32 Morse-code-like signals in Wish (1967)'s study were 5-element sequences $T_1P_1T_2P_2T_3$, where T stands for a tone (short or long) and P stands for a pause (1 or 3 units long). The stimuli are labeled $A, B, \ldots, Z, 0, 1, \ldots, 5$, in the order they are presented in Wish (1967)'s article.

`regMin` and `noRegMin`—Artificial data of fictitious discrimination probabilities among 10 stimuli. The `regMin` and `noRegMin` data frames consist of 10 rows and 10 columns, representing the fictitious stimuli presented in the first and second observation area, respectively. Each number, a numeric, in the data frames is assumed to give the relative frequency of perceivers scoring "different" to the row stimulus followed by the column stimulus. These data sets are artificial and included for illustrating regular minimality in the non-canonical form (`regMin`) and regular minimality being violated (`noRegMin`). They differ only in the entry in the ninth row and the tenth column.

4.2. Checking data format and regular minimality/maximality

The data set `morse` is of percent-same format, the `wish` data set of probability-different format (the R output is omitted, for typographic reasons):

```
R> check.data(morse, format = "percent.same")
R> check.data(wish, format = "probability.different")
```

An example matrix without labeling of the rows and columns, of general format; `check.data` does the labeling automatically:

```
R> (X <- ((-1) * matrix(1:16, nrow = 4)))

     [,1] [,2] [,3] [,4]
[1,]   -1   -5   -9  -13
[2,]   -2   -6  -10  -14
[3,]   -3   -7  -11  -15
[4,]   -4   -8  -12  -16

R> check.data(X, format = "general")

   a1 b1  c1  d1
a1 -1 -5  -9 -13
b1 -2 -6 -10 -14
c1 -3 -7 -11 -15
d1 -4 -8 -12 -16
```

The data set `wish` satisfies regular minimality in the canonical form:

```
R> check.regular(wish)$check
```

```
[1] "regular minimality"
```

```
R> check.regular(wish)$in.canonical.form
```

```
[1] TRUE
```

The data set `morse` satisfies regular maximality in the canonical form:

```
R> check.regular(morse, type = "percent.same")$check
```

```
[1] "regular maximality"
```

```
R> check.regular(morse, type = "percent.same")$in.canonical.form
```

```
[1] TRUE
```

For typographic reasons only, in the sequel we consider small subsets of these stimulus sets, chosen to form "self-contained" subspaces: a geodesic loop for any two elements of such a subset (computed using the complete data set) is contained entirely within the subset. (Note that the results obtained from Fechnerian scaling analyses restricted to self-contained subspaces are the same as the results obtained based on the entire stimulus sets. See below.) For instance, a particular self-contained 10-code subspace of the 36 Morse codes consists of the codes for the letter B and the digits $0, 1, 2, 4, 5, \ldots, 9$.

```
R> indices <- which(is.element(names(morse), c("B", c(0, 1, 2, 4:9))))
R> f.scal.morse <- fechner(morse, format = "percent.same")
R> f.scal.morse$geodesic.loops[indices, indices]
```

```
        B    1     2    4    5      6      7       8      9      0
B       B  B1B   B2B B46B  B5B    B6B  B676B B67876B B6789B   B06B
1     1B1    1   121  141  151    161   1781     181    191    101
2     2B2  212     2  242  252    262    272     282   2192  21092
4    46B4  414   424    4  454   46B4    474    4784    494    404
5     5B5  515   525  545    5   56B5    575     585    595    505
6     6B6  616   626 6B46 6B56      6    676   67876 678976    606
7   76B67 7817   727  747  757    767      7     787   7897 789097
8 876B678  818   828 8478  858  87678    878       8    898   8908
9  9B6789  919  9219  949  959 976789   9789     989      9    909
0    06B0  010 09210  040  050    060 097890    0890    090      0
```

This part of the `morse` data satisfies regular maximality in the canonical form:

```
R> (morse.subspace <- morse[indices, indices])
```

```
   B  1  2  4  5  6  7  8  9  0
B 84 12 17 40 32 74 43 17  4  4
1  5 84 63  8 10  8 19 32 57 55
```

```
2 14 62 89 20  5 14 20 21 16 11
4 19  5 26 89 42 44 32 10  3  3
5 45 14 10 69 90 42 24 10  6  5
6 80 15 14 24 17 88 69 14  5 14
7 33 22 29 15 12 61 85 70 20 13
8 23 42 29 16  9 30 60 89 61 26
9 14 57 39 12  4 11 42 56 91 78
0  3 50 26 11  5 22 17 52 81 94
```

```
R> check.regular(morse.subspace, type = "reg.maximal")$in.canonical.form
```

```
[1] TRUE
```

We see that the Morse code discrimination probability data violates constant self-dissimilarity. For example, the Morse code for digit 1 was judged different from itself by 16% of respondents, but only by 6% for digit 0. Symmetry is violated as well: The digits 4 and 5, for instance, were judged to be different in 58% of cases when 4 was presented first, but in only 31% when 4 was presented second. Since the subspace is self-contained, the results must be the same:

```
R> f.scal.subspace.mo <- fechner(morse.subspace, format = "percent.same")
R> identical(f.scal.morse$geodesic.loops[indices, indices],
R+ f.scal.subspace.mo$geodesic.loops)
```

```
[1] TRUE
```

```
R> identical(f.scal.morse$overall.Fechnerian.distances[indices, indices],
R+ f.scal.subspace.mo$overall.Fechnerian.distances)
```

```
[1] TRUE
```

Similarly, a self-contained 10-code subspace of the 32 Morse-code-like signals consists of the codes for $S, U, W, X, 0, 1, \ldots, 5$. This part of the `wish` data satisfies regular minimality in the canonical form. Nonconstant self-dissimilarity and non-symmetry are also manifest in these Morse-code-like signals data.

```
R> indices <- which(is.element(names(wish), c("S", "U", "W", "X", 0:5)))
R> (wish.subspace <- wish[indices, indices])
```

```
     S    U    W    X    0    1    2    3    4    5
S 0.06 0.16 0.38 0.45 0.35 0.73 0.81 0.70 0.89 0.97
U 0.28 0.06 0.44 0.24 0.59 0.56 0.49 0.51 0.71 0.69
W 0.44 0.42 0.04 0.11 0.78 0.40 0.79 0.55 0.48 0.83
X 0.64 0.71 0.26 0.03 0.86 0.51 0.73 0.27 0.31 0.44
0 0.34 0.55 0.56 0.46 0.06 0.52 0.39 0.69 0.39 0.95
1 0.84 0.75 0.22 0.33 0.70 0.03 0.69 0.17 0.40 0.97
2 0.81 0.44 0.62 0.31 0.45 0.50 0.07 0.41 0.35 0.26
3 0.94 0.85 0.44 0.17 0.85 0.19 0.84 0.02 0.63 0.47
4 0.89 0.73 0.26 0.20 0.65 0.38 0.67 0.45 0.03 0.49
5 1.00 0.94 0.74 0.11 0.83 0.95 0.58 0.67 0.25 0.03
```

```
R> check.regular(wish.subspace, type = "reg.minimal")$in.canonical.form

[1] TRUE
```

The data set `regMin` satisfies regular minimality in non-canonical form and so is canonically transformed and relabeled:

```
R> regMin

      V1   V2   V3   V4   V5   V6   V7   V8   V9  V10
V1  0.21 0.36 0.62 0.49 0.93 0.93 0.92 0.98 0.97 0.18
V2  0.34 0.20 0.43 0.68 0.74 0.94 0.90 0.80 0.92 0.51
V3  0.14 0.26 0.19 0.39 0.65 0.91 0.88 0.69 0.87 0.39
V4  0.19 0.36 0.21 0.15 0.68 0.94 0.86 0.69 0.86 0.46
V5  0.37 0.34 0.18 0.45 0.35 0.97 0.54 0.48 0.91 0.77
V6  0.63 0.73 0.22 0.55 0.21 0.79 0.51 0.56 0.94 0.90
V7  0.87 0.98 0.81 0.90 0.55 0.29 0.32 0.81 0.76 0.98
V8  0.91 0.86 0.54 0.86 0.28 0.56 0.27 0.52 0.67 0.94
V9  0.56 0.87 0.42 0.69 0.31 0.92 0.68 0.14 0.68 1.00
V10 0.93 0.90 0.82 0.88 0.76 0.75 0.44 0.49 0.27 0.98

R> check.regular(regMin)

$canonical.representation
      V1   V2   V3   V4   V5   V6   V7   V8   V9  V10
V1  0.18 0.36 0.21 0.49 0.62 0.93 0.93 0.92 0.98 0.97
V2  0.51 0.20 0.34 0.68 0.43 0.74 0.94 0.90 0.80 0.92
V3  0.39 0.26 0.14 0.39 0.19 0.65 0.91 0.88 0.69 0.87
V4  0.46 0.36 0.19 0.15 0.21 0.68 0.94 0.86 0.69 0.86
V5  0.77 0.34 0.37 0.45 0.18 0.35 0.97 0.54 0.48 0.91
V6  0.90 0.73 0.63 0.55 0.22 0.21 0.79 0.51 0.56 0.94
V7  0.98 0.98 0.87 0.90 0.81 0.55 0.29 0.32 0.81 0.76
V8  0.94 0.86 0.91 0.86 0.54 0.28 0.56 0.27 0.52 0.67
V9  1.00 0.87 0.56 0.69 0.42 0.31 0.92 0.68 0.14 0.68
V10 0.98 0.90 0.93 0.88 0.82 0.76 0.75 0.44 0.49 0.27

$canonical.transformation
   observation.area.1 observation.area.2 common.label
1                  V1                V10           V1
2                  V2                 V2           V2
3                  V3                 V1           V3
4                  V4                 V4           V4
5                  V5                 V3           V5
6                  V6                 V5           V6
7                  V7                 V6           V7
8                  V8                 V7           V8
9                  V9                 V8           V9
```

```
10                V10                  V9          V10

$check
[1] "regular minimality"

$in.canonical.form
[1] FALSE
```

The data set `noRegMin` does satisfy neither regular minimality nor regular maximality:

```
R> noRegMin

      V1   V2   V3   V4   V5   V6   V7   V8   V9  V10
V1  0.21 0.36 0.62 0.49 0.93 0.93 0.92 0.98 0.97 0.18
V2  0.34 0.20 0.43 0.68 0.74 0.94 0.90 0.80 0.92 0.51
V3  0.14 0.26 0.19 0.39 0.65 0.91 0.88 0.69 0.87 0.39
V4  0.19 0.36 0.21 0.15 0.68 0.94 0.86 0.69 0.86 0.46
V5  0.37 0.34 0.18 0.45 0.35 0.97 0.54 0.48 0.91 0.77
V6  0.63 0.73 0.22 0.55 0.21 0.79 0.51 0.56 0.94 0.90
V7  0.87 0.98 0.81 0.90 0.55 0.29 0.32 0.81 0.76 0.98
V8  0.91 0.86 0.54 0.86 0.28 0.56 0.27 0.52 0.67 0.94
V9  0.56 0.87 0.42 0.69 0.31 0.92 0.68 0.14 0.68 0.05
V10 0.93 0.90 0.82 0.88 0.76 0.75 0.44 0.49 0.27 0.98

R> check.regular(noRegMin, type = "reg.minimal")

Error in check.regular(noRegMin, type = "reg.minimal") :
  regular minimality is violated:
  entry in row #1 and column #10 is minimal in row #1 but not in column #10

R> check.regular(noRegMin, type = "reg.maximal")

Error in check.regular(noRegMin, type = "reg.maximal") :
  regular maximality is violated:
  entry in row #2 and column #6 is maximal in row #2 but not in column #6
```

4.3. The main function for Fechnerian scaling

The function `fechner` is the main function of the package and provides all the Fechnerian scaling computations.

Fechnerian scaling analysis using short computation

```
R> f.scal.subspace.wi <- fechner(wish.subspace,
R+ format="probability.different",compute.all=FALSE,check.computation=FALSE)
R> f.scal.subspace.wi
```

```
overall Fechnerian distances:
     S    U    W    X    0    1    2    3    4    5
S 0.00 0.32 0.72 0.89 0.57 1.19 1.12 1.28 1.19 1.38
U 0.32 0.00 0.76 0.79 0.89 1.07 0.80 1.16 1.07 1.28
W 0.72 0.76 0.00 0.30 1.19 0.55 1.22 0.67 0.58 0.79
X 0.89 0.79 0.30 0.00 1.23 0.67 0.94 0.39 0.45 0.49
0 0.57 0.89 1.19 1.23 0.00 1.13 0.71 1.43 0.95 1.32
1 1.19 1.07 0.55 0.67 1.13 0.00 1.09 0.31 0.72 1.08
2 1.12 0.80 1.22 0.94 0.71 1.09 0.00 1.16 0.92 0.74
3 1.28 1.16 0.67 0.39 1.43 0.31 1.16 0.00 0.84 0.77
4 1.19 1.07 0.58 0.45 0.95 0.72 0.92 0.84 0.00 0.68
5 1.38 1.28 0.79 0.49 1.32 1.08 0.74 0.77 0.68 0.00

geodesic loops:
       S      U     W     X     0      1     2      3      4      5
S      S    SUS   SWS  SUXS   S0S  SU1WS SU2US SUX3XS SUX4WS SUX5XS
U    USU      U   UWU  UXWU USOSU   U1WU   U2U UX31WU  UX4WU UX5XWU
W    WSW    WUW     W   WXW  WS0W    W1W  W2XW  WX31W   WX4W  WX5XW
X   XSUX   XWUX   XWX     X   X0X  X31WX   X2X    X3X    X4X    X5X
0    0S0  0SUS0  0WS0   0X0     0    010   020   0130    040   0250
1  1WSU1   1WU1   1W1 1WX31   101      1   121    131    141 135X31
2  2USU2    2U2  2XW2   2X2   202    212     2    232    242    252
3 3XSUX3 31WUX3 31WX3   3X3  3013    313   323      3  3X4X3   35X3
4 4WSUX4  4WUX4  4WX4   4X4   404    414   424  4X3X4      4    454
5 5XSUX5 5XWUX5 5XWX5   5X5  5025 5X3135   525   5X35    545      5
```

These are the overall Fechnerian distances and the geodesic loops for the self-contained 10-code subspace of the 32 Morse-code-like signals. The geodesic chain from stimulus S to stimulus 3, for instance, in the first observation area is $(S, U, X, 3)$, and that from 3 to S is $(3, X, S)$. In the second observation area, the geodesic chains are read from right to left: $(S, X, 3)$ from S to 3, and $(3, X, U, S)$ from 3 to S. The oriented Fechnerian distances (psychometric lengths of the geodesic chains) of the first and second kind are computed under long computation (discussed later).

The information provided using short computation, an overview:

```
R> attributes(f.scal.subspace.wi)

$names
[1] "points.of.subjective.equality"   "canonical.representation"
[3] "overall.Fechnerian.distances"    "geodesic.loops"
[5] "graph.lengths.of.geodesic.loops" "S.index"

$computation
[1] "short"

$class
[1] "fechner"
```

For instance, the S–index:

```
R> f.scal.subspace.wi$S.index
```

```
     S    U    W    X    0    1    2    3    4    5
S 0.00 0.32 0.72 1.00 0.57 1.48 1.49 1.56 1.69 1.88
U 0.32 0.00 0.76 0.86 1.02 1.22 0.80 1.28 1.35 1.54
W 0.72 0.76 0.00 0.30 1.24 0.55 1.30 0.93 0.67 1.50
X 1.00 0.86 0.30 0.00 1.23 0.78 0.94 0.39 0.45 0.49
0 0.57 1.02 1.24 1.23 0.00 1.13 0.71 1.46 0.95 1.69
1 1.48 1.22 0.55 0.78 1.13 0.00 1.09 0.31 0.72 1.86
2 1.49 0.80 1.30 0.94 0.71 1.09 0.00 1.16 0.92 0.74
3 1.56 1.28 0.93 0.39 1.46 0.31 1.16 0.00 1.03 1.09
4 1.69 1.35 0.67 0.45 0.95 0.72 0.92 1.03 0.00 0.68
5 1.88 1.54 1.50 0.49 1.69 1.86 0.74 1.09 0.68 0.00
```

Fechnerian scaling analysis using long computation

An overview of the information computed under long computation, which additionally yields intermediate results and also allows for a check of computations:

```
R> f.scal.subspace.long.wi <- fechner(wish.subspace,
R+ format="probability.different",compute.all=TRUE,check.computation=TRUE)
R> attributes(f.scal.subspace.long.wi)
```

```
$names
 [1] "points.of.subjective.equality"      "canonical.representation"
 [3] "psychometric.increments.1"          "psychometric.increments.2"
 [5] "oriented.Fechnerian.distances.1"    "overall.Fechnerian.distances.1"
 [7] "oriented.Fechnerian.distances.2"    "overall.Fechnerian.distances.2"
 [9] "check"                              "geodesic.chains.1"
[11] "geodesic.loops.1"                   "graph.lengths.of.geodesic.chains.1"
[13] "graph.lengths.of.geodesic.loops.1"  "geodesic.chains.2"
[15] "geodesic.loops.2"                   "graph.lengths.of.geodesic.chains.2"
[17] "graph.lengths.of.geodesic.loops.2"  "S.index"

$computation
[1] "long"

$class
[1] "fechner"
```

The oriented Fechnerian distances (psychometric lengths of the geodesic chains) of the first kind are:

```
R> f.scal.subspace.long.wi$oriented.Fechnerian.distances.1
```

```
     S    U    W    X    0    1    2    3    4    5
S 0.00 0.10 0.32 0.28 0.29 0.60 0.53 0.52 0.56 0.69
U 0.22 0.00 0.38 0.18 0.51 0.50 0.43 0.42 0.46 0.59
W 0.40 0.38 0.00 0.07 0.69 0.36 0.75 0.31 0.35 0.48
X 0.61 0.61 0.23 0.00 0.83 0.41 0.70 0.24 0.28 0.41
0 0.28 0.38 0.50 0.40 0.00 0.46 0.33 0.60 0.33 0.52
1 0.59 0.57 0.19 0.26 0.67 0.00 0.66 0.14 0.37 0.59
2 0.59 0.37 0.47 0.24 0.38 0.43 0.00 0.34 0.28 0.19
3 0.76 0.74 0.36 0.15 0.83 0.17 0.82 0.00 0.43 0.45
4 0.63 0.61 0.23 0.17 0.62 0.35 0.64 0.41 0.00 0.46
5 0.69 0.69 0.31 0.08 0.80 0.49 0.55 0.32 0.22 0.00
```

The psychometric length of the geodesic chain $(S, U, X, 3)$ from S to 3 in the first observation area is $G^{(1)}_{S3} = 0.52$, and of the geodesic chain $(3, X, S)$ from 3 to S it is $G^{(1)}_{3S} = 0.76$. In the second observation area, the psychometric length $G^{(2)}_{S3}$ of the geodesic chain $(S, X, 3)$ from S to 3 is

```
R> f.scal.subspace.long.wi$oriented.Fechnerian.distances.2["S", "3"]

[1] 0.72
```

and the psychometric length $G^{(2)}_{3S}$ of the geodesic chain $(3, X, U, S)$ from 3 to S is

```
R> f.scal.subspace.long.wi$oriented.Fechnerian.distances.2["3", "S"]

[1] 0.56
```

The psychometric lengths of the geodesic loops in both observation areas add up to the same value, $G_{S3} = 1.28$, as they should, because by theory $G_{ij} = G^{(1)}_{ij} + G^{(1)}_{ji} = G^{(2)}_{ij} + G^{(2)}_{ji}$. Geodesic loops are concatenations of geodesic chains, hence the following equality of graph-theoretic (edge-based) lengths of chains and loops holds:

```
R> identical(f.scal.subspace.long.wi$graph.lengths.of.geodesic.chains.1 +
R+ t(f.scal.subspace.long.wi$graph.lengths.of.geodesic.chains.1),
R+ f.scal.subspace.long.wi$graph.lengths.of.geodesic.loops.1)

[1] TRUE
```

The check for whether the overall Fechnerian distance of the first kind is equal to the overall Fechnerian distance of the second kind; the difference, by theory a zero matrix (an excerpt is shown):

```
R> f.scal.subspace.long.wi$check[[1]][1:4, 1:4]

              S            U            W             X
S  0.000000e+00 0.000000e+00 0.000000e+00 -1.110223e-16
U  0.000000e+00 0.000000e+00 0.000000e+00  1.110223e-16
W  0.000000e+00 0.000000e+00 0.000000e+00  5.551115e-17
X -1.110223e-16 1.110223e-16 5.551115e-17  0.000000e+00
```

Or, the logical indicating whether this matrix of differences is equal to the zero matrix up to machine precision:

```
R> f.scal.subspace.long.wi$check[2]

$are.nearly.equal
[1] TRUE
```

4.4. Plotting and summarizing

Objects of the class `fechner` can be plotted or summarized. Plotting the `fechner` object `f.scal.morse` (computed based on the entire Morse code data set; see Section 4.2)

```
R> plot(f.scal.morse)
```

gives the scatterplot shown in Figure 1.

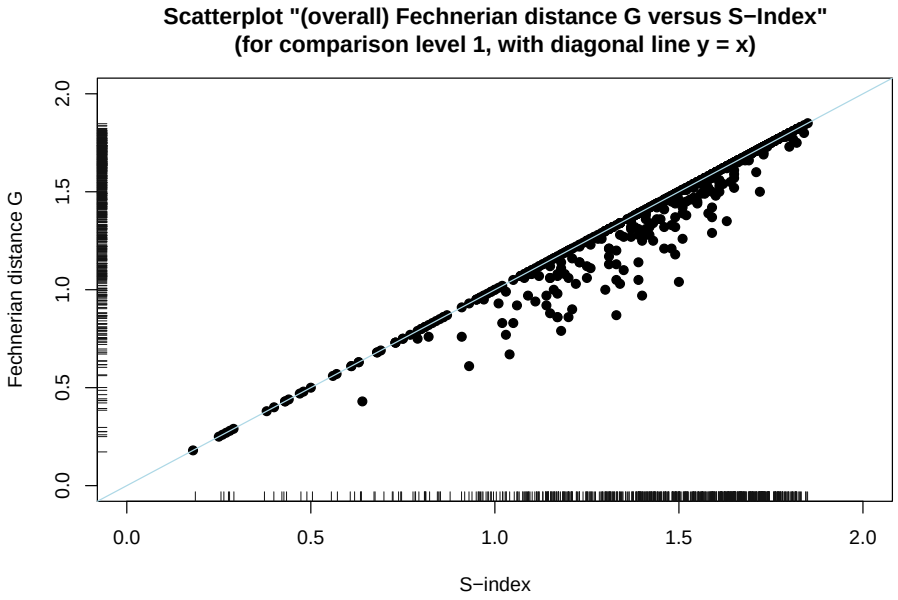


Figure 1: G versus S–index for Morse code data (all stimuli pairs).

Rugs are added to the axes and jittered (`amount = 0.01` of noise) to accommodate ties in the S–index and G values. The plot is for all (off-diagonal) pairs of stimuli (with geodesic loops containing at least 0 edges). If comparison is to involve only those S–index and G values that have geodesic loops containing not less than 4 edges, the argument `level` must be set 5 (Figure 2):

```
R> plot(f.scal.morse, level = 5)
```

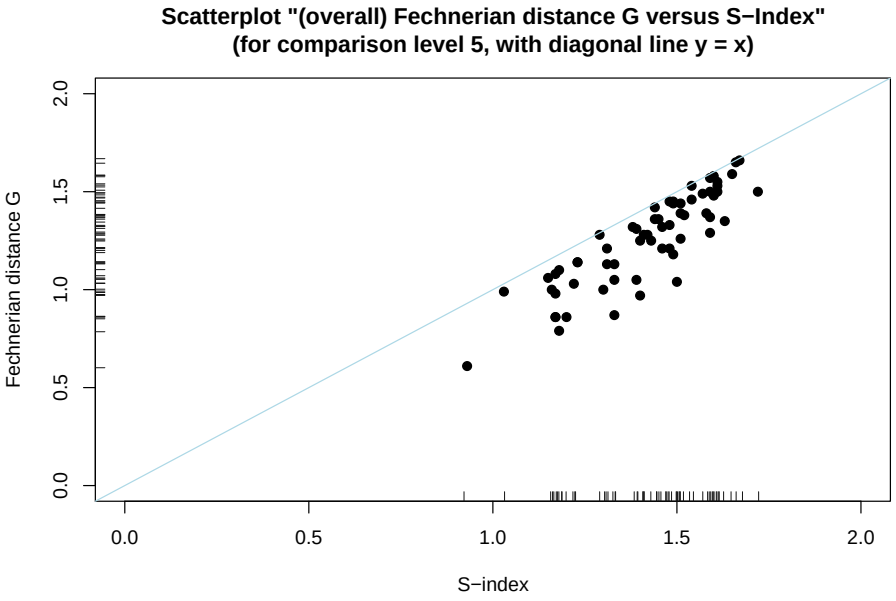


Figure 2: G versus S–index for Morse code data (specific stimuli pairs).

The corresponding summary of the `fechner` object `f.scal.morse`, including the Pearson correlation coefficient and the C–index:

```
R> summary(f.scal.morse)

number of stimuli pairs used for comparison: 630

summary of corresponding S-index values:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.180   1.260   1.520   1.435   1.670   1.850

summary of corresponding Fechnerian distance G values:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.180   1.203   1.490   1.405   1.660   1.850

Pearson correlation: 0.9764753

C-index: 0.002925355

comparison level: 1
```

In particular, detailed summary information can be accessed through assignment:

```
R> detailed.summary.mo <- summary(f.scal.morse, level = 5)
R> str(detailed.summary.mo, vec.len = 2)
```

```
List of 4
 $ pairs.used.for.comparison:'data.frame':        63 obs. of  3 variables:
  ..$ stimuli.pairs        : chr [1:63] "B.J" "B.K" ...
  ..$ S.index              : num [1:63] 1.41 1.17 1.58 1.17 1.51 ...
  ..$ Fechnerian.distance.G: num [1:63] 1.28 0.86 1.39 0.98 1.44 ...
 $ Pearson.correlation      : num 0.87
 $ C.index                  : num 0.0219
 $ comparison.level         : num 5
 - attr(*, "class")= chr "summary.fechner"
```

For instance, the pair of stimuli (B, J) and the corresponding S–index and G values can be retrieved through:

```
R> detailed.summary.mo$pairs.used.for.comparison[1, ]

  stimuli.pairs S.index Fechnerian.distance.G
1           B.J    1.41                  1.28
```

To verify that obtained information:

```
R> f.scal.morse$graph.lengths.of.geodesic.loops["B", "J"]

[1] 4

R> f.scal.morse$S.index["B", "J"]

[1] 1.41

R> f.scal.morse$overall.Fechnerian.distances["B", "J"]

[1] 1.28
```

5. Conclusion

We have introduced the package **fechner** for performing Fechnerian scaling of object sets in the R language and environment for statistical computing and graphics. The package has functions for checking the required data format and the regular minimality/maximality property, a fundamental property of discrimination in psychophysics. The main function of the package provides all the Fechnerian scaling computations, in the short and long variants. We have described the functions of the package **fechner** and demonstrated their usage on real and artificial data sets accompanying this package.

By contributing the package **fechner** in R we hope to have established a basis for computational work in this field. Interactive visualization and computational statistics approaches can be utilized in post-Fechnerian analyses to make the results obtained by Fechnerian scaling (e.g., overall Fechnerian distances and geodesic loops) more explorable and interpretable. We plan

to extend this package to incorporate such graphics as the matrix visualization, in particular combined with seriation, the fluctuation diagram variant of the mosaic plot, or the parallel coordinates plot—all as far as possible, interactively linked. (Available R packages providing for such graphics are, for example, **seriation** and **iplots**.) These visualization approaches could be used in conjunction with post-Fechnerian analyses based on multidimensional scaling (MDS) or cluster analysis (CA), as described by Dzhafarov and Colonius (2006a). Various MDS, dimensionality reduction, and CA techniques, as well as such methods as principal component analysis or factor analysis, are envisioned to be explored for their applicability to Fechnerian scaling in greater depth. The package **fechner** will have to be extended to incorporate such approaches.

The realization of Fechnerian scaling in R may also prove valuable in applying current or conventional statistical methods to the theory of Fechnerian scaling. For instance, the determination of confidence regions (e.g., for overall Fechnerian distances) and hypothesis testing (e.g., testing for RM) in Fechnerian scaling are likely to be based on resampling methods. Such an endeavor would involve extensive computer simulation, something R would be ideally suited for.

References

Dzhafarov EN (2001). "Fechnerian Psychophysics." In N Smelser, P Baltes (eds.), "International Encyclopedia of the Social and Behavioral Sciences," volume 8, pp. 5437–5440. Pergamon Press, New York.

Dzhafarov EN (2002a). "Multidimensional Fechnerian Scaling: Pairwise Comparisons, Regular Minimality, and Nonconstant Self-Similarity." *Journal of Mathematical Psychology*, **46**, 583–608.

Dzhafarov EN (2002b). "Multidimensional Fechnerian Scaling: Probability-Distance Hypothesis." *Journal of Mathematical Psychology*, **46**, 352–374.

Dzhafarov EN (2008a). "Dissimilarity Cumulation Theory in Arc-Connected Spaces." *Journal of Mathematical Psychology*, **52**, 73–92.

Dzhafarov EN (2008b). "Dissimilarity Cumulation Theory in Smoothly-Connected Spaces." *Journal of Mathematical Psychology*, **52**, 93–115.

Dzhafarov EN, Colonius H (1999). "Fechnerian Metrics in Unidimensional and Multidimensional Stimulus Spaces." *Psychonomic Bulletin and Review*, **6**, 239–268.

Dzhafarov EN, Colonius H (2001). "Multidimensional Fechnerian Scaling: Basics." *Journal of Mathematical Psychology*, **45**, 670–719.

Dzhafarov EN, Colonius H (2005a). "Psychophysics Without Physics: A Purely Psychological Theory of Fechnerian Scaling in Continuous Stimulus Spaces." *Journal of Mathematical Psychology*, **49**, 1–50.

Dzhafarov EN, Colonius H (2005b). "Psychophysics Without Physics: Extension of Fechnerian Scaling from Continuous to Discrete and Discrete-Continuous Stimulus Spaces." *Journal of Mathematical Psychology*, **49**, 125–141.

Dzhafarov EN, Colonius H (2006a). "Reconstructing Distances Among Objects from Their Discriminability." *Psychometrika*, **71**, 365–386.

Dzhafarov EN, Colonius H (2006b). "Regular Minimality: A Fundamental Law of Discrimination." In H Colonius, EN Dzhafarov (eds.), "Measurement and Representation of Sensations," pp. 1–46. Erlbaum, Mahwah, NJ.

Dzhafarov EN, Colonius H (2007). "Dissimilarity Cumulation Theory and Subjective Metrics." *Journal of Mathematical Psychology*, **51**, 290–304.

Fechner GT (1860). *Elemente der Psychophysik [Elements of Psychophysics]*. Breitkopf & Härtel, Leipzig.

Kruskal JB, Wish M (1978). *Multidimensional Scaling*. Sage, Beverly Hills.

R Development Core Team (2006). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Rothkopf EZ (1957). "A Measure of Stimulus Similarity and Errors in Some Paired-Associate Learning Tasks." *Journal of Experimental Psychology*, **53**, 94–101.

Wish M (1967). "A Model for the Perception of Morse Code-Like Signals." *Human Factors*, **9**, 529–540.

Affiliation:

Ali Ünlü
Institute of Mathematics
University of Augsburg
D-86135 Augsburg, Germany
E-mail: ali.uenlue@math.uni-augsburg.de
URL: http://www.math.uni-augsburg.de/~uenlueal/

Thomas Kiefer
Institute of Mathematics
University of Augsburg
D-86135 Augsburg, Germany
E-mail: thomas.kiefer@student.uni-augsburg.de

Ehtibar N. Dzhafarov
Department of Psychological Sciences
Purdue University
West Lafayette, IN 47907-2081, USA
E-mail: ehtibar@purdue.edu
URL: http://www.psych.purdue.edu/~ehtibar/

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume VV, Issue II *Submitted:* yyyy-mm-dd
MMMMMM YYYY *Accepted:* yyyy-mm-dd