Food and Agriculture Organization
of the United Nations

Manual for the implementation of the FAO Voices of the Hungry methods to estimate food insecurity:

RM.weights package in R

Written by Sara Viviani
FAO Statistics Division, Voices of the Hungry Project

Version 2
September, 2016

Foreword

This manual accompanies a software package in R (https://www.r-project.org/) that has been developed to estimate the prevalence of food insecurity using data from the Food Insecurity Experience Scale (FIES) applying methodology used by the Voices of the Hungry (VoH) project. The manual and the accompanying R package have been written to assist data analysts to conduct statistical validation of the FIES and to estimate population prevalence rates at national and sub-national level. An added feature is the ability to produce comparable estimates of the prevalence rates of food insecurity across different cultures and populations, both within and across countries. The package can be used by anyone with an interest in learning to use the R software and who have responsibility to analyse FIES data for monitoring food security in populations. Users will include national statistical offices carrying out their functions for assessment and monitoring through national surveys and for other organizations involved in food security assessment among different population groups.

The manual provides instructions and screenshots for each of the steps that can be carried out by the software. It is divided into 5 sections:

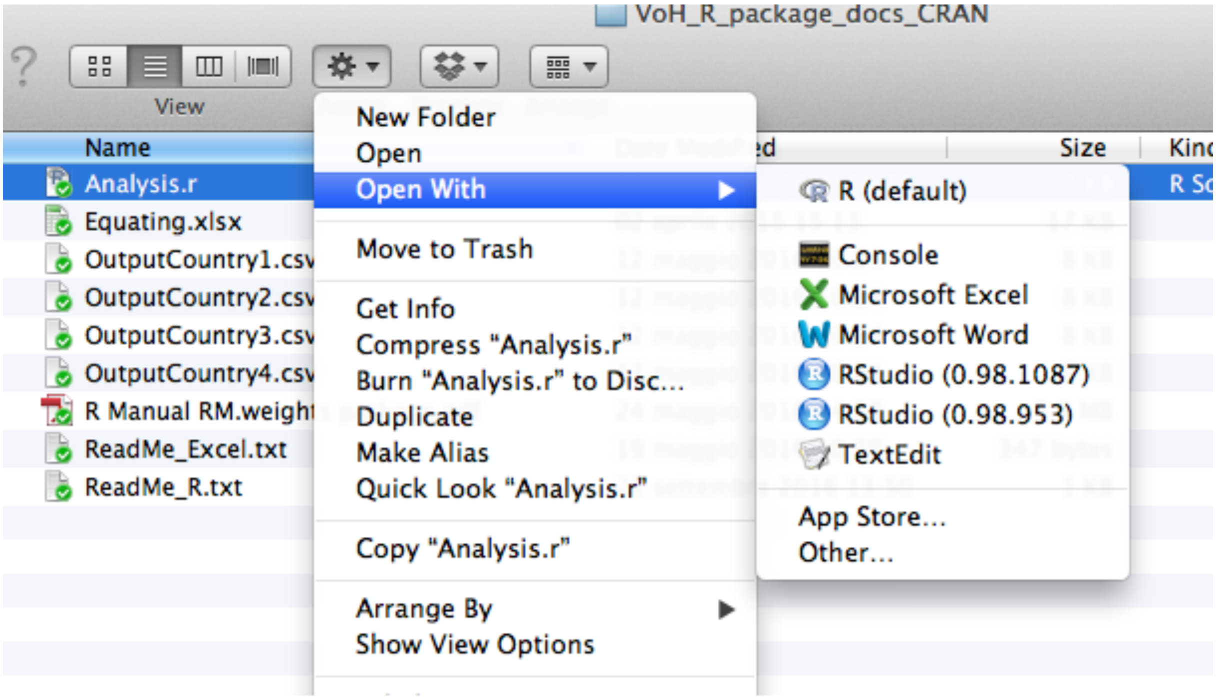
1. **Introduction to R and RStudio**. This section gets you started with the general R logic.
2. **Statistical Validation**. This section gives you preliminary instructions on installing the "RM.weights" package**,** loading and coding the FIES data in R. It then guides you through parameter estimation and interpretation using the weighted Rasch model with the "RM.w" function.
3. **Data included in the package from Gallup World Poll® 2013-2014**. This Section describes the sample datasets included in the package and assists the user to apply the statistical validation with real data.
4. **Additional functions in the RM.weights package**. In this section, we briefly illustrate the other functions included in the "RM.weights" package.
5. **An example of discrete and probabilistic assignment and some principles of equating**. This section will help you run through the entire process of discrete and probabilistic assignment to food insecurity classes using real data. Some applications of the equating procedure to calculate comparable prevalence rates are also shown.

Suggested citation

Sara Viviani (2016). Manual for the implementation of the FAO Voices of the Hungry methods to estimate food insecurity: RM.weights package in R. FAO, Rome.

The R software and manual were developed by Sara Viviani with invaluable assistance from Carlo Cafiero, Mark Nord, Chiamaka Nwosu, Filippo Gheri and Gabriela Dos Santos. Implementation of the Voices of the Hungry project has been made possible by the direct financial support from the United Kingdom Department for International Development (DfID) and from the Kingdom of Belgium through FAO Multipartner Programme Support Mechanism (FMM).

BEFORE WE START: The user should open the zip file called "VoH_R_package_docs_CRAN", which includes some explanatory documents for the data analysis, such as the syntax file called "Analysis.r". This file can be opened with Rstudio and is an example of data analysis that will facilitate the learning of the software and of data analysis.



As the official statistical software of the Voices of the Hungry (VoH) project, `R` is used for the implementation of methods to estimate food insecurity prevalence. Because of its flexibility in creating new packages, `R` is the ideal tool to carry out innovative statistical methods. This document presents the main `R` functions for the statistical validation of the Food Insecurity Experience Scale (FIES), a questionnaire used to evaluate the food insecurity severity of a population, using the Rasch model based on Item Response theory.

Users are advised to read the methodology sections of the VoH Technical Report (2016), available at http://www.fao.org/3/a-i4830e.pdf and the working paper on Rasch modelling based on Item Response Theory, available at http://www.fao.org/3/a-i3946e.pdf.

1. Introduction to R and RStudio

`R` is an open-source statistical environment widely used for data analysis. Thanks to its external packages, it offers a wide variety of statistical methods. To get an overview of its features, visit the task view relevant to your field.

Click here to download R for Mac OS X, Windows or Linux.
Once downloaded, R can be used to program directly in the console, or through a user-friendly compiler, RStudio.

RStudio is an integrated development environment for R. It includes a console, a syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management.

For online courses you can try the R online simulator
http://tryr.codeschool.com/levels/1/challenges/1
and the some of the lectures by Roger Peng
https://www.youtube.com/watch?v=EiKxy5IecUw

R is structured in packages, i.e. program modules that implement specific statistical techniques. Creating new R packages has the scope of updating the software through the implementation of new techniques. The R packages' repository is called CRAN (Comprehensive R Archive Network). Once a package is uploaded on CRAN, it becomes available for every user who has R installed.

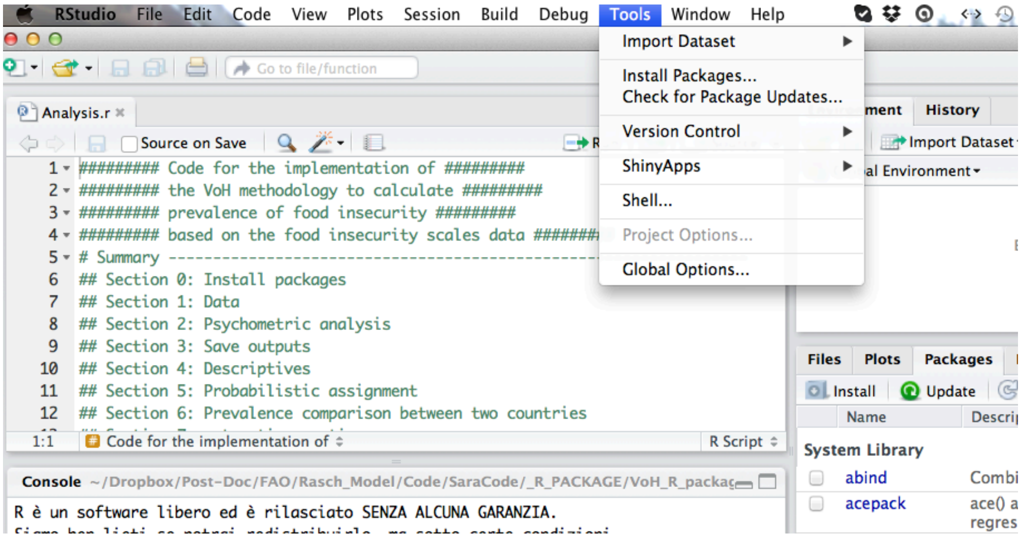
2. Statistical Validation

VoH project implemented an R package to analyze the FIES called “RM.weights”, available on CRAN.

2.1. Installation of the “RM.weights” package (Section 0 in the “Analysis.r” script)

The steps to install the “RM.weights” package in RStudio are the following:

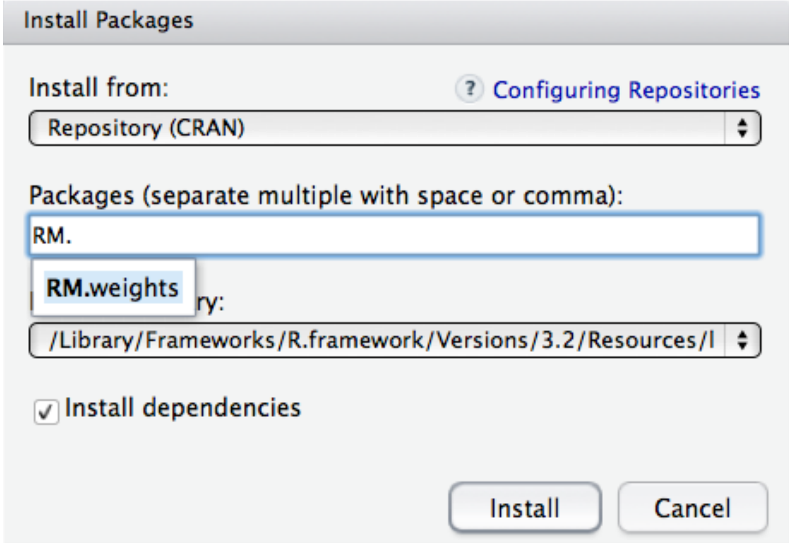
- Open RStudio (>=3.2)
- Install the “RM.weights” package from RStudio window menu (“Tools → Install Packages…”)



```
######### Code for the implementation of #########
######### the VoH methodology to calculate #########
######### prevalence of food insecurity #########
######### based on the food insecurity scales data #######
# Summary ---------------------------------------------------
## Section 0: Install packages
## Section 1: Data
## Section 2: Psychometric analysis
## Section 3: Save outputs
## Section 4: Descriptives
## Section 5: Probabilistic assignment
## Section 6: Prevalence comparison between two countries
```

```
~/Dropbox/Post-Doc/FAO/Rasch_Model/Code/SaraCode/_R_PACKAGE/VoH_R_packag
R è un software libero ed è rilasciato SENZA ALCUNA GARANZIA.
```

- In the “Install Packages” window, set “Repository (CRAN)”. Under “Package archive”, type “RM.weights” and select the “RM.weights” package. Under “Install to library”, leave the default option. Tick the “Install dependency” box. Finally, click on “Install”.



Install from: Repository (CRAN)

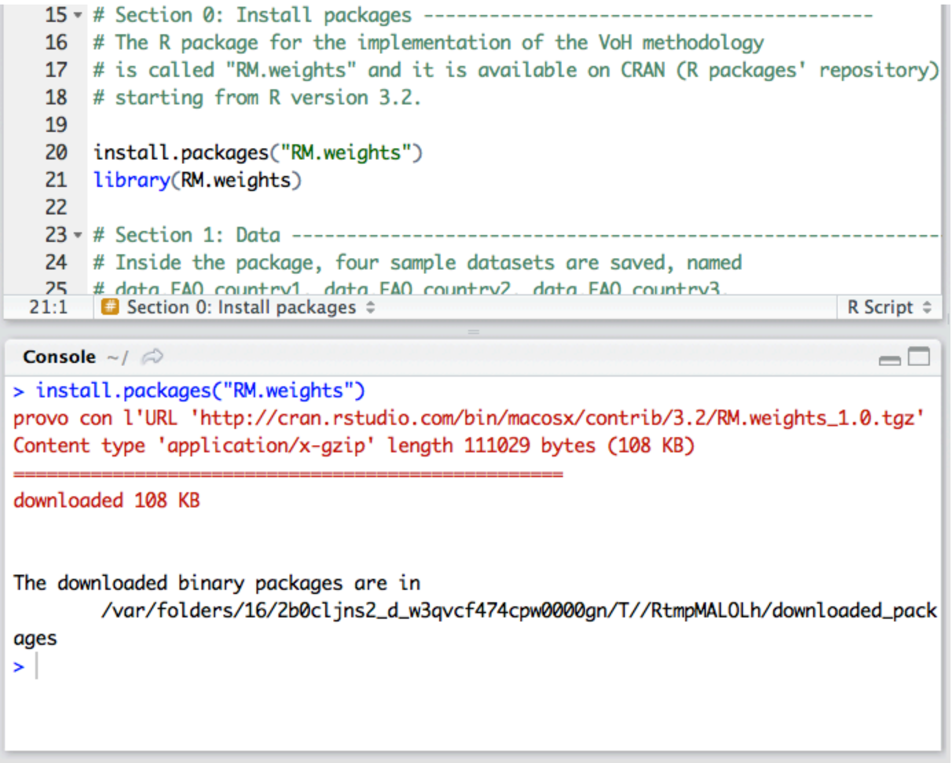
Packages (separate multiple with space or comma): RM.

RM.weights

ry: /Library/Frameworks/R.framework/Versions/3.2/Resources/l

☑ Install dependencies

- In the RStudio command window, you should read the following lines:



```
# Section 0: Install packages ----------------------------------------
# The R package for the implementation of the VoH methodology
# is called "RM.weights" and it is available on CRAN (R packages' repository)
# starting from R version 3.2.

install.packages("RM.weights")
library(RM.weights)

# Section 1: Data ------------------------------------------------------
# Inside the package, four sample datasets are saved, named
# data_FAO_country1, data_FAO_country2, data_FAO_country3
```

```
> install.packages("RM.weights")
provo con l'URL 'http://cran.rstudio.com/bin/macosx/contrib/3.2/RM.weights_1.0.tgz'
Content type 'application/x-gzip' length 111029 bytes (108 KB)
==================================================
downloaded 108 KB


The downloaded binary packages are in
	/var/folders/16/2b0cljns2_d_w3qvcf474cpw0000gn/T//RtmpMAL0Lh/downloaded_pack
ages
> 
```

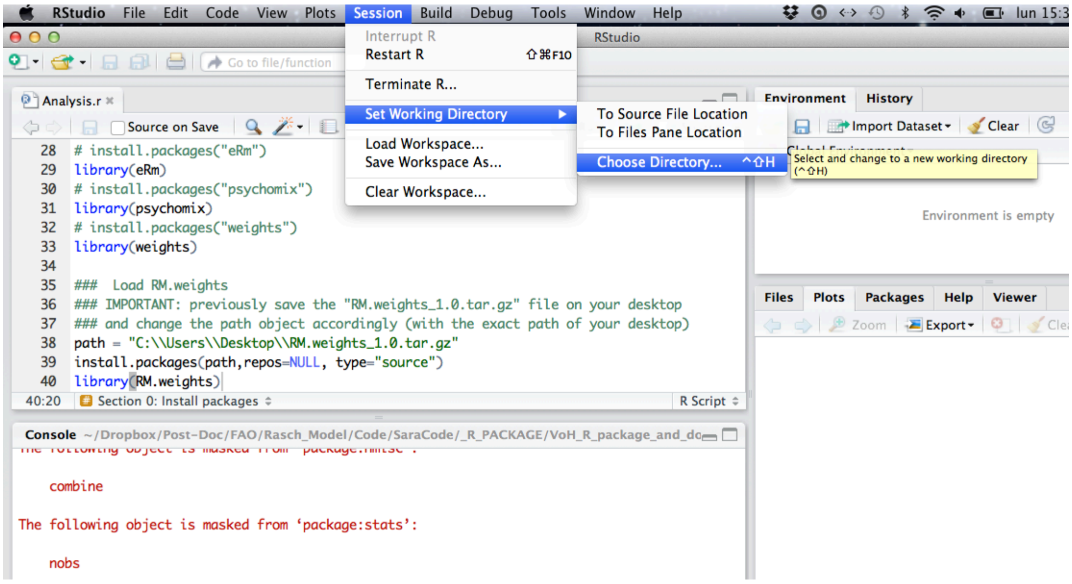
- To finally load the package, type the following code in the RStudio command window:

```
library(RM.weights)
```

- The package is now uploaded!

2.2. Load and code the FIES data in R

To load your data set in R, the first step is to set the working directory as the one where the data are saved as follows:



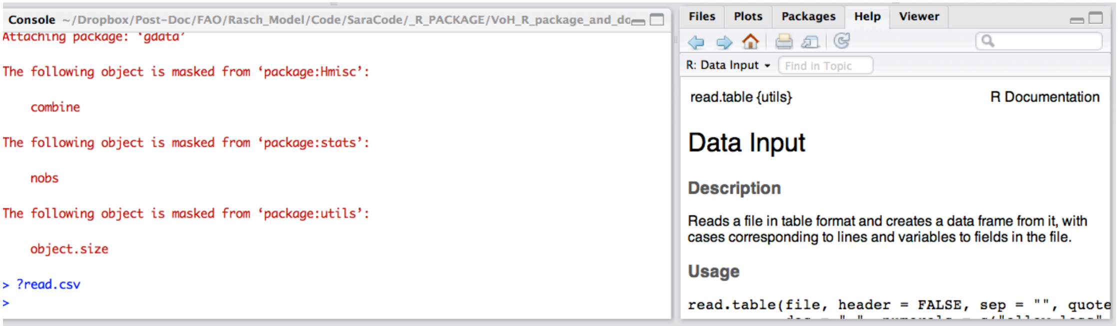
```
# install.packages("eRm")
library(eRm)
# install.packages("psychomix")
library(psychomix)
# install.packages("weights")
library(weights)

###  Load RM.weights
### IMPORTANT: previously save the "RM.weights_1.0.tar.gz" file on your desktop
### and change the path object accordingly (with the exact path of your desktop)
path = "C:\\Users\\Desktop\\RM.weights_1.0.tar.gz"
install.packages(path,repos=NULL, type="source")
library(RM.weights)
```

```

    combine

The following object is masked from ‘package:stats’:

    nobs
```

Choose the directory that contains the data file (for example if the data are saved on the desktop, choose the desktop as working directory).

After the working directory has been set, you can follow the procedure below:

- If your data are saved in a CSV format, use the function "read.csv" (type "?read.csv" in R to see the help



```
Attaching package: 'gdata'

The following object is masked from 'package:Hmisc':

    combine

The following object is masked from 'package:stats':

    nobs

The following object is masked from 'package:utils':

    object.size

> ?read.csv
>
```

read.table {utils} R Documentation

Data Input

Description

Reads a file in table format and creates a data frame from it, with cases corresponding to lines and variables to fields in the file.

Usage

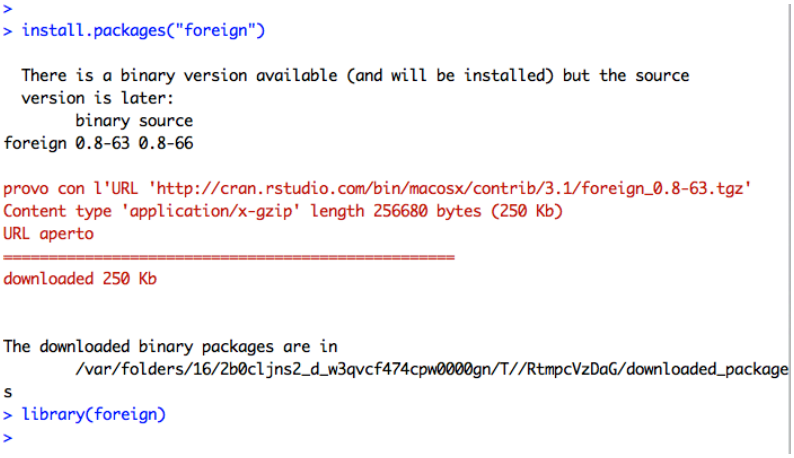
```
read.table(file, header = FALSE, sep = "", quote
```

For example

```
data = read.csv("datasetname.csv", header = T)
```

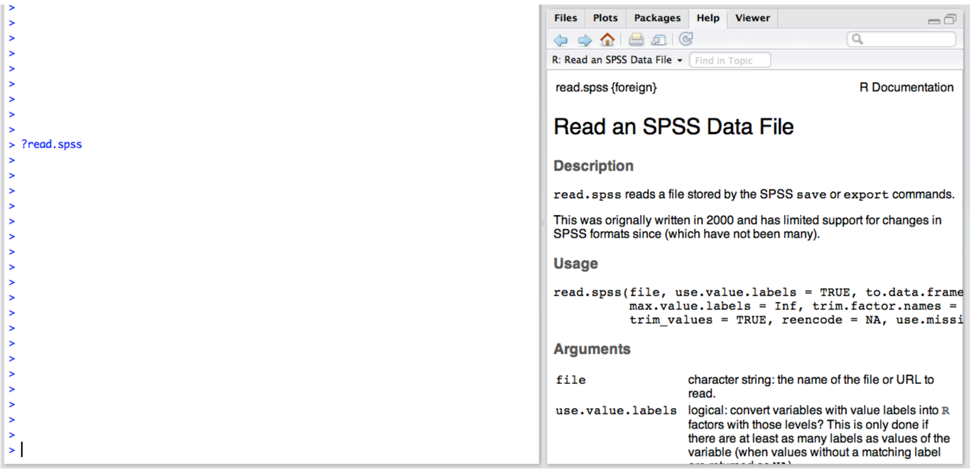
- If your data are saved in SPSS format, you need to use the "foreign" package, using the code

```
install.packages("foreign")
library(foreign)
```



```
>
> install.packages("foreign")

  There is a binary version available (and will be installed) but the source
  version is later:
        binary source
foreign 0.8-63 0.8-66

provo con l'URL 'http://cran.rstudio.com/bin/macosx/contrib/3.1/foreign_0.8-63.tgz'
Content type 'application/x-gzip' length 256680 bytes (250 Kb)
URL aperto
==================================================
downloaded 250 Kb


The downloaded binary packages are in
        /var/folders/16/2b0cljns2_d_w3qvcf474cpw0000gn/T//RtmpcVzDaG/downloaded_package
s
> library(foreign)
>
```

and then use the function "read.spss"



```
>
>
>
>
>
>
>
>
>
> ?read.spss
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
> |
```

read.spss {foreign} R Documentation

Read an SPSS Data File

Description

`read.spss` reads a file stored by the SPSS `save` or `export` commands.

This was orignally written in 2000 and has limited support for changes in SPSS formats since (which have not been many).

Usage

```
read.spss(file, use.value.labels = TRUE, to.data.frame
          max.value.labels = Inf, trim.factor.names =
          trim_values = TRUE, reencode = NA, use.missi
```

Arguments

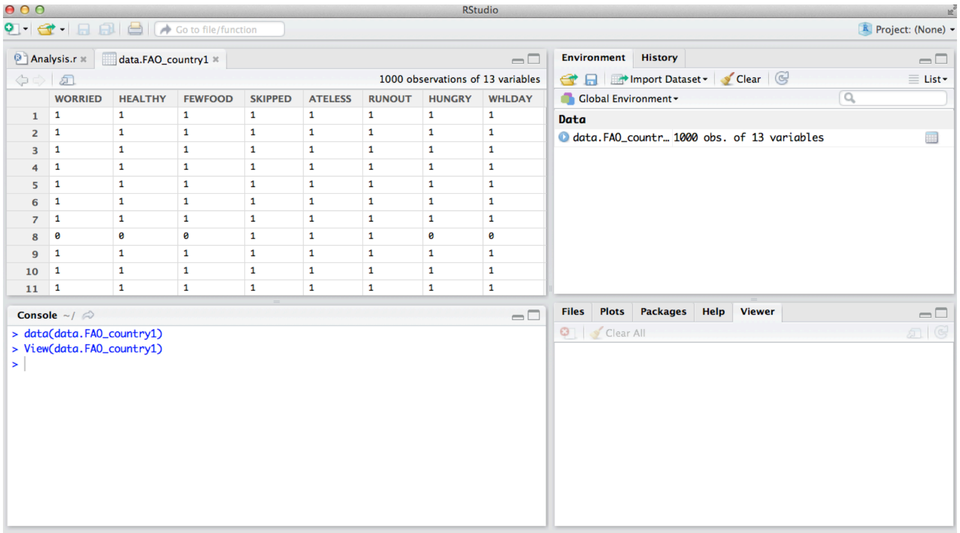
`file`	character string: the name of the file or URL to read.
`use.value.labels`	logical: convert variables with value labels into `R` factors with those levels? This is only done if there are at least as many labels as values of the variable (when values without a matching label

For example:

```
data = read.spss("datasetname.sav", to.data.frame = T)
```

- If your data are saved in STATA format, you still need to use the "foreign" package, and use the "read.dta" function. If you are using a version of STATA 13.0 or over, you first need to save the data in the "Stata 11/12 Data" format for compatibility.

- The data will be visible by clicking on the data object on the top high window in RStudio:



1000 observations of 13 variables

	WORRIED	HEALTHY	FEWFOOD	SKIPPED	ATELESS	RUNOUT	HUNGRY	WHLDAY
1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1
8	0	0	0	1	1	1	0	0
9	1	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1	1
11	1	1	1	1	1	1	1	1

Data

data.FAO_countr… 1000 obs. of 13 variables

```
> data(data.FAO_country1)
> View(data.FAO_country1)
> 
```

Once the data are loaded, you need to extract the FIES variables from the dataset. An easy way to do that is to select the columns corresponding to the FIES data. For example, if the FIES is recorded in columns 4 - 11, you can use the code:

```
XX = data[,4:11]
```

The same holds for sampling weights, if present. For example, if they are saved in column 12 of the dataset, they can be extracted with the code

```
wt = data[,12]
```

Note that "`data`", "`XX`" and "`wt`" are just invented names that can be modified should the user prefer other labels.

Note: the FIES data (here labeled `XX`) should be in a zero/one format (0 for No, 1 for Yes answers). If data are coded differently (for example 1 for Yes and 2 for No), you can recode them as follows:

```
XX[XX==2] = 0
```

or recode them before importing the data.

2.3. Use of the "RM.weights" package

"`RM.weights`" is a package that includes several functions related to the Rasch Model.

The principal function in the package is called "`RM.w`", and fits the one parameter logistic (Rasch) model[1] by using the conditional maximum likelihood (CML) approach, with the possibility of including sampling weights and many other features not available in other `R` packages.

The function's syntax is illustrated in detail in the paragraph below.

2.4. Estimation of the Rasch model: `RM.w` function

This function computes the parameter estimates of a Rasch model for binary item responses by using weighted CML estimation[2].
To see the help of the function, type the code "`?RM.w`" on the R console.

Usage

```
RM.w(.data, .w = NULL, .d=NULL, country=NULL,
      se.control = T, quantile.seq = NULL, write.file = F)
```

Arguments

`.data`	Input 0/1 data matrix or data frame; affirmative responses must be coded as 1. Rows represent individuals, columns represent items. Missing values are inserted as NA.
`.w`	Vector of sampling weights. The length must be the same as the number of rows of `.data`. If left unspecified, all the individuals will be equally weighted (`.w = rep(1, nrow(.data))`).
`.d`	Optional vector for the assumption on the extreme raw score parameters. Default is `0.5` and `(k-0.5)`, `k` being the maximum number of items (columns of .data).
`country`	Name of the country that data refer to.
`se.control`	Are the extreme parameter standard errors fixed to the ones corresponding to raw score `0.5` and `(k-0.5)`? If `FALSE`, the actual standard errors for the extreme parameters are estimated.
`quantile.seq`	Quantiles corresponding to the observed and the expected case fit statistic distributions.
`write.file`	If `TRUE`, a CSV file with the main results will be saved in

---

[1] See also page 3 of Introduction to Item Response Theory applied to Food Security Measurement

[2] See also page 7 of Introduction to Item Response Theory applied to Food Security Measurement

the working directory.

Details

The weighted CML method is used to estimate the item parameter. Respondent parameters[3] are estimated post-hoc. Cases with missing responses to some items may be included, but will not be used to estimate the Rasch model.

As the parameters for the extreme raw scores (0 and `k)`, are undefined under the CML, some assumptions are needed for population-level prevalence estimates unless the proportions of respondents with those raw scores are so small that they can be considered to be measured as highly secure/highly insecure without error. Vector `.d` gives the possibility to include up to four alternative assumptions on each of the extreme parameters.

Note: default assumptions on the extreme raw score parameter are `.d=c(0.5,k-0.5)`. This means that, instead of being estimated for raw scores 0 and k (which would lead to undefined parameters), extreme raw score parameters are estimated for 0.5 and k-0.5. These assumptions are valid for standard datasets where the proportion in raw score 0 or k is not extremely high (<40%). Otherwise, flexibility is introduced in the package to estimate alternative extreme raw score parameters as follows.

`.d` can be a two, three or four element vector:

- If `length(.d) = 4`, then the first two elements have to refer to the assumptions upon raw score 0, and the second two elements to raw score k. For instance `.d = c(0.1, 0.7, 7.1, 7.6)`, if the maximum raw score is 8.
- If `length(.d) = 3`, then the first two elements can either refer to the assumptions upon raw score 0 or raw score k, and the last one is defined accordingly. For instance `.d = c(0.1, 7.1, 7.6)` or `.d = c(0.1, 0.7, 7.6)`, if the maximum raw score is 8.
- If `length(.d) = 2`, then the first element have to refer to the assumption upon raw score 0, and the second element to raw score k. For instance `.d = c(0.1, 7.6)`, if the maximum raw score is 8.

Examples

Note: The simplest way to use the RM.w function is to specify only the name of the data and the sampling weights (all the other specifications are already set as default). The output is saved in the list named “rr.country1” in this example.

```
data(data.FAO_country1)
```

Questionnaire data and weights:

---

[3] See also page 8 of Introduction to Item Response Theory applied to Food Security Measurement

```
XX.country1 = data.FAO_country1[,1:8]
wt.country1 = data.FAO_country1$wt
```

Fit weighted Rasch:

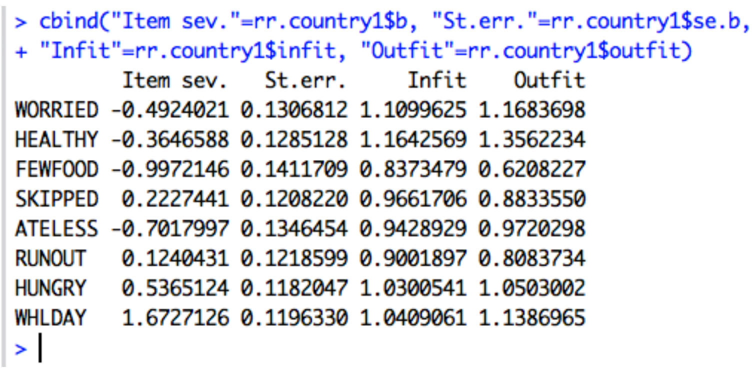
```
rr.country1 = RM.w(XX.country1, wt.country1)
```

Fit unweighted Rasch (the weights will be set all to 1)

```
rr.country1.nw = RM.w(XX.country1)
```

Display the item severities, standard errors, infits[4] and outfits[5]…

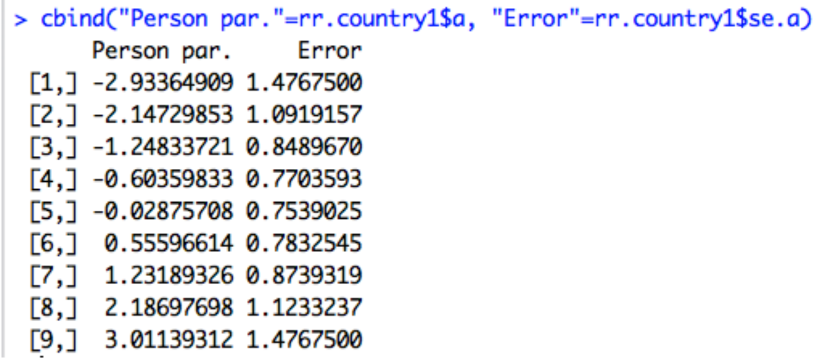
```
cbind("Item sev."=rr.country1$b, "St.err."=rr.country1$se.b,
"Infit"=rr.country1$infit, "Outfit"=rr.country1$outfit)
```



```
> cbind("Item sev."=rr.country1$b, "St.err."=rr.country1$se.b,
+ "Infit"=rr.country1$infit, "Outfit"=rr.country1$outfit)
         Item sev.   St.err.     Infit    Outfit
WORRIED -0.4924021 0.1306812 1.1099625 1.1683698
HEALTHY -0.3646588 0.1285128 1.1642569 1.3562234
FEWFOOD -0.9972146 0.1411709 0.8373479 0.6208227
SKIPPED  0.2227441 0.1208220 0.9661706 0.8833550
ATELESS -0.7017997 0.1346454 0.9428929 0.9720298
RUNOUT   0.1240431 0.1218599 0.9001897 0.8083734
HUNGRY   0.5365124 0.1182047 1.0300541 1.0503002
WHLDAY   1.6727126 0.1196330 1.0409061 1.1386965
> |
```

Display respondent severities and measurement errors:

```
cbind("Person par."=rr.country1$a, "Error"=rr.country1$se.a)
```



```
> cbind("Person par."=rr.country1$a, "Error"=rr.country1$se.a)
       Person par.     Error
 [1,] -2.93364909 1.4767500
 [2,] -2.14729853 1.0919157
 [3,] -1.24833721 0.8489670
 [4,] -0.60359833 0.7703593
 [5,] -0.02875708 0.7539025
 [6,]  0.55596614 0.7832545
 [7,]  1.23189326 0.8739319
 [8,]  2.18697698 1.1233237
 [9,]  3.01139312 1.4767500
```

Display Rasch reliability based on observed distribution of cases across raw scores

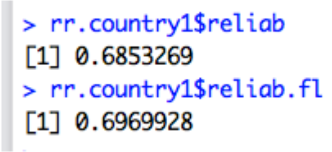
```
rr.country1$reliab
```

---

[4] See also page 7 of Introduction to Item Response Theory applied to Food Security Measurement
[5] See also page 8 of Introduction to Item Response Theory applied to Food Security Measurement

Display Rasch reliability[6] based on equal proportion of cases in each non-extreme raw score (more comparable across datasets)

```
rr.country1$reliab.fl
```



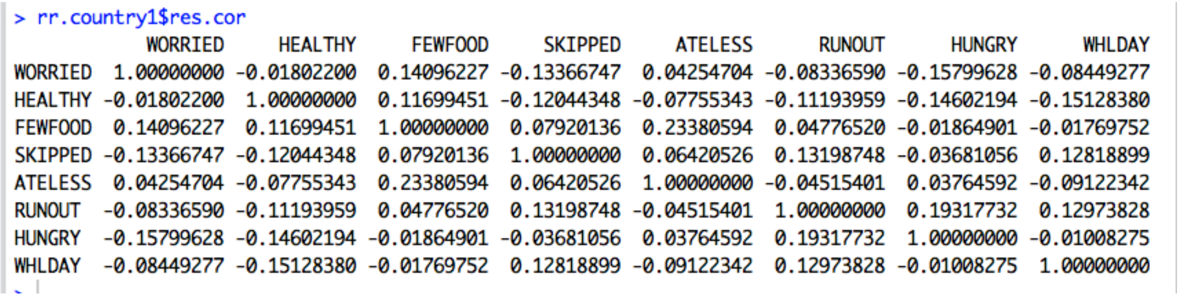
```
> rr.country1$reliab
[1] 0.6853269
> rr.country1$reliab.fl
[1] 0.6969928
```

Calculate observed and expected respondent infit distribution:

```
quantile.seq = c(0,.01,.02,.05,.10,.25,.50,.75,.90,.95,
                 .98,.99,1)
q.infit = rr.country1$q.infit
q.infit.theor  = rr.country1$q.infit.theor
plot(quantile.seq, q.infit, type = "b", xlab = "Quantiles",
     ylab = "Observed infit", ylim = c(0, 6))
lines(quantile.seq, q.infit.theor, type = "b", col = 2)
```

Display conditional independence[7] matrix

```
rr.country1$res.cor
```



```
> rr.country1$res.cor
            WORRIED     HEALTHY     FEWFOOD     SKIPPED     ATELESS     RUNOUT      HUNGRY      WHLDAY
WORRIED  1.00000000 -0.01802200  0.14096227 -0.13366747  0.04254704 -0.08336590 -0.15799628 -0.08449277
HEALTHY -0.01802200  1.00000000  0.11699451 -0.12044348 -0.07755343 -0.11193959 -0.14602194 -0.15128380
FEWFOOD  0.14096227  0.11699451  1.00000000  0.07920136  0.23380594  0.04776520 -0.01864901 -0.01769752
SKIPPED -0.13366747 -0.12044348  0.07920136  1.00000000  0.06420526  0.13198748 -0.03681056  0.12818899
ATELESS  0.04254704 -0.07755343  0.23380594  0.06420526  1.00000000 -0.04515401  0.03764592 -0.09122342
RUNOUT  -0.08336590 -0.11193959  0.04776520  0.13198748 -0.04515401  1.00000000  0.19317732  0.12973828
HUNGRY  -0.15799628 -0.14602194 -0.01864901 -0.03681056  0.03764592  0.19317732  1.00000000 -0.01008275
WHLDAY  -0.08449277 -0.15128380 -0.01769752  0.12818899 -0.09122342  0.12973828 -0.01008275  1.00000000
```

Rerun analysis to save outputs to csv file with country name

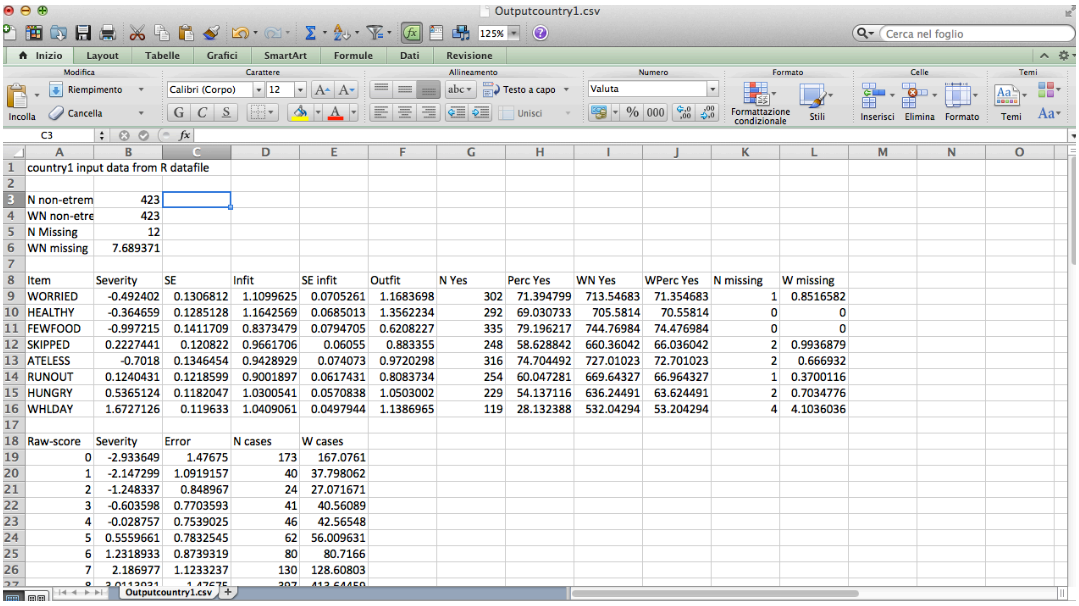
```
rr.country1 = RM.w(XX.country1, wt.country1, country =
                        "country1", write.file = T)
```

---

[6] See also page 12 of Introduction to Item Response Theory applied to Food Security Measurement

[7] See also page 15 of Introduction to Item Response Theory applied to Food Security Measurement

The output will be saved in the working directory and called “Outputcountry1.csv”:



	A	B	C
1	country1 input data from R datafile		
2			
3	N non-etrem	423	
4	WN non-etre	423	
5	N Missing	12	
6	WN missing	7.689371	

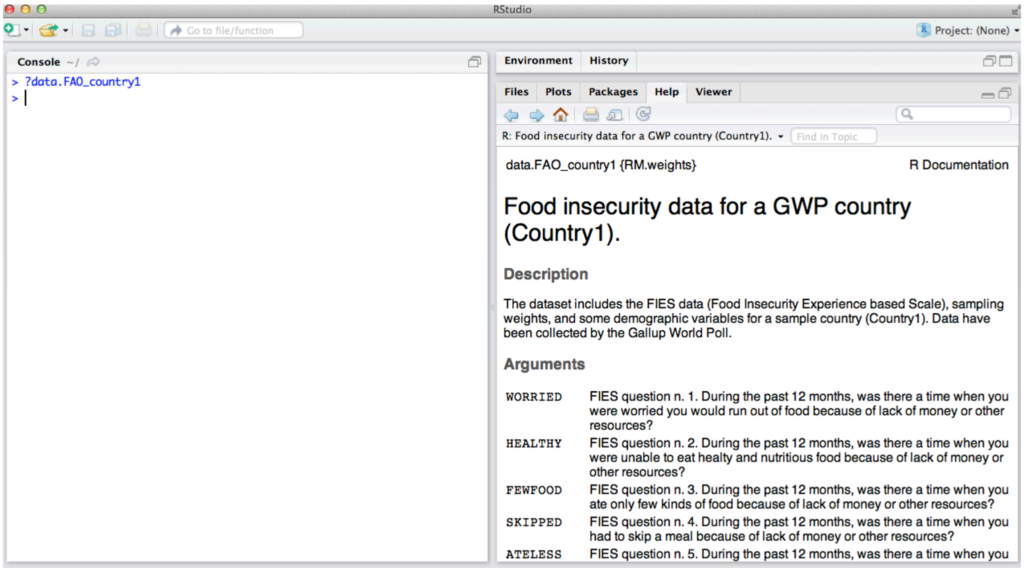
Item	Severity	SE	Infit	SE infit	Outfit	N Yes	Perc Yes	WN Yes	WPerc Yes	N missing	W missing
WORRIED	-0.492402	0.1306812	1.1099625	0.0705261	1.1683698	302	71.394799	713.54683	71.354683	1	0.8516582
HEALTHY	-0.364659	0.1285128	1.1642569	0.0685013	1.3562234	292	69.030733	705.5814	70.55814	0	0
FEWFOOD	-0.997215	0.1411709	0.8373479	0.0794705	0.6208227	335	79.196217	744.76984	74.476984	0	0
SKIPPED	0.2227441	0.120822	0.9661706	0.06055	0.883355	248	58.628842	660.36042	66.036042	2	0.9936879
ATELESS	-0.7018	0.1346454	0.9428929	0.074073	0.9720298	316	74.704492	727.01023	72.701023	2	0.666932
RUNOUT	0.1240431	0.1218599	0.9001897	0.0617431	0.8083734	254	60.047281	669.64327	66.964327	1	0.3700116
HUNGRY	0.5365124	0.1182047	1.0300541	0.0570838	1.0503002	229	54.137116	636.24491	63.624491	2	0.7034776
WHLDAY	1.6727126	0.119633	1.0409061	0.0497944	1.1386965	119	28.132388	532.04294	53.204294	4	4.1036036

Raw-score	Severity	Error	N cases	W cases
0	-2.933649	1.47675	173	167.0761
1	-2.147299	1.0919157	40	37.798062
2	-1.248337	0.848967	24	27.071671
3	-0.603598	0.7703593	41	40.56089
4	-0.028757	0.7539025	46	42.56548
5	0.5559661	0.7832545	62	56.009631
6	1.2318933	0.8739319	80	80.7166
7	2.186977	1.1233237	130	128.60803

3. Data included in the package from Gallup World Poll® 2013-2014

VoH receives data collected via the Gallup World Poll ® (GWP), including the FIES, from 150 countries every year. In the “RM.weights” package, four sample datasets from the GWP are uploaded. These datasets are named `data.FAO_country1`, `data.FAO_country2`, `data.FAO_country3`, `data.FAO_country4`. The datasets include the FIES data, sampling weights, and other demographic variables. For additional information on the data, you can use the help as follows:

`?data.FAO_country1`



```
> ?data.FAO_country1
> |
```

data.FAO_country1 {RM.weights} R Documentation

Food insecurity data for a GWP country (Country1).

Description

The dataset includes the FIES data (Food Insecurity Experience based Scale), sampling weights, and some demographic variables for a sample country (Country1). Data have been collected by the Gallup World Poll.

Arguments

`WORRIED`	FIES question n. 1. During the past 12 months, was there a time when you were worried you would run out of food because of lack of money or other resources?
`HEALTHY`	FIES question n. 2. During the past 12 months, was there a time when you were unable to eat healty and nutritious food because of lack of money or other resources?
`FEWFOOD`	FIES question n. 3. During the past 12 months, was there a time when you ate only few kinds of food because of lack of money or other resources?
`SKIPPED`	FIES question n. 4. During the past 12 months, was there a time when you had to skip a meal because of lack of money or other resources?
`ATELESS`	FIES question n. 5. During the past 12 months, was there a time when you

To attach the data and extract the FIES and the sampling weights, use the following code:

```
data(data.FAO_country1)
XX.country1 = data.FAO_country1[,1:8]
wt.country1 = data.FAO_country1$wt
```

Data can be explored with the function `tab.weight`.

3.1. Descriptives: `tab.weight`

This function computes the main descriptive tables, weighted and unweighted, for the FIES scale items and respondents. It can also be used to compute simple and cross tables for external demographic and geographic variables.

Usage

```
tab.weight(variab, wt, XX = NULL)
```

Arguments

`variab`	User-specified variable considered for weighted tabulation. It could be a single variable, or a list of two variables, of `factor` type, and `length(var.extern)` must be equal to `nrow(XX)`.
`wt`	Vector of sampling weights. The length must be the same as the number of rows of `.data`. If left unspecified, all the individuals will be weighted equally (`.w = rep(1, nrow(.data))`).
`XX`	Input 0/1 data matrix or data frame; affirmative responses must be coded as 1. Rows represent individuals, columns represent items. Missing values are inserted as NA.

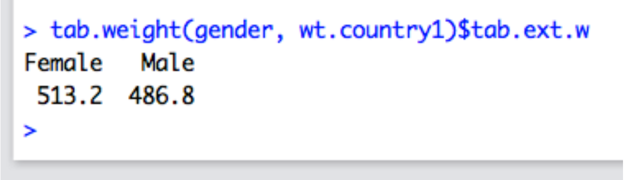
Examples

Set the data

```
data(data.FAO_country1)
XX.country1 = data.FAO_country1[,1:8]
wt.country1 = data.FAO_country1$wt
gender = data.FAO_country1$gender
urbanrural = data.FAO_country1$urbanrural
```

Univariate weighted table by gender

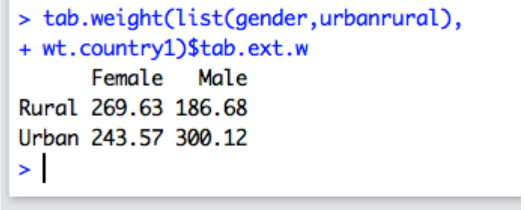
```
tab.weight(gender, wt.country1)$tab.ext.w
```



```
> tab.weight(gender, wt.country1)$tab.ext.w
Female   Male
 513.2  486.8
>
```

Bivariate weighted table by gender and urban/rural

```
tab.weight(list(gender,urbanrural),wt.country1)$tab.ext.w
```



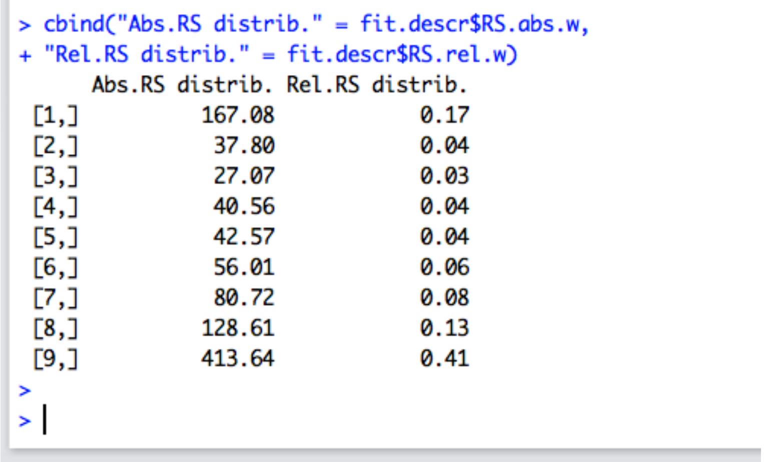
```
> tab.weight(list(gender,urbanrural),
+ wt.country1)$tab.ext.w
      Female   Male
Rural 269.63 186.68
Urban 243.57 300.12
> |
```

Calculate Rasch descriptives

```
fit.descr = tab.weight(wt = wt.country1, XX =
XX.country1)
```

Display weighted distribution across raw-scores (absolute and relative):

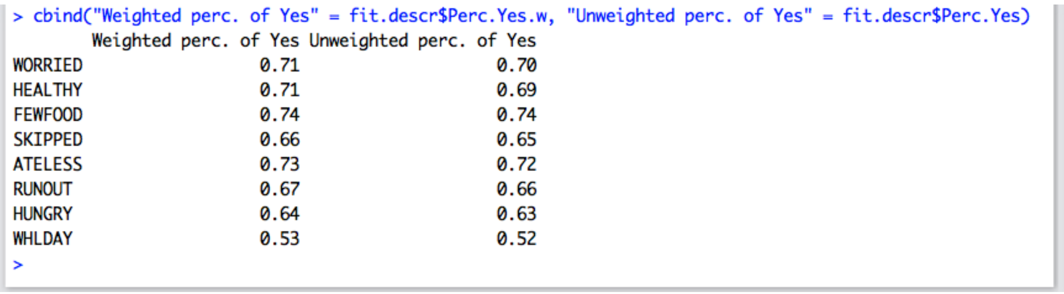
```
cbind("Abs.RS distrib." = fit.descr$RS.abs.w,
"Rel.RS distrib." = fit.descr$RS.rel.w)
```



```
> cbind("Abs.RS distrib." = fit.descr$RS.abs.w,
+ "Rel.RS distrib." = fit.descr$RS.rel.w)
      Abs.RS distrib. Rel.RS distrib.
 [1,]          167.08            0.17
 [2,]           37.80            0.04
 [3,]           27.07            0.03
 [4,]           40.56            0.04
 [5,]           42.57            0.04
 [6,]           56.01            0.06
 [7,]           80.72            0.08
 [8,]          128.61            0.13
 [9,]          413.64            0.41
>
> |
```

Display weighted and unweighted percentage of Yes per item:

```
cbind("Weighted perc. of Yes" = fit.descr$Perc.Yes.w,
      "Unweighted perc. of Yes" = fit.descr$Perc.Yes)
```

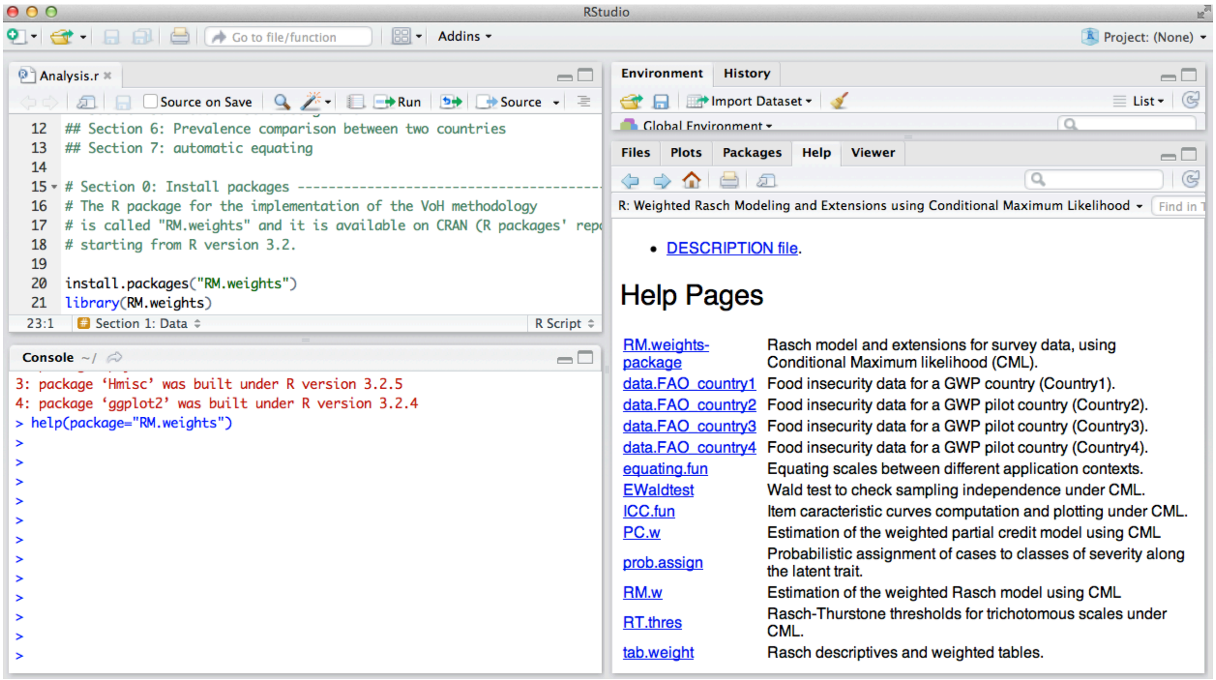


```
> cbind("Weighted perc. of Yes" = fit.descr$Perc.Yes.w, "Unweighted perc. of Yes" = fit.descr$Perc.Yes)
        Weighted perc. of Yes Unweighted perc. of Yes
WORRIED                  0.71                    0.70
HEALTHY                  0.71                    0.69
FEWFOOD                  0.74                    0.74
SKIPPED                  0.66                    0.65
ATELESS                  0.73                    0.72
RUNOUT                   0.67                    0.66
HUNGRY                   0.64                    0.63
WHLDAY                   0.53                    0.52
>
```

4. Additional functions in the RM.weights package

The package includes many other functions that can be displayed using the command:

```
help(package="RM.weights")
```



```
## Section 6: Prevalence comparison between two countries
## Section 7: automatic equating

# Section 0: Install packages -----------------------------------------
# The R package for the implementation of the VoH methodology
# is called "RM.weights" and it is available on CRAN (R packages' repo
# starting from R version 3.2.

install.packages("RM.weights")
library(RM.weights)
```

```
3: package ‘Hmisc’ was built under R version 3.2.5
4: package ‘ggplot2’ was built under R version 3.2.4
> help(package="RM.weights")
>
>
>
>
>
>
>
>
>
>
>
>
```

- DESCRIPTION file.

Help Pages

RM.weights-package	Rasch model and extensions for survey data, using Conditional Maximum likelihood (CML).
data.FAO_country1	Food insecurity data for a GWP country (Country1).
data.FAO_country2	Food insecurity data for a GWP pilot country (Country2).
data.FAO_country3	Food insecurity data for a GWP pilot country (Country3).
data.FAO_country4	Food insecurity data for a GWP pilot country (Country4).
equating.fun	Equating scales between different application contexts.
EWaldtest	Wald test to check sampling independence under CML.
ICC.fun	Item caracteristic curves computation and plotting under CML.
PC.w	Estimation of the weighted partial credit model using CML
prob.assign	Probabilistic assignment of cases to classes of severity along the latent trait.
RM.w	Estimation of the weighted Rasch model using CML
RT.thres	Rasch-Thurstone thresholds for trichotomous scales under CML.
tab.weight	Rasch descriptives and weighted tables.

Of particular interest is the function `prob.assign`, that can be used to derive prevalence of food insecurity using the same methodology of the VoH project. The code for this procedure can be found in the “Analysis.r” file, Section 5.

Section 5 in the “Analysis.r” file is preparatory for Section 6, which shows how to calculate comparable prevalence estimates between countries or sub-groups (for example languages) within a country. The first phase of this equating process is also reported in file “Equating.xlsx”. The following section will describe an example of discrete and probabilistic assignment and of the equating.

5. An example of discrete and probabilistic assignment and some principles of equating

Note: Some principles of equating are also included in the file “Equating.xlsx”

In this Section we will illustrate how to assign cases to food insecurity classes deterministically and probabilistically and we will briefly show some applications of the equating procedure to calculate comparable prevalence rates.

Classification of cases in food insecurity classes can be pursued in two ways.
The deterministic classification can be performed by setting thresholds in terms of raw score. The raw score is the sum of affirmative answers given to the 8 FIES items by each interviewed subject and can be calculated in R as follows:

```
rs.country1 = rowSums(XX.country1)
```

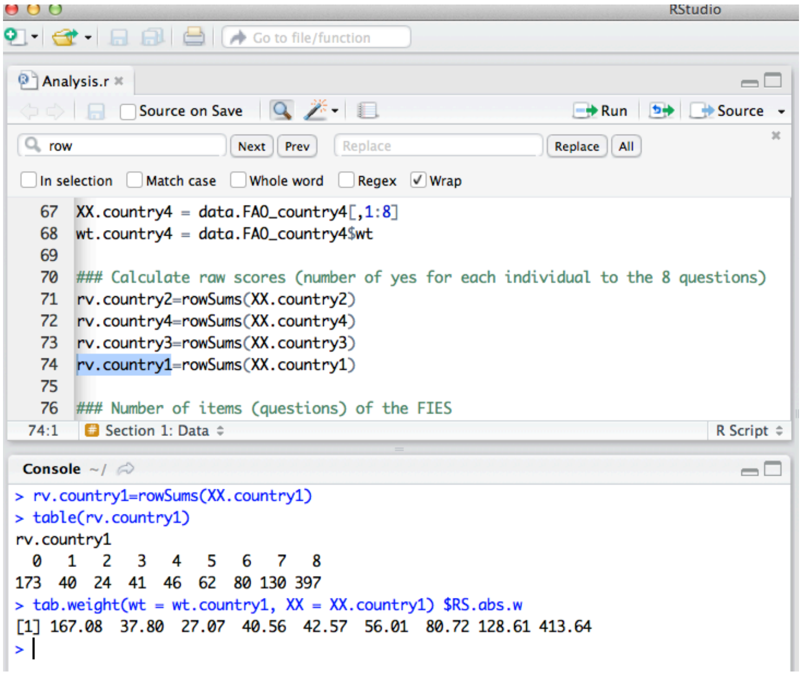
(where `XX.country1` is the matrix that includes the 0/1 answers to the FIES)

To display the (unweighted) distribution of individuals reporting a given raw score, you can type

```
table(rs.country1)
```

while for the weighted distribution you can use the “tab.weight” function (see Section 3 of this document)

```
tab.weight(wt = wt.country1, XX = XX.country1)$RS.abs.w
```



row

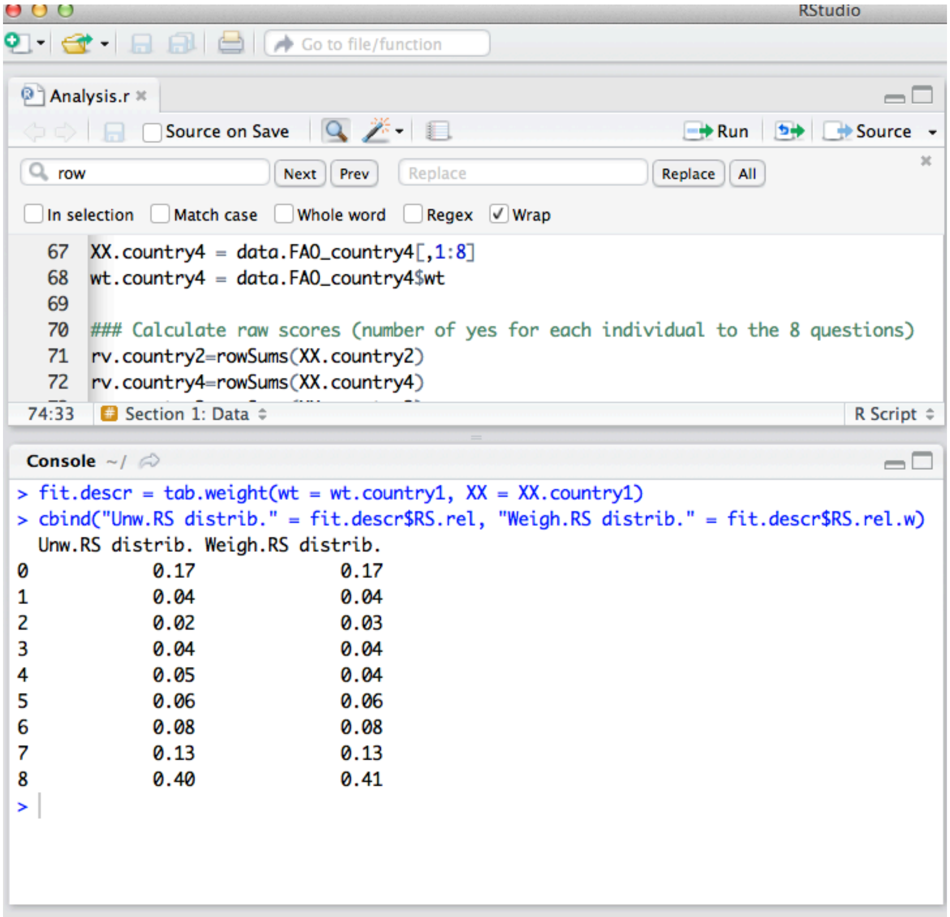
☐ In selection ☐ Match case ☐ Whole word ☐ Regex ☑ Wrap

```
XX.country4 = data.FAO_country4[,1:8]
wt.country4 = data.FAO_country4$wt

### Calculate raw scores (number of yes for each individual to the 8 questions)
rv.country2=rowSums(XX.country2)
rv.country4=rowSums(XX.country4)
rv.country3=rowSums(XX.country3)
rv.country1=rowSums(XX.country1)

### Number of items (questions) of the FIES
```

```
> rv.country1=rowSums(XX.country1)
> table(rv.country1)
rv.country1
  0   1   2   3   4   5   6   7   8 
173  40  24  41  46  62  80 130 397 
> tab.weight(wt = wt.country1, XX = XX.country1) $RS.abs.w
[1] 167.08  37.80  27.07  40.56  42.57  56.01  80.72 128.61 413.64
> 
```

Relative distribution across raw scores can also be calculated with the “tab.weight” function:

```
fit.descr = tab.weight(wt = wt.country1, XX = XX.country1)

cbind("Unw.RS distrib." = fit.descr$RS.rel,
"Weigh.RS distrib." = fit.descr$RS.rel.w)
```



☐ In selection ☐ Match case ☐ Whole word ☐ Regex ☑ Wrap

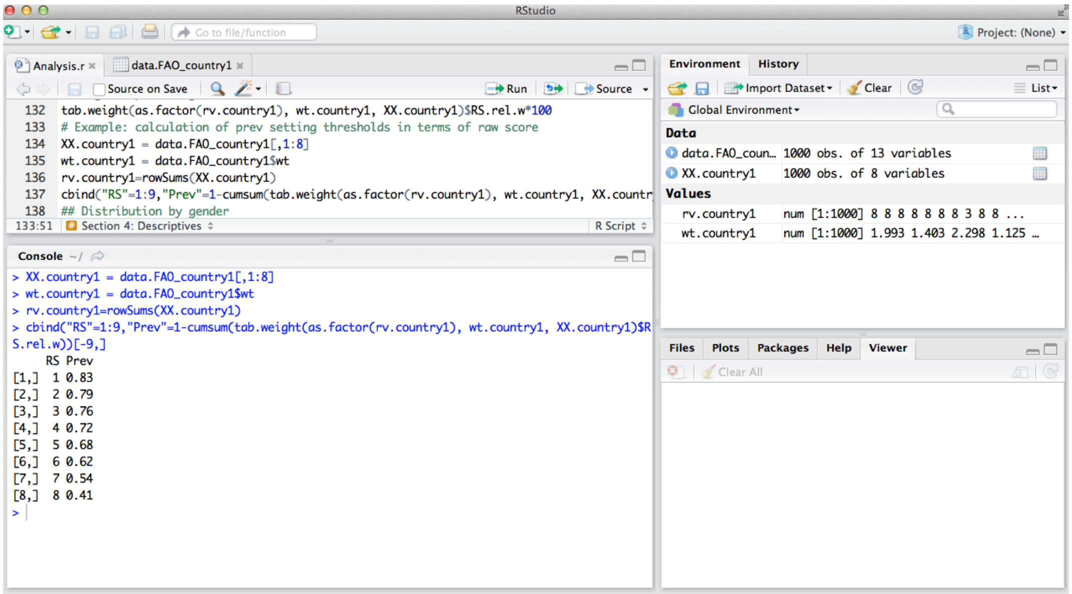
```
XX.country4 = data.FAO_country4[,1:8]
wt.country4 = data.FAO_country4$wt

### Calculate raw scores (number of yes for each individual to the 8 questions)
rv.country2=rowSums(XX.country2)
rv.country4=rowSums(XX.country4)
```

```
> fit.descr = tab.weight(wt = wt.country1, XX = XX.country1)
> cbind("Unw.RS distrib." = fit.descr$RS.rel, "Weigh.RS distrib." = fit.descr$RS.rel.w)
  Unw.RS distrib. Weigh.RS distrib.
0            0.17              0.17
1            0.04              0.04
2            0.02              0.03
3            0.04              0.04
4            0.05              0.04
5            0.06              0.06
6            0.08              0.08
7            0.13              0.13
8            0.40              0.41
> 
```

This relative distribution can be used to calculate the food insecurity prevalence rates at different levels.

First, one minus the cumulative distribution across all levels of raw score is calculated:

```
XX.country1 = data.FAO_country1[,1:8]

wt.country1 = data.FAO_country1$wt

rv.country1=rowSums(XX.country1)

cbind("RS"=1:9,"Prev"=1-cumsum(tab.weight(as.factor(rv.country1), wt.country1, XX.country1)$RS.rel.w))[-9,]
```



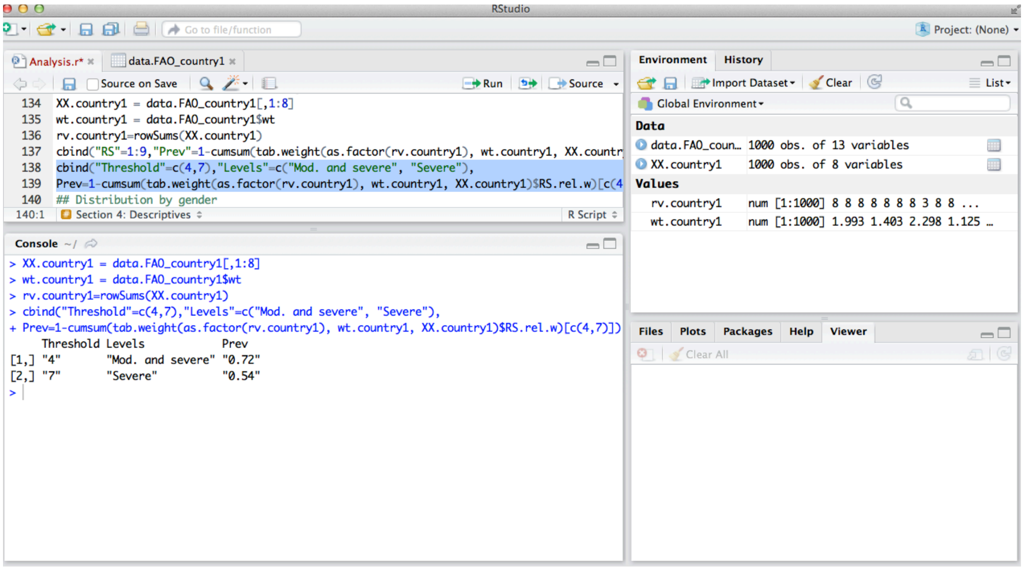
```r
tab.weight(as.factor(rv.country1), wt.country1, XX.country1)$RS.rel.w*100
# Example: calculation of prev setting thresholds in terms of raw score
XX.country1 = data.FAO_country1[,1:8]
wt.country1 = data.FAO_country1$wt
rv.country1=rowSums(XX.country1)
cbind("RS"=1:9,"Prev"=1-cumsum(tab.weight(as.factor(rv.country1), wt.country1, XX.countr
## Distribution by gender
```

```
> XX.country1 = data.FAO_country1[,1:8]
> wt.country1 = data.FAO_country1$wt
> rv.country1=rowSums(XX.country1)
> cbind("RS"=1:9,"Prev"=1-cumsum(tab.weight(as.factor(rv.country1), wt.country1, XX.country1)$R
S.rel.w))[-9,]
     RS Prev
[1,]  1 0.83
[2,]  2 0.79
[3,]  3 0.76
[4,]  4 0.72
[5,]  5 0.68
[6,]  6 0.62
[7,]  7 0.54
[8,]  8 0.41
> 
```

```
Data
data.FAO_coun…  1000 obs. of 13 variables
XX.country1     1000 obs. of 8 variables
Values
rv.country1     num [1:1000] 8 8 8 8 8 8 8 3 8 8 ...
wt.country1     num [1:1000] 1.993 1.403 2.298 1.125 …
```

Then, setting a threshold of, for example, 4 to classify “moderate or severe” food insecure, and of 7 to classify “severe” food insecure subjects (or households), will lead to the following prevalence rates:

```
cbind("Threshold"=c(4,7),"Levels"=c("Mod. or severe", "Severe"),

Prev=1-cumsum(tab.weight(as.factor(rv.country1), wt.country1, XX.country1)$RS.rel.w)[c(4,7)])
```

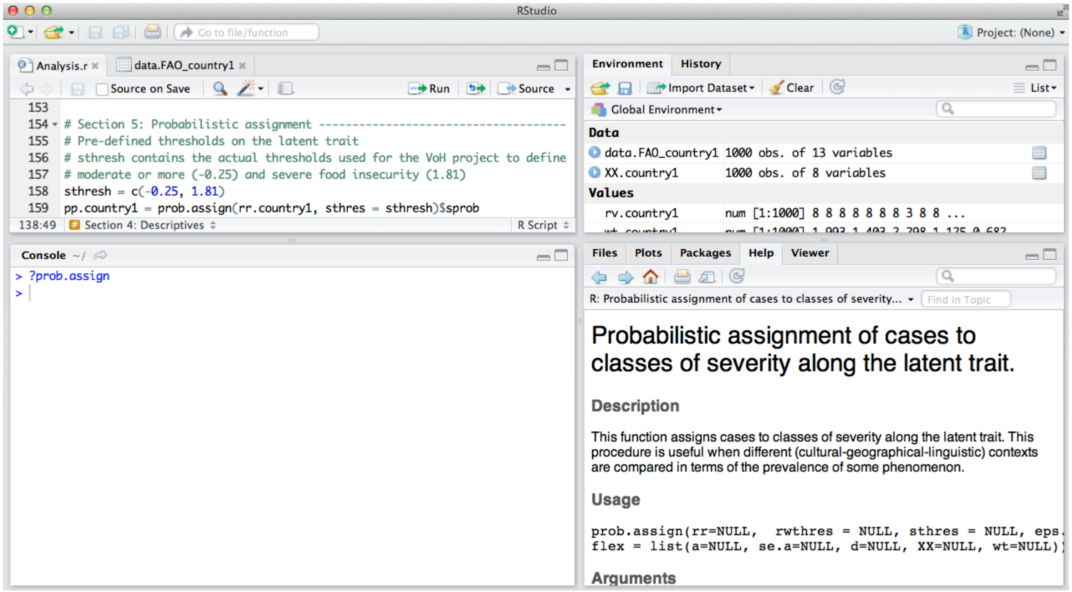


```r
XX.country1 = data.FAO_country1[,1:8]
wt.country1 = data.FAO_country1$wt
rv.country1=rowSums(XX.country1)
cbind("RS"=1:9,"Prev"=1-cumsum(tab.weight(as.factor(rv.country1), wt.country1, XX.countr
cbind("Threshold"=c(4,7),"Levels"=c("Mod. and severe", "Severe"),
Prev=1-cumsum(tab.weight(as.factor(rv.country1), wt.country1, XX.country1)$RS.rel.w)[c(4
## Distribution by gender
```

```
> XX.country1 = data.FAO_country1[,1:8]
> wt.country1 = data.FAO_country1$wt
> rv.country1=rowSums(XX.country1)
> cbind("Threshold"=c(4,7),"Levels"=c("Mod. and severe", "Severe"),
+ Prev=1-cumsum(tab.weight(as.factor(rv.country1), wt.country1, XX.country1)$RS.rel.w)[c(4,7)])
     Threshold Levels            Prev
[1,] "4"       "Mod. and severe" "0.72"
[2,] "7"       "Severe"          "0.54"
> 
```

```
Data
data.FAO_coun…  1000 obs. of 13 variables
XX.country1     1000 obs. of 8 variables
Values
rv.country1     num [1:1000] 8 8 8 8 8 8 8 3 8 8 ...
wt.country1     num [1:1000] 1.993 1.403 2.298 1.125 …
```

The probabilistic classification (or **probabilistic assignment**), on the other hand, can be performed by setting thresholds in terms of latent trait. The advantage of using this approach is that, after a procedure of equating, it can be used to calculate comparable prevalence rates across countries. In the “RM.weights” package, the function to perform the probabilistic assignment is called “`prob.assign`” (type `?prob.assign` in `R` to see the help):



```
# Section 5: Probabilistic assignment ------------------------------------
# Pre-defined thresholds on the latent trait
# sthresh contains the actual thresholds used for the VoH project to define
# moderate or more (-0.25) and severe food insecurity (1.81)
sthresh = c(-0.25, 1.81)
pp.country1 = prob.assign(rr.country1, sthres = sthresh)$sprob
```

```
> ?prob.assign
>
```

Data

```
data.FAO_country1  1000 obs. of 13 variables
XX.country1        1000 obs. of 8 variables
```

Values

```
rv.country1        num [1:1000] 8 8 8 8 8 8 8 3 8 8 ...
```

Probabilistic assignment of cases to classes of severity along the latent trait.

Description

This function assigns cases to classes of severity along the latent trait. This procedure is useful when different (cultural-geographical-linguistic) contexts are compared in terms of the prevalence of some phenomenon.

Usage

```
prob.assign(rr=NULL,  rwthres = NULL, sthres = NULL, eps.
flex = list(a=NULL, se.a=NULL, d=NULL, XX=NULL, wt=NULL))
```

Arguments

For example, to calculate the probability, for a given country, of being beyond pre-determined thresholds on the latent trait one can write the following code:

```
sthresh = c(-0.25, 1.81)
```

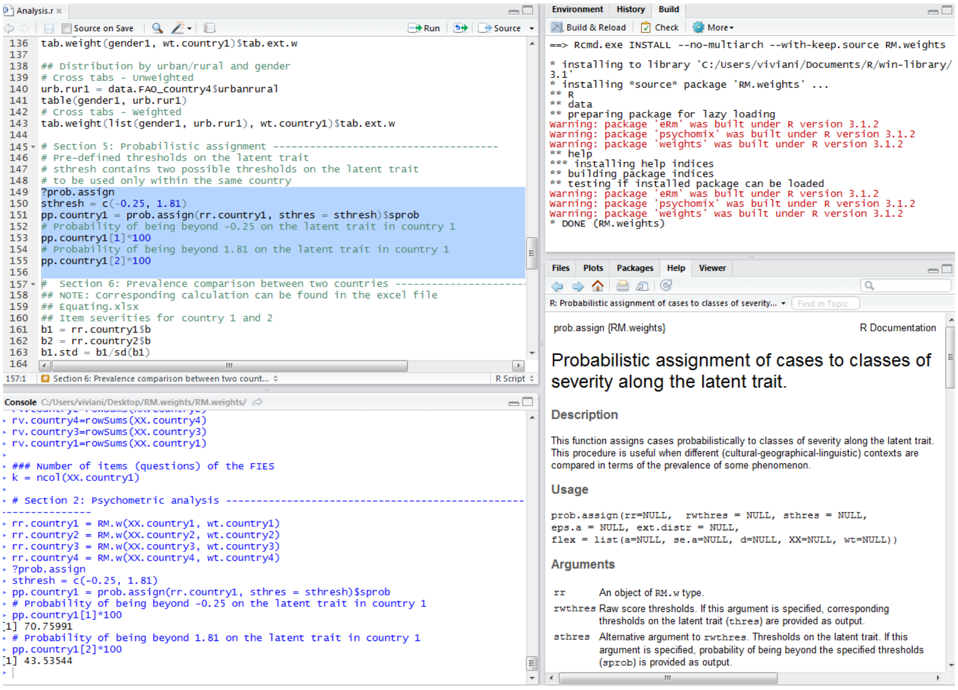
```
pp.country1 = prob.assign(rr.country1, sthres = sthresh)$sprob
```

The probability of being beyond -0.25 on the latent trait in country 1 is

```
pp.country1[1]*100
```

while the probability of being beyond 1.81 on the latent trait in country 1 is

```
pp.country1[2]*100
```



```
tab.weight(gender1, wt.country1)$tab.ext.w

## Distribution by urban/rural and gender
# Cross tabs - Unweighted
urb.rur1 = data.FAO_country4$urbanrural
table(gender1, urb.rur1)
# Cross tabs - Weighted
tab.weight(list(gender1, urb.rur1), wt.country1)$tab.ext.w

# Section 5: Probabilistic assignment ------------------------------------
# Pre-defined thresholds on the latent trait
# sthresh contains two possible thresholds on the latent trait
# to be used only within the same country
?prob.assign
sthresh = c(-0.25, 1.81)
pp.country1 = prob.assign(rr.country1, sthres = sthresh)$sprob
# Probability of being beyond -0.25 on the latent trait in country 1
pp.country1[1]*100
# Probability of being beyond 1.81 on the latent trait in country 1
pp.country1[2]*100

#  Section 6: Prevalence comparison between two countries -----------------
## NOTE: Corresponding calculation can be found in the excel file
## Equating.xlsx
## Item severities for country 1 and 2
b1 = rr.country1$b
b2 = rr.country2$b
b1.std = b1/sd(b1)
```

```
> rv.country4=rowSums(XX.country4)
> rv.country3=rowSums(XX.country3)
> rv.country1=rowSums(XX.country1)
>
> ### Number of items (questions) of the FIES
> k = ncol(XX.country1)
>
> # Section 2: Psychometric analysis --------------------------------------------
---------------
> rr.country1 = RM.w(XX.country1, wt.country1)
> rr.country2 = RM.w(XX.country2, wt.country2)
> rr.country3 = RM.w(XX.country3, wt.country3)
> rr.country4 = RM.w(XX.country4, wt.country4)
> ?prob.assign
> sthresh = c(-0.25, 1.81)
> pp.country1 = prob.assign(rr.country1, sthres = sthresh)$sprob
> # Probability of being beyond -0.25 on the latent trait in country 1
> pp.country1[1]*100
[1] 70.75991
> # Probability of being beyond 1.81 on the latent trait in country 1
> pp.country1[2]*100
[1] 43.53544
>
```

```
==> Rcmd.exe INSTALL --no-multiarch --with-keep.source RM.weights

* installing to library 'C:/Users/viviani/Documents/R/win-library/
3.1'
* installing *source* package 'RM.weights' ...
** R
** data
** preparing package for lazy loading
Warning: package 'eRm' was built under R version 3.1.2
Warning: package 'psychomix' was built under R version 3.1.2
Warning: package 'weights' was built under R version 3.1.2
** help
*** installing help indices
** building package indices
** testing if installed package can be loaded
Warning: package 'eRm' was built under R version 3.1.2
Warning: package 'psychomix' was built under R version 3.1.2
Warning: package 'weights' was built under R version 3.1.2
* DONE (RM.weights)
```

prob.assign {RM.weights} R Documentation

Probabilistic assignment of cases to classes of severity along the latent trait.

Description

This function assigns cases probabilistically to classes of severity along the latent trait. This procedure is useful when different (cultural-geographical-linguistic) contexts are compared in terms of the prevalence of some phenomenon.

Usage

```
prob.assign(rr=NULL,  rwthres = NULL, sthres = NULL,
eps.a = NULL, ext.distr = NULL,
flex = list(a=NULL, se.a=NULL, d=NULL, XX=NULL, wt=NULL))
```

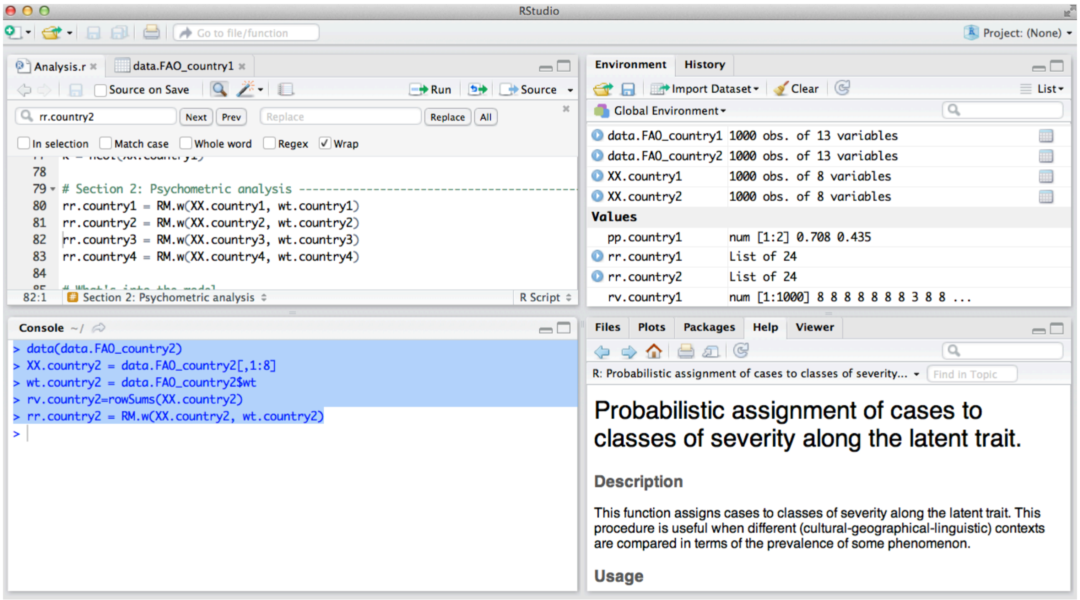
Arguments

`rr`	An object of `RM.w` type.
`rwthres`	Raw score thresholds. If this argument is specified, corresponding thresholds on the latent trait (`thres`) are provided as output.
`sthres`	Alternative argument to `rwthres`. Thresholds on the latent trait. If this argument is specified, probability of being beyond the specified thresholds (`sprob`) is provided as output.

Now let us assume that we want to calculate a comparable prevalence rate of food insecurity between country 1 and country 2.

We load the data for country 2 and fit the Rasch model:

```
data(data.FAO_country2)

XX.country2 = data.FAO_country2[,1:8]

wt.country2 = data.FAO_country2$wt

rv.country2=rowSums(XX.country2)

rr.country2 = RM.w(XX.country2, wt.country2)
```



```
# Section 2: Psychometric analysis ------------------------------------
rr.country1 = RM.w(XX.country1, wt.country1)
rr.country2 = RM.w(XX.country2, wt.country2)
rr.country3 = RM.w(XX.country3, wt.country3)
rr.country4 = RM.w(XX.country4, wt.country4)
```

```
> data(data.FAO_country2)
> XX.country2 = data.FAO_country2[,1:8]
> wt.country2 = data.FAO_country2$wt
> rv.country2=rowSums(XX.country2)
> rr.country2 = RM.w(XX.country2, wt.country2)
>
```

```
data.FAO_country1 1000 obs. of 13 variables
data.FAO_country2 1000 obs. of 13 variables
XX.country1       1000 obs. of 8 variables
XX.country2       1000 obs. of 8 variables
Values
pp.country1       num [1:2] 0.708 0.435
rr.country1       List of 24
rr.country2       List of 24
rv.country1       num [1:1000] 8 8 8 8 8 8 8 3 8 8 ...
```

Probabilistic assignment of cases to classes of severity along the latent trait.

Description

This function assigns cases to classes of severity along the latent trait. This procedure is useful when different (cultural-geographical-linguistic) contexts are compared in terms of the prevalence of some phenomenon.

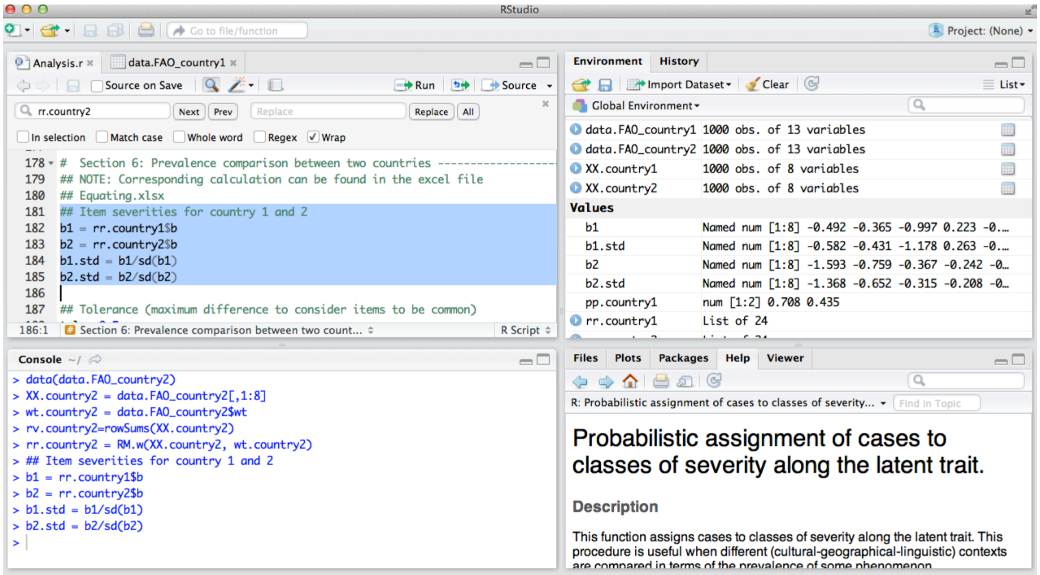
Usage

We then define a common metric between country 1 and country 2 based on item severities (b1 and b2 in the following code):

```
b1 = rr.country1$b

b2 = rr.country2$b
```

The first step is to standardize item severities to a standard deviation of 1:

```
b1.std = b1/sd(b1)

b2.std = b2/sd(b2)
```



```
#  Section 6: Prevalence comparison between two countries ------------------
## NOTE: Corresponding calculation can be found in the excel file
## Equating.xlsx
## Item severities for country 1 and 2
b1 = rr.country1$b
b2 = rr.country2$b
b1.std = b1/sd(b1)
b2.std = b2/sd(b2)

## Tolerance (maximum difference to consider items to be common)
```

```
> data(data.FAO_country2)
> XX.country2 = data.FAO_country2[,1:8]
> wt.country2 = data.FAO_country2$wt
> rv.country2=rowSums(XX.country2)
> rr.country2 = RM.w(XX.country2, wt.country2)
> ## Item severities for country 1 and 2
> b1 = rr.country1$b
> b2 = rr.country2$b
> b1.std = b1/sd(b1)
> b2.std = b2/sd(b2)
> 
```

```
data.FAO_country1 1000 obs. of 13 variables
data.FAO_country2 1000 obs. of 13 variables
XX.country1       1000 obs. of 8 variables
XX.country2       1000 obs. of 8 variables
Values
b1                Named num [1:8] -0.492 -0.365 -0.997 0.223 -0.…
b1.std            Named num [1:8] -0.582 -0.431 -1.178 0.263 -0.…
b2                Named num [1:8] -1.593 -0.759 -0.367 -0.242 -0…
b2.std            Named num [1:8] -1.368 -0.652 -0.315 -0.208 -0…
pp.country1       num [1:2] 0.708 0.435
rr.country1       List of 24
```

R: Probabilistic assignment of cases to classes of severity...

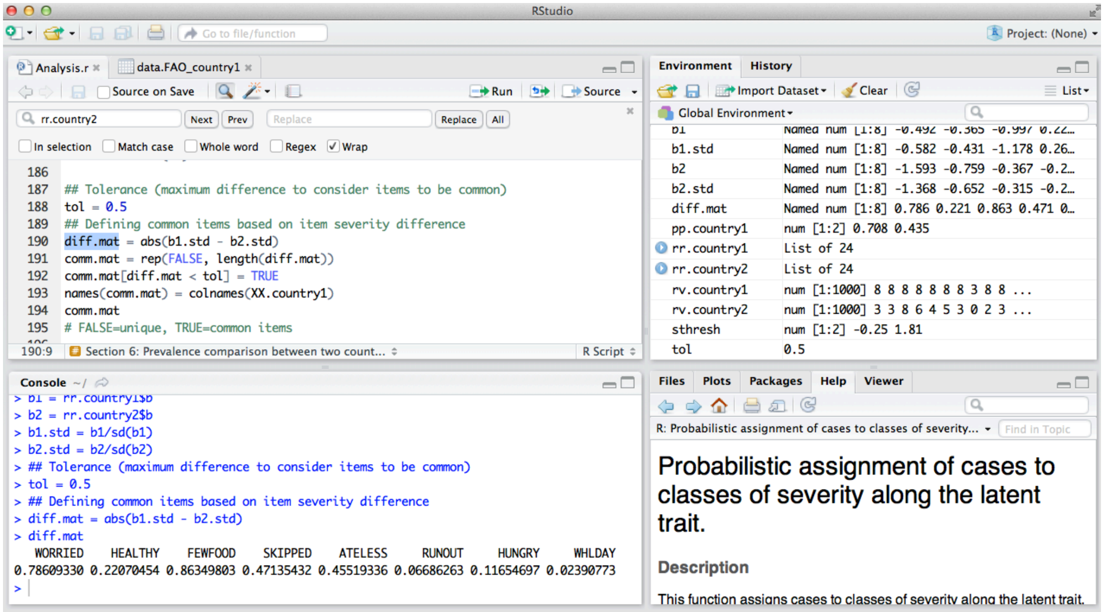
Probabilistic assignment of cases to classes of severity along the latent trait.

Description

This function assigns cases to classes of severity along the latent trait. This procedure is useful when different (cultural-geographical-linguistic) contexts are compared in terms of the prevalence of some phenomenon

We calculate the difference between the standardized item severities and compare them with a set threshold:

```
tol = 0.5

diff.mat = abs(b1.std - b2.std)

diff.mat
```



```

## Tolerance (maximum difference to consider items to be common)
tol = 0.5
## Defining common items based on item severity difference
diff.mat = abs(b1.std - b2.std)
comm.mat = rep(FALSE, length(diff.mat))
comm.mat[diff.mat < tol] = TRUE
names(comm.mat) = colnames(XX.country1)
comm.mat
# FALSE=unique, TRUE=common items
```

```
> b1 = rr.country1$b
> b2 = rr.country2$b
> b1.std = b1/sd(b1)
> b2.std = b2/sd(b2)
> ## Tolerance (maximum difference to consider items to be common)
> tol = 0.5
> ## Defining common items based on item severity difference
> diff.mat = abs(b1.std - b2.std)
> diff.mat
   WORRIED    HEALTHY    FEWFOOD    SKIPPED    ATELESS     RUNOUT     HUNGRY     WHLDAY 
0.78609330 0.22070454 0.86349803 0.47135432 0.45519336 0.06686263 0.11654697 0.02390773 
> 
```

```
b1                Named num [1:8] -0.492 -0.365 -0.997 0.22…
b1.std            Named num [1:8] -0.582 -0.431 -1.178 0.26…
b2                Named num [1:8] -1.593 -0.759 -0.367 -0.2…
b2.std            Named num [1:8] -1.368 -0.652 -0.315 -0.2…
diff.mat          Named num [1:8] 0.786 0.221 0.863 0.471 0…
pp.country1       num [1:2] 0.708 0.435
rr.country1       List of 24
rr.country2       List of 24
rv.country1       num [1:1000] 8 8 8 8 8 8 8 3 8 8 ...
rv.country2       num [1:1000] 3 3 8 6 4 5 3 0 2 3 ...
sthresh           num [1:2] -0.25 1.81
tol               0.5
```

R: Probabilistic assignment of cases to classes of severity...

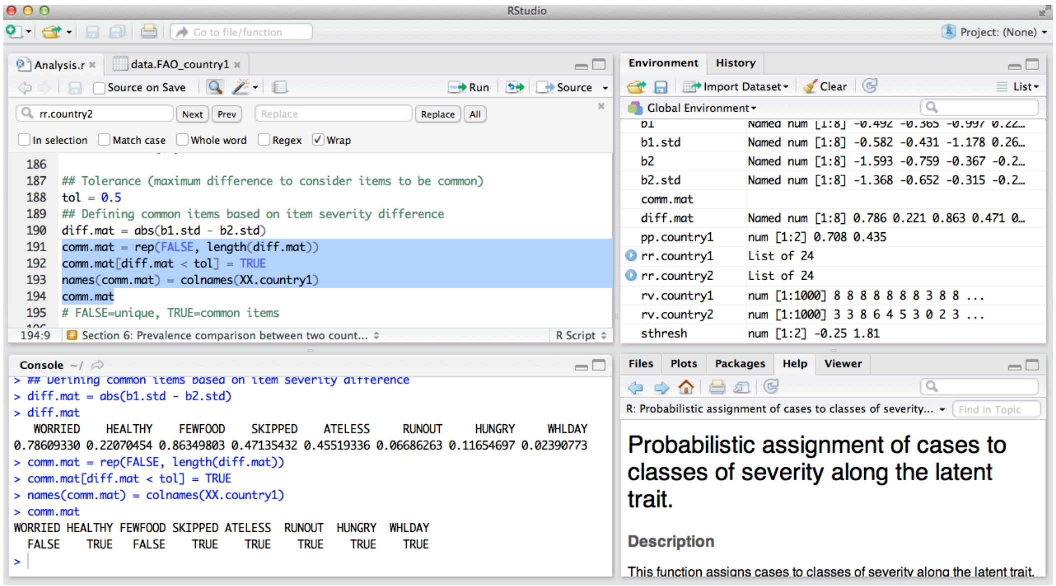
Probabilistic assignment of cases to classes of severity along the latent trait.

Description

This function assigns cases to classes of severity along the latent trait.

The vector “`comm.mat`” will be TRUE if the difference between standardized severities is less than the set tolerance, and false otherwise:

```
comm.mat = rep(FALSE, length(diff.mat))

comm.mat[diff.mat < tol] = TRUE

names(comm.mat) = colnames(XX.country1)

comm.mat
```



```r

## Tolerance (maximum difference to consider items to be common)
tol = 0.5
## Defining common items based on item severity difference
diff.mat = abs(b1.std - b2.std)
comm.mat = rep(FALSE, length(diff.mat))
comm.mat[diff.mat < tol] = TRUE
names(comm.mat) = colnames(XX.country1)
comm.mat
# FALSE=unique, TRUE=common items
```

```
> ## Defining common items based on item severity difference
> diff.mat = abs(b1.std - b2.std)
> diff.mat
   WORRIED    HEALTHY    FEWFOOD    SKIPPED    ATELESS     RUNOUT     HUNGRY     WHLDAY 
0.78609330 0.22070454 0.86349803 0.47135432 0.45519336 0.06686263 0.11654697 0.02390773 
> comm.mat = rep(FALSE, length(diff.mat))
> comm.mat[diff.mat < tol] = TRUE
> names(comm.mat) = colnames(XX.country1)
> comm.mat
WORRIED HEALTHY FEWFOOD SKIPPED ATELESS  RUNOUT  HUNGRY  WHLDAY 
  FALSE    TRUE   FALSE    TRUE    TRUE    TRUE    TRUE    TRUE 
> 
```

```
b1            Named num [1:8] -0.492 -0.365 -0.997 0.22…
b1.std        Named num [1:8] -0.582 -0.431 -1.178 0.26…
b2            Named num [1:8] -1.593 -0.759 -0.367 -0.2…
b2.std        Named num [1:8] -1.368 -0.652 -0.315 -0.2…
comm.mat
diff.mat      Named num [1:8] 0.786 0.221 0.863 0.471 0…
pp.country1   num [1:2] 0.708 0.435
rr.country1   List of 24
rr.country2   List of 24
rv.country1   num [1:1000] 8 8 8 8 8 8 8 3 8 8 ...
rv.country2   num [1:1000] 3 3 8 6 4 5 3 0 2 3 ...
sthresh       num [1:2] -0.25 1.81
```

Probabilistic assignment of cases to classes of severity along the latent trait.

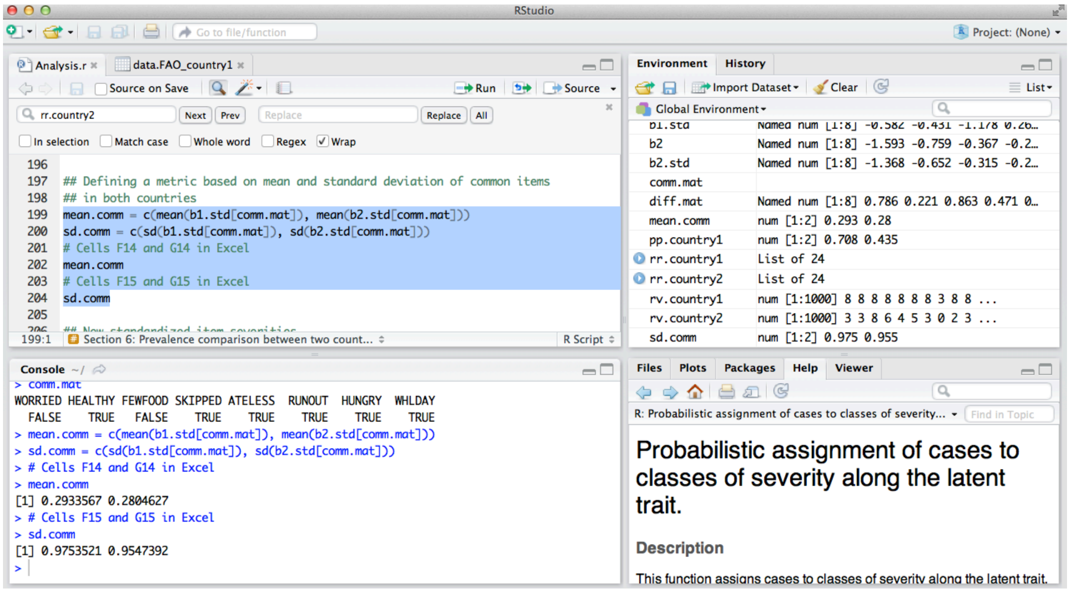
Description

This function assigns cases to classes of severity along the latent trait.

In this example, the unique (i.e. not common) items are “WORRIED” and “FEWFOOD”.

Using the “`comm.mat`” vector, we define a metric based on mean and standard deviation of common items in both countries:

```r
mean.comm = c(mean(b1.std[comm.mat]), mean(b2.std[comm.mat]))

sd.comm = c(sd(b1.std[comm.mat]), sd(b2.std[comm.mat]))

mean.comm

sd.comm
```



```r

## Defining a metric based on mean and standard deviation of common items
## in both countries
mean.comm = c(mean(b1.std[comm.mat]), mean(b2.std[comm.mat]))
sd.comm = c(sd(b1.std[comm.mat]), sd(b2.std[comm.mat]))
# Cells F14 and G14 in Excel
mean.comm
# Cells F15 and G15 in Excel
sd.comm

## New standardized item severities
```

```
> comm.mat
WORRIED HEALTHY FEWFOOD SKIPPED ATELESS  RUNOUT  HUNGRY  WHLDAY 
  FALSE    TRUE   FALSE    TRUE    TRUE    TRUE    TRUE    TRUE 
> mean.comm = c(mean(b1.std[comm.mat]), mean(b2.std[comm.mat]))
> sd.comm = c(sd(b1.std[comm.mat]), sd(b2.std[comm.mat]))
> # Cells F14 and G14 in Excel
> mean.comm
[1] 0.2933567 0.2804627
> # Cells F15 and G15 in Excel
> sd.comm
[1] 0.9753521 0.9547392
> 
```

```
b1.std        Named num [1:8] -0.582 -0.431 -1.178 0.26…
b2            Named num [1:8] -1.593 -0.759 -0.367 -0.2…
b2.std        Named num [1:8] -1.368 -0.652 -0.315 -0.2…
comm.mat
diff.mat      Named num [1:8] 0.786 0.221 0.863 0.471 0…
mean.comm     num [1:2] 0.293 0.28
pp.country1   num [1:2] 0.708 0.435
rr.country1   List of 24
rr.country2   List of 24
rv.country1   num [1:1000] 8 8 8 8 8 8 8 3 8 8 ...
rv.country2   num [1:1000] 3 3 8 6 4 5 3 0 2 3 ...
sd.comm       num [1:2] 0.975 0.955
```

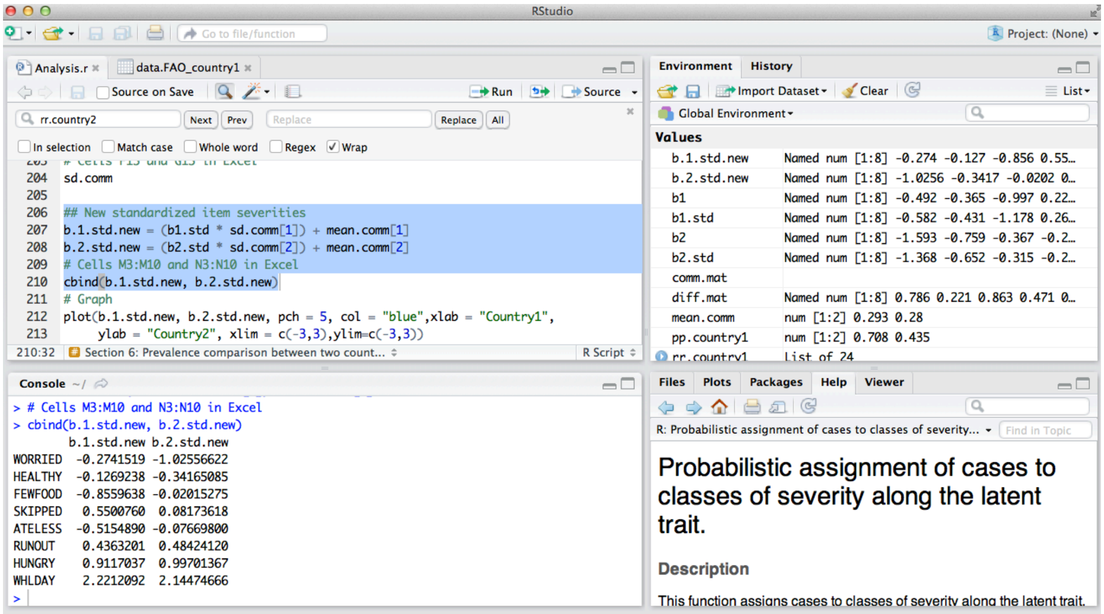
Probabilistic assignment of cases to classes of severity along the latent trait.

Description

This function assigns cases to classes of severity along the latent trait.

The new standardized item severities are also based on this common metric:

```r
b.1.std.new = (b1.std * sd.comm[1]) + mean.comm[1]

b.2.std.new = (b2.std * sd.comm[2]) + mean.comm[2]

cbind(b.1.std.new, b.2.std.new)
```



```r
sd.comm

## New standardized item severities
b.1.std.new = (b1.std * sd.comm[1]) + mean.comm[1]
b.2.std.new = (b2.std * sd.comm[2]) + mean.comm[2]
# Cells M3:M10 and N3:N10 in Excel
cbind(b.1.std.new, b.2.std.new)
# Graph
plot(b.1.std.new, b.2.std.new, pch = 5, col = "blue",xlab = "Country1",
     ylab = "Country2", xlim = c(-3,3),ylim=c(-3,3))
```

```
> # Cells M3:M10 and N3:N10 in Excel
> cbind(b.1.std.new, b.2.std.new)
        b.1.std.new b.2.std.new
WORRIED  -0.2741519 -1.02556622
HEALTHY  -0.1269238 -0.34165085
FEWFOOD  -0.8559638 -0.02015275
SKIPPED   0.5500760  0.08173618
ATELESS  -0.5154890 -0.07669800
RUNOUT    0.4363201  0.48424120
HUNGRY    0.9117037  0.99701367
WHLDAY    2.2212092  2.14474666
>
```

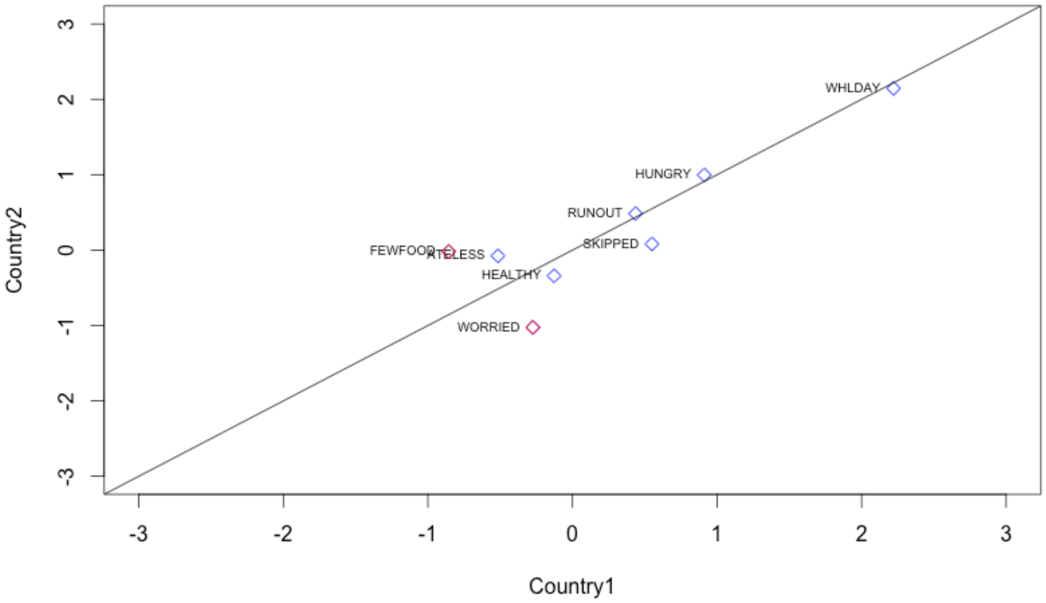
```
b.1.std.new    Named num [1:8] -0.274 -0.127 -0.856 0.55…
b.2.std.new    Named num [1:8] -1.0256 -0.3417 -0.0202 0…
b1             Named num [1:8] -0.492 -0.365 -0.997 0.22…
b1.std         Named num [1:8] -0.582 -0.431 -1.178 0.26…
b2             Named num [1:8] -1.593 -0.759 -0.367 -0.2…
b2.std         Named num [1:8] -1.368 -0.652 -0.315 -0.2…
comm.mat
diff.mat       Named num [1:8] 0.786 0.221 0.863 0.471 0…
mean.comm      num [1:2] 0.293 0.28
pp.country1    num [1:2] 0.708 0.435
rr.country1    List of 24
```

Probabilistic assignment of cases to classes of severity along the latent trait.

Description

This function assigns cases to classes of severity along the latent trait.

These new standardized severities are plotted below (blue dots are common, red dots are unique items and 45 degree line indicates equal severity):



using the following code:

```
plot(b.1.std.new, b.2.std.new, pch = 5, col = "blue",xlab =
"Country1", ylab = "Country2", xlim = c(-3,3),ylim=c(-3,3))

abline(c(0,1))

text(b.1.std.new, b.2.std.new, colnames(XX.country1), cex =
0.6, pos=2)

points(b.1.std.new[!comm.mat], b.2.std.new[!comm.mat], col = 2,
pch = 5)
```

Comparable prevalence rates need to be calculated corresponding to thresholds adjusted to the same common metric.

Reporting thresholds on the metric of common items:

```
int1=mean.comm[1]

slop1=sd.comm[1]/sd(b1)

int2=mean.comm[2]

slop2=sd.comm[2]/sd(b2)

sthresh = c(-0.25, 1.81)

sthesh.new1 = (sthresh - int1)/slop1

sthesh.new2 = (sthresh - int2)/slop2
```

Calculating prevalence using the equated thresholds

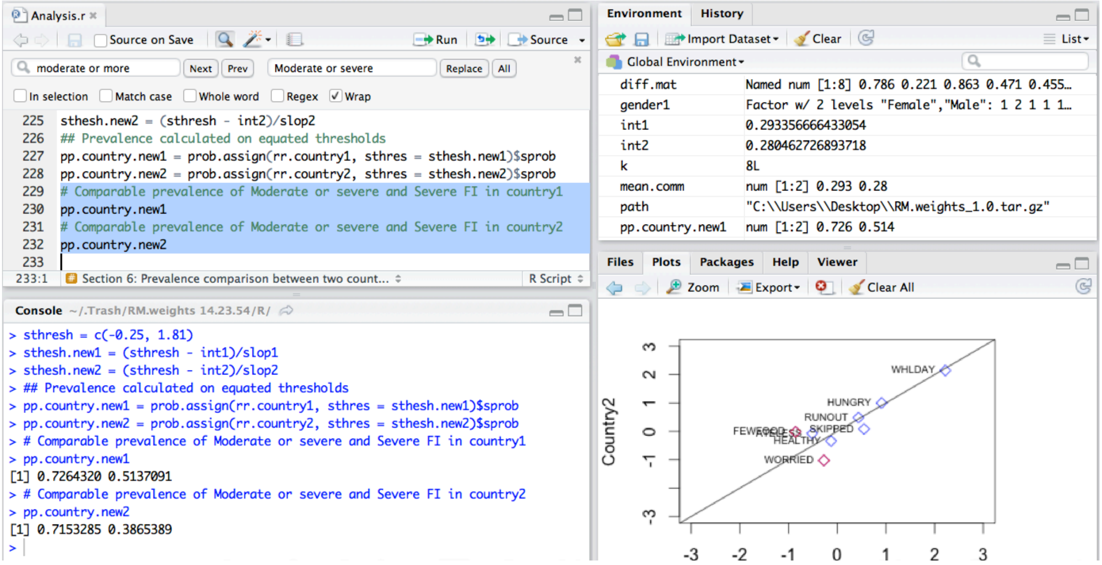
```
pp.country.new1 = prob.assign(rr.country1, sthres =
sthesh.new1)$sprob

pp.country.new2 = prob.assign(rr.country2, sthres =
sthesh.new2)$sprob
```

The comparable prevalence of Moderate or severe and Severe FI in country1 is

```
pp.country.new1
```

while the comparable prevalence of Moderate or severe and Severe FI in country2 is

```
pp.country.new2
```



moderate or more

Moderate or severe

```
sthesh.new2 = (sthresh - int2)/slop2
## Prevalence calculated on equated thresholds
pp.country.new1 = prob.assign(rr.country1, sthres = sthesh.new1)$sprob
pp.country.new2 = prob.assign(rr.country2, sthres = sthesh.new2)$sprob
# Comparable prevalence of Moderate or severe and Severe FI in country1
pp.country.new1
# Comparable prevalence of Moderate or severe and Severe FI in country2
pp.country.new2
```

```
> sthresh = c(-0.25, 1.81)
> sthesh.new1 = (sthresh - int1)/slop1
> sthesh.new2 = (sthresh - int2)/slop2
> ## Prevalence calculated on equated thresholds
> pp.country.new1 = prob.assign(rr.country1, sthres = sthesh.new1)$sprob
> pp.country.new2 = prob.assign(rr.country2, sthres = sthesh.new2)$sprob
> # Comparable prevalence of Moderate or severe and Severe FI in country1
> pp.country.new1
[1] 0.7264320 0.5137091
> # Comparable prevalence of Moderate or severe and Severe FI in country2
> pp.country.new2
[1] 0.7153285 0.3865389
>
```

```
diff.mat         Named num [1:8] 0.786 0.221 0.863 0.471 0.455…
gender1          Factor w/ 2 levels "Female","Male": 1 2 1 1 1…
int1             0.293356666433054
int2             0.280462726893718
k                8L
mean.comm        num [1:2] 0.293 0.28
path             "C:\\Users\\Desktop\\RM.weights_1.0.tar.gz"
pp.country.new1  num [1:2] 0.726 0.514
```

We can than conclude that severe prevalence rate FI in country 1 (about 51%) is higher than in country 2 (about 39%) because the two quantities are calculated on a common metric.

5.1. The “equating.fun” function

The equating process as described in the previous section is a particularly delicate step of the analysis, and it is preferable not to run it automatically. However, as a matter of comparison, the “equating.fun” function in the “RM.weights” package can be used in parallel with the manual process to perform the equating and calculate comparable prevalence rates. The calibration process run by this function is the following: at the first step, the most discrepant item (if exceeds the tolerance) is taken out from the set of common items. If all the remaining items align well with the global standard (or the country of reference), the function stops, otherwise, the second most discrepant item is taken out from the common set, and so on up to a maximum number of unique items.

Usage

```
equating.fun(rr1, st=NULL, tol = .35, spec.com1 = 1:8,
spec.com2=1:8, thres = c(-0.25,  1.83), maxuniq=3,
plot=F,iterative=T, excl.prior1, excl.prior2)
```

Arguments

`rr1`	The Rasch model fitted with the RM.w function for the country of interest.
`st`	Item severity parameter estimates for the reference country or the standard. The number of items for the standard can differ from the number of items of the country of interest.
`tol`	Tolerance expressed in the metric of the standard, above which items are considered unique (i.e. not contributing to the common metric between the two countries). Default is 0.35.
`spec.com1`	A priori numbered set of comparable items for the country of iterest to the standard. Items must be specified in the same order as they are used for the comparison.
`spec.com2`	A priori set of comparable items for the standard. The length of spec.com1 and spec.com2 should be the same.
`thres`	Thresholds (along the latent trait) corresponding to which the comparable prevalence rate of the phenomenon of interest is calculated.
`maxuniq`	A priori maximum number of unique items allowed.
`plot`	Logical argument. If TRUE, a pdf file with a plot of the equated item severities is produced in the working directory. Default is FALSE.
`iterative`	Logical argument. If TRUE, an iterative process to identify which item is unique is performed. Otherwise, unique items are specified in the arguments excl.prior1 and excl.prior2 and are taken as fixed. Default is TRUE.
`excl.prior1`	Numbered set of fixed unique items for the country of interest. It needs to be specified only if iterative = FALSE.
`excl.prior2`	Numbered set of fixed unique items for the standard. It needs to be specified only if iterative = FALSE.

Value

`scale`	Scale parameter to be applied to the metric of the country of interest to be considered equivalent to the standard.
`shift`	Shift parameter to be applied to the metric of the country of interest to be considered equivalent to the reference country. The standardized item parameters for the country of interest will be: b.country.st=shift+scale*b.country where b.country are estimated item severities for the country of interest.
`common`	Vector of logical, TRUE when the item in the country of interest is common.
`prevs`	Prevalence rates for the country of interest, calculated on the metric of the standard.
`prevs.rs`	Prevalence rates for the country of interest, calculated on the metric of the standard, calculated at each raw score threshold.
`cor.comm.items`	Correlation between common items.

An example of application can be found below (Section 7 of “Analysis.r”). The example entails the calculation of comparable prevalence rates for country 1 using as a standard the 2014-2015 global standard calculated by VoH.

```
# VoH 2014-2015 global standard
b.tot=c(-1.2590036, -0.8991436, -1.0876362,  0.4163556, -
0.2506451,  0.4466926,  0.8065710, 1.8268093)
# Equating of country 1 to the global standard
ee=equating.fun(rr.country1, st=b.tot, tol=0.5)
# Equated prevalence rates
ee$prevs*100
# Correlation between common items
ee$cor.comm.items
# Producing a plot of the item severities
ee=equating.fun(rr.country1, st=b.tot, tol=0.5, plot=T)
# The plot will be saved as a pdf called
"Equating_plot.pdf" file in the working directory
```