HOUSEHOLD FOOD INSECURITY INDICATOR

This chapter describes the prevalence of households with moderate and severe food insecurity using the Food Insecurity Experience Scale (FIES), a Feed the Future phase two indicator. This chapter has two sections; the first section describes the guidelines to construct the indicator, and the second section outlines the step-by-step procedures to calculate the indicator as well as the prevalence of moderate food insecurity and severe food insecurity.

15.1 Guidelines to construct the FIES indicator

This section provides the guidelines to construct the Feed the Future food insecurity indicator: the prevalence of moderate and severe food insecurity in the population, based on the FIES.

The prevalence of moderate and severe insecurity is calculated from the respondent's raw score—that is, the number of affirmative responses given to the eight FIES questions. The raw score is simply an integer with a value between 0 and 8, and hence there will always be up to 9 distinct values of respondent parameters, one for each possible raw score (0–8). Although the raw scores and associated respondent parameters depend only on the number of affirmative responses, certain response patterns are expected to conform to the Rasch model's assumptions—that is, when arranged in the order of increasing severity, responses start with "yes" and are followed by "no" (without alternating). This guide provides steps for statistical validation that must precede the calculation of prevalence estimates.

The Food and Agriculture Organization of the United Nations (FAO) produces two FIES indicators for global monitoring: moderate and severe food insecurity and severe food insecurity. In addition to (and the reason for) being a Feed the Future indicator, the moderate and severe food insecurity indicator is a monitoring indicator for Sustainable Development Goal 2 (Target 2.1). For further guidance, the analyst should review the guidance available at http://www.fao.org/in-action/voices-of-the-hungry/using-fies/en/.

There are 12 main steps in the step-by-step procedures (see Section 15.2 for details) that cover the key components in working with the FIES, including how to run the Rasch model and assess model fit, calibration of item severity scores to ensure a common scale between Rasch model outputs from multiple surveys, calculation of the prevalence of moderate and severe food insecurity at the aggregate and disaggregate levels, and suggestions for exporting data.

- Step 1: Set up the R environment and read the data into R
- Step 2: Prepare the data and determine how to handle missing data
- Step 3: Run the Rasch model and assess the model fit
- Step 4: Calibrate the item severity scores between Midline and Baseline Surveys[1]
- Step 5: Calibrate the FAO global scale to reference the Baseline Survey scale to obtain moderate and severe food insecurity thresholds[2]
- Step 6: Calculate the prevalence of moderate, severe, and moderate and severe food insecurity
- Step 7: Create binary variables that indicate moderate, severe, and moderate and severe food insecurity

[1] If not performing a baseline-midline comparative analysis, this step is not relevant.
[2] If not performing a baseline-midline comparative analysis, the Midline Survey is the reference scale.

- Step 8: Calculate the margins of error (MOE) and conduct tests of significance to assess change over time
- Step 9: Store the aggregate results in a common data frame
- Step 10: Calculate food insecurity prevalence estimates for disaggregate variables
- Step 11: Export dataset or results in CSV format
- Step 12: Export dataset or results in Stata format

The main statistical model used for FIES data assessment and scale construction is an Item Response Function known as the Rasch model. This model is used in Step 3. The Rasch model was developed in educational testing to score a latent trait of ability based on a student responding correctly to a series of questions. In applying the model to food insecurity, the scale is defined as a scale of "severity," or the degree of negative impact on the household's or individual's welfare due to the inability to freely access the food one needs. This is a probabilistic model that scores households based on the number 'yes' responses to a series of questions related to the experience of food insecurity. According to this model, the probability of a respondent affirming the j−th item is modeled as a logistic function of the distance between two parameters, one representing the item's severity (b_j) and one representing the respondent's severity (θ) (Cafiero, Viviani & Nord, 2017; Onori, Viviani, & Brutti, 2021):

$$P_j(\theta) = P\left(X_j = 1 \middle| \theta; b_j\right) = \frac{\exp(\theta - b_j)}{1 + \exp(\theta - b_j)}$$

In Steps 4 and 5, scale calibration is done to ensure that a common scale for the item and raw severity scores is used for different surveys. This is a necessary step to ensure that alike comparisons are being made between different contexts and is particularly important if working with Midline Survey data in comparison to Baseline Survey data. The item and raw severity scores of the Baseline Survey are typically used as the reference scale, and the Midline Survey is adjusted to fit the baseline scale because the baseline was done first.

When performing the calibration, the item severity scores for the two surveys are first compared to identify common items—that is items with similar severity scores for the two surveys. Then the adjusted midline item severity scores (z_i) are obtained by subtracting the mean of the midline common item severity scores ($\bar{x}$) from each midline item severity score (x_i) and dividing by the product of the population standard deviations (SD) of the common item severity scores for baseline and midline ($\sigma_y \times \sigma_{x)}$) plus the mean of the baseline common item severity scores ($\bar{y}$).

The mean of the midline common item severity scores ($\bar{x}$) is calculated as follows:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

Where:

x_i= the severity score of item i
N = the number of common items

The mean of the baseline common item severity scores ($\bar{y}$) is calculated as follows:

$$\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$$

Where:

y_i= the severity score of item i

The population SD of the midline common item severity scores (σ_x) is calculated as follows:

$$\sigma_x = \sqrt{\frac{(x_i - \bar{x}^2)}{N}}$$

The population SD of the baseline common item severity scores (σ_x) is calculated as follows:

$$\sigma_y = \sqrt{\frac{(y_i - \bar{y}^2)}{N}}$$

The adjusted midline item severity scores (z_i) are calculated as follows, using the values generated in the previous equations:

$$z_i = \frac{x_i - \bar{x}}{\bar{y} + (\sigma_y \times \sigma_{x)}}$$

The same calculation is used for the adjusted raw score severity scores (s_i), in which we substitute the individual raw score severity scores (r_i) for the item severity scores (x_i).

$$s_i = \frac{r_i - \bar{x}}{\bar{y} + (\sigma_y \times \sigma_x)}$$

The adjusted SD of the raw score severity scores (σ_s) is calculated by dividing each one by the product of the SD of the common item severity scores for baseline and midline:

$$\sigma_s = \frac{\sigma_r}{(\sigma_y \times \sigma_x)}$$

The same set of equations are used for the calibration of the global scale onto the local scale in Step 5.[3] The FAO global scale comprises the item severity scores obtained from a series of surveys done in multiple countries for the Gallup World Poll from 2014 to 2016. The fifth item (**ATELESS**) and eighth item (**WHLDAY**) of the global scale serve as the thresholds for moderate and severe food insecurity and must be mapped onto the local reference scale, whether baseline or midline. For this reason, the global scale is adjusted to fit onto the local reference scale, as was done with adjusting the midline to the baseline.

[3] The local scale refers to the scale used as the anchor for analysis. The baseline serves as the local scale if calibration between the baseline and midline for comparative analysis is done. If no baseline is used in analysis, then the midline serves as the local scale.

Obtaining the adjusted global scale item severity scores (z_i) is done by subtracting the mean of the global scale common item severity scores ($\bar{g}$) from each global scale item severity score (g_i) divided by the product of the population SD of the common item severity scores for the global scale onto the local scale ($\sigma_g \times \sigma_l$) plus the mean of the local common item severity scores ($\bar{l}$).

The mean of the common global scores (g_i) is calculated as follows:

$$\bar{g} = \frac{1}{N}\sum_{i=1}^{N} g_i$$

Where:

g_i= the severity score of item i

The mean of the common local scores (l_i) is calculated as follows:

$$\bar{l} = \frac{1}{N}\sum_{i=1}^{N} l_i$$

Where:

l_i= the severity score of item i

The SD of the common global scores (σ_g) is calculated as follows:

$$\sigma_g = \sqrt{\frac{(g_i - \bar{g}^2)}{N}}$$

The SD of the common local scores (σ_l) is calculated as follows:

$$\sigma_l = \sqrt{\frac{(l_i - \bar{l}^2)}{N}}$$

The adjusted global reference scale item severity scores (w_i) are calculated as follows, using the values generated in the previous equations:

$$w_i = \frac{g_i - \bar{g}}{\bar{l} + (\sigma_l \times \sigma_{g)}}$$

15.2 Step-by-step procedures to calculate the FIES indicator

This section describes the detailed step-by-step procedures to calculate the prevalence of moderate or severe food insecurity in the population, based on the FIES. Several suggestions in the steps are made in the context of working with Midline Survey data alongside Baseline Survey data or multiple surveys done sequentially in which one can serve as a reference point. If this context does not fit your analysis needs, then it is recommended either to skip particular sections or consult the materials provided on the FAO website in the link provided in Section 15.1.

Definitions

Numerator	Number of households with a probability of exceeding the moderate food insecurity severity level
Denominator	Total number of surveyed households
Unit of measure	Percentage
Level of data	Household
Sampling weight	Household
Disaggregation levels	Gendered household type* Residency (urban/rural) Wealth quintile Shock exposure severity
Treatment of missing data	Households with missing or refused values for all FIES items are excluded from the indicator calculations. Instructions are provided on handling missing values on the FIES items in Step 2d.
Survey variables used	**v301**, **v302**, **v303**, **v304**, **v305**, **v306**, **v307**, **v308**, **hhea**, **hhnum**, **hhsize_dj**, **wgt_hh**, **ahtype**, **strata**
Analytic variables used	**genhhtype_dj**, **awiquint**, **shock_sev**
Analytic variables created	**wt**, **WORRIED**, **HEALTHY**, **FEWFOOD**, **SKIPPED**, **ATELESS**, **RUNOUT**, **HUNGRY**, **WHLDAY**, **prob_mod**, **prod_mod_sev**, **prob_sev**, **fi_mod_sev_bin**, **fi_mod_bin**, **fi_sev_bin**

*Standard Feed the Future disaggregate

Calculations

FAO's Voices of the Hungry project provides a free analytical tool for FIES analysis using R software.

Following the FAO technical resources, this guide describes the steps required to compute the prevalence of moderate food insecurity, severe food insecurity, and moderate or severe food insecurity based on the FIES scale. It also describes how to compare FIES estimates over time for overall prevalence and disaggregates.

Step 1. Download R, install the required packages, set a working directory, and read the data into R.

Step 1a. Go to https://cran.r-project.org/ to download R. After it is downloaded, R can be used directly in the console or through a user-friendly compiler, RStudio, which needs to be downloaded separately.

Step 1b. Go to https://www.rstudio.com/products/rstudio/download/ to download RStudio, which is an integrated development environment for R. It includes a console, a syntax-highlighting editor that supports direct code execution, and tools for plotting, tracking history, debugging, and managing the workspace.

Step 1c. Install and upload the required packages.

Install the "**RM.weights**," "**survey**," "**haven**," and "**tidyverse**" packages if not already installed.

```
install.packages("RM.weights") # for Rasch modeling
install.packages("survey") # for analysis of complex survey
samples
```

```
install.packages("haven") # to read data stored in Stata, SPSS,
or other software
install.packages("tidyverse") # for data manipulation
```

After the packages are installed, upload the packages in the working library.

```
library(RM.weights)
library(survey)
library(haven)
library(tidyverse)
```

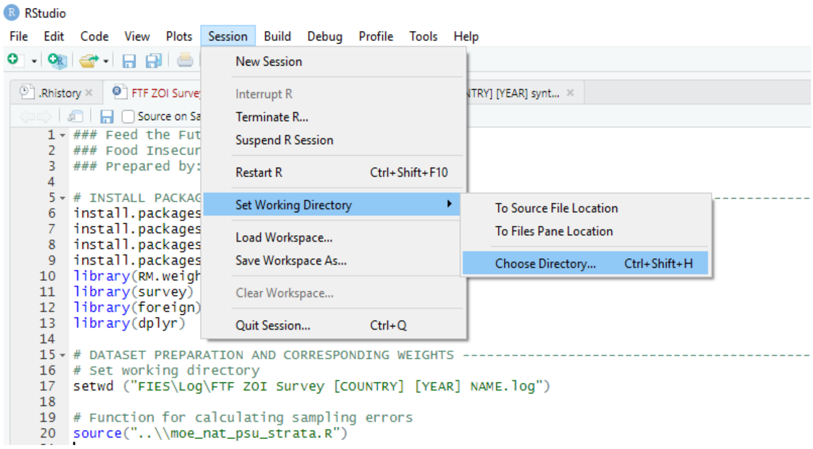
Note that you only need to install the packages above one time. However, each time before you use the FIES code, you need to upload the packages using the `library` function.

Step 1d. Specify the working directory in which the data are stored. This will serve as the root directory.

```
setwd("FIES\\FTF ZOI Survey [COUNTRY] [YEAR] NAME")
```

The working directory should be changed to reflect the directory path and folder names. Alternatively, go to RStudio console to set the working directory. Click on Session, and then Set Working Directory and Choose Directory (Figure 1).

Figure 1: Setting up a Working Directory in R Studio



```
### Feed the Fut
### Food Insecur
### Prepared by:

# INSTALL PACKAG
install.packages
install.packages
install.packages
install.packages
library(RM.weigh
library(survey)
library(foreign)
library(dplyr)

# DATASET PREPARATION AND CORRESPONDING WEIGHTS -----------------------------------------
# Set working directory
setwd ("FIES\Log\FTF ZOI Survey [COUNTRY] [YEAR] NAME.log")

# Function for calculating sampling errors
source("..\\moe_nat_psu_strata.R")
```

Step 1e. Load in script to calculate the MOE.

Read the source code of the **moe** function to calculate the CI and DEFF. Before running the code, ensure that the "`moe_nat_psu_strata_v2.R`" file is located in the working directory, and call the **moe** function into the code.[4]

```
source("moe_nat_psu_strata_v2.R")
```

[4] This script is available in the Midline Toolkit.

The **moe** function facilitates the calculations of the CI and DEFF using the survey package of R, which accounts for the complex survey design, such as clustering and unequal probabilities of selection, by specifying the primary sampling unit, strata, and sampling weights.

Step 1f. Create functions to compute the population-level SD and variance.

```
#create a new function to compute population standard deviation
var.p <- function(x) var(x) * (length(x)-1) / length(x)
sd.p <- function(x) sqrt(var.p(x))
```

The population SD is used in later steps to calibrate different scales created from the Rasch model and takes into account the entire population.

Step 1g. Use the **haven** package to read in the ZOI Survey household-level analytic data file with the Stata labels preserved.

```
ftfzoi_midline_import <- read_dta(“FTF ZOI Survey [COUNTRY]
[YEAR] household data analytic.dta”)
```

Step 2. Prepare the data.

Note: For simplicity, Steps 2 and 3 show only preparation and analysis with the Midline Survey data. If this analysis is done alongside Baseline Survey data for comparative analysis, be sure to follow these steps for the baseline data as well. Variable names in the baseline data may differ.

Step 2a. Create a data frame (**ftfzoi_midline**) in which the FIES survey variables are renamed to the corresponding items that will be used in the Rasch model using the **rename** function.

```
ftfzoi_midline <-
  ftfzoi_midline_import %>%
  rename(
    "WORRIED" = "v301",
    "HEALTHY" = "v302",
    "FEWFOOD" = "v303",
    "SKIPPED" = "v304",
    "ATELESS" = "v305",
    "RUNOUT"  = "v306",
    "HUNGRY"  = "v307",
    "WHLDAY"  = "v308")
```

Step 2b. Recode all variables in the **ftfzoi_midline** data frame created in Step 2a. Keep ‘YES’ responses coded as ‘1,’ and recode ‘NO’ responses as ‘0.’ ‘REFUSED’ responses (‘7’) will be set to missing (NA) and excluded from the analysis.

```
ml_hh_midline <- ml_hh_midline %>% mutate_at(
  dplyr::funs(case_when(
  x == 1 ~ 1,
  x == 2 ~ 0
 )
)
```

Step 2c. Store the complex survey design variables and disaggregate variables that will be used later for analysis. Remember to edit the variable names, values, and labels as needed.

```
# variables used for complex survey design - weights, household
# size, cluster ID, HHID, strata
ftfzoi_midline$wt <- ftfzoi_midline$wgt_hh
ftfzoi_midline$hhsize_dj <- ftfzoi_midline$hhsize_dj
ftfzoi_midline$hhnum <- ftfzoi_midline$hhnum
ftfzoi_midline$strata <- ftfzoi_midline$strata

# Disaggregates – gendered HH type, residency, shock severity,
   wealth index
ftfzoi_midline$ahtype <- factor(ftfzoi_midline$ahtype,
   levels=c(1,2), labels=c("urban","rural"))
ftfzoi_midline$genhhtype_dj <-
   factor(ftfzoi_midline$genhhtype_dj,
levels=c(1,2,3,4), labels=c("Male and Female adults","Female
   adults only","Male adults only", "Children only"))
ftfzoi_midline$shock_sev <- factor(ftfzoi_midline$shock_sev,
   levels = c(1,2,3,4), labels = c(“None”, “Low”, “Medium”,
   “High”))
ftfzoi_midline$awiquint <- factor(ftfzoi_midline$awiquint, levels
   = c(1,2,3,4,5), labels = c("Poorest", "Second”, "Middle",
   "Fourth", "Wealthiest"))

# select variables
ftfzoi_midline  <- ftfzoi_midline %>%
select(wt, mem, hhea, hhnum, strata, ahtype, genhhtype_dj,
   shock_sev, awiquint, WORRIED:WHLDAY)
```

Step 2d. Treat missing values on the FIES items.

There are two types of missing values (NA) that may be encountered: (1) all FIES items are missing values, and (2) individual FIES items are missing values but other FIES items have values.

(1) If values are missing for all FIES items, remove the household from the analysis entirely.

```
# remove HHs where all FIES items are missing
ftfzoi_midline <- ftfzoi_midline %>%
   filter_at(dplyr::vars(WORRIED:WHLDAY),
dplyr::all_vars(!is.na(.)))
```

(2) More caution is required handling individual item non-response. Imputation of values is recommended only in cases in which a pattern is clear. For example, if a household has values of "NO" (0) for all items, except for **ATELESS**, which has a missing value, **ATELESS** can be safely imputed as 0.

To explore the missing values in the data, use the following code:

```
      ftfzoi_midline %>%
      filter_at(dplyr::vars(WORRIED:WHLDAY),
dplyr::any_vars(is.na(.)))
```

To impute a value if **ATELESS** is missing a value and all other FIES items have a value of "NO" (0), the following code may be used:

```
      ftfzoi_midline <- ftfzoi_midline %>%
      dplyr::mutate(ATELESS = case_when(WORRIED == 0 & HEALTHY == 0 &
            FEWFOOD == 0 & SKIPPED == 0 & RUNOUT  == 0 & HUNGRY  == 0 &
            WHLDAY  == 0 & is.na(ATELESS) ~ 0, TRUE ~
            as.numeric(ATELESS)))
```

This code identifies households that follow the exact item value pattern specified and assigns 0 to **ATELESS**. For other households that do not have this item value pattern, it keeps **ATELESS** at the original value. Similar code can be used for other items with missing values in cases in which all other items are all "NO" responses.

Another example of acceptable imputation of individual item response is when a household has values of "YES" (1) for less and more severe items surrounding the item with a missing value; this item can be safely imputed as 1.

Assigning values in less absolute cases is not recommended. However, if analysts find themselves working with a high number of missing values, they may want to. When assigning values, it is important to pay close attention to the hierarchy of the items, and where evident patterns emerge, such as a missing value between two "YES" values for the next most severe and next least severe items. To assess item severity patterns without running the Rasch model, calculate the proportion of households with a value of "YES" for each FIES item and order the items by increasing proportion. These proportions will largely follow the item severity scores generated in Step 3 and can provide a sufficient idea of the severity ranking of the FIES items and can be used to justify imputation of individual item missing values in similar cases.

Because the FIES items were converted to binary variables in Step 2b, the **colMeans** function can be used to calculate column proportions (i.e., the proportion of households with a value of "YES" for each FIES item):

```
      ftfzoi_midline %>%
            select(WORRIED:WHLDAY) %>%
            colMeans()
```

The output will look like this:

```
WORRIED    HEALTHY    FEWFOOD    SKIPPED    ATELESS
0.37320770 0.22408849 0.18189267 0.04547317 0.10651372
RUNOUT        HUNGRY      WHLDAY
0.08152397     0.05489553  0.02458009
```

The percentages will usually decrease with each subsequent item, although each survey has its own exceptions due to differences in linguistic interpretation and behavior patterns within the surveyed population. In the example above, SKIPPED has the second lowest percentage of "YES" responses among households.

Step 2e. Create a data frame (`FIES_midline`) that includes only the binary FIES items after all imputation has been completed using the `select` function:

```
FIES_midline <- ftfzoi_midline %>%
  select(WORRIED:WHLDAY)
```

Step 3. Run the Rasch model to obtain severity scores used for calculating food insecurity prevalence and diagnostic test results to assess model fit.

The Rasch model (1) assesses the suitability of the eight FIES items for scale construction, (2) computes parameter estimates and assessment statistics for each item, (3) generates a scale from the items, and (4) assesses the location of a household along the continuum of the scale that uniquely reflects the food insecurity situation of that household. The model is critical for Step 4, in which item severity scores and the FAO global scale are used to produce the moderate and severe food insecurity prevalence rates.

Step 3a. Run the Rasch model on the selected FIES items and households after data preparation and handling missing data, using the data frame created in Step 2e (`FIES_midline`). Use the **`RM.w`** function of the **`RMweights`** package to run the model.

```
FIES_midline_rasch <- RM.w(FIES_midline, write.file = TRUE,
     country = "FTF ZOI SURVEY COUNTRY")
FIES_midline_RS = rowSums(FIES_midline)
```

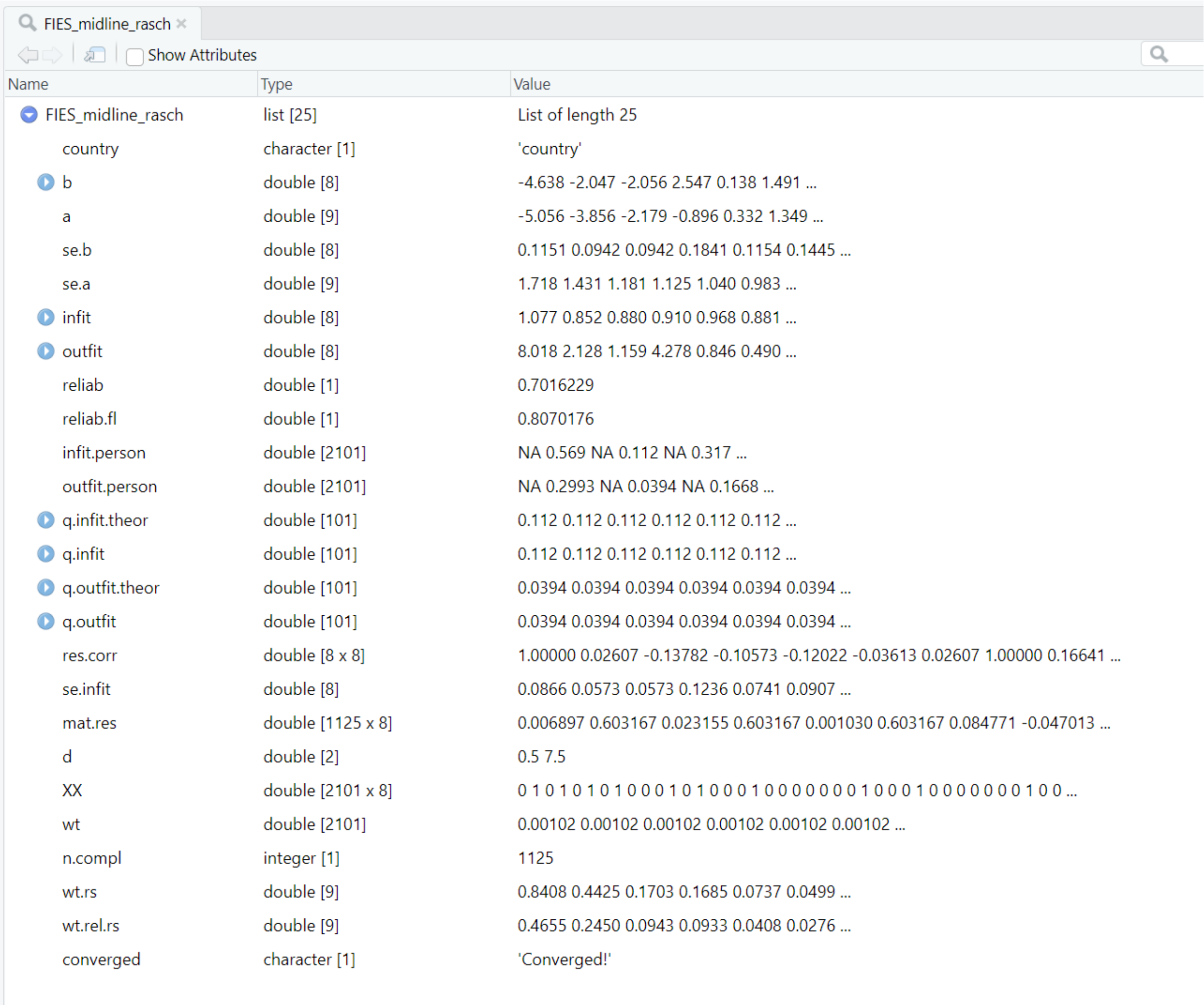
The **`RM.w`** function has a weights argument that is optional. Given that Rasch models are invariant to population-based weights, they are not used for ZOI Survey analyses.[5] The function automatically exports the Rasch output to a CSV file saved in the working directory. These results can be used to test the quality of data collected.

The Rasch model will also be saved in a new object in the R environment (`FIES_midline_rasch`) that contains several important outputs (see Figure 2), including the raw score severity scores (`FIES_midline_rasch$a`), item severity scores (`FIES_midline_rasch$b`), the correlation of residuals among items (`FIES_midline_rasch$res.corr`), a residual matrix (`FIES_midline_rasch$mat.res)`, the distribution of valid responses, missing data by item, and a detailed output that shows the observed response proportion and the predicted response

[5] If required to include, add the **`.wt = x`** argument to the function above.

proportion for each raw score and items. In addition, the output includes the results of several important diagnostic tests that are useful to assess the measurement reliability and quality of data.

Figure 2: Example Rasch Module Output



Name	Type	Value
FIES_midline_rasch	list [25]	List of length 25
country	character [1]	'country'
b	double [8]	-4.638 -2.047 -2.056 2.547 0.138 1.491 ...
a	double [9]	-5.056 -3.856 -2.179 -0.896 0.332 1.349 ...
se.b	double [8]	0.1151 0.0942 0.0942 0.1841 0.1154 0.1445 ...
se.a	double [9]	1.718 1.431 1.181 1.125 1.040 0.983 ...
infit	double [8]	1.077 0.852 0.880 0.910 0.968 0.881 ...
outfit	double [8]	8.018 2.128 1.159 4.278 0.846 0.490 ...
reliab	double [1]	0.7016229
reliab.fl	double [1]	0.8070176
infit.person	double [2101]	NA 0.569 NA 0.112 NA 0.317 ...
outfit.person	double [2101]	NA 0.2993 NA 0.0394 NA 0.1668 ...
q.infit.theor	double [101]	0.112 0.112 0.112 0.112 0.112 0.112 ...
q.infit	double [101]	0.112 0.112 0.112 0.112 0.112 0.112 ...
q.outfit.theor	double [101]	0.0394 0.0394 0.0394 0.0394 0.0394 0.0394 ...
q.outfit	double [101]	0.0394 0.0394 0.0394 0.0394 0.0394 0.0394 ...
res.corr	double [8 x 8]	1.00000 0.02607 -0.13782 -0.10573 -0.12022 -0.03613 0.02607 1.00000 0.16641 ...
se.infit	double [8]	0.0866 0.0573 0.0573 0.1236 0.0741 0.0907 ...
mat.res	double [1125 x 8]	0.006897 0.603167 0.023155 0.603167 0.001030 0.603167 0.084771 -0.047013 ...
d	double [2]	0.5 7.5
XX	double [2101 x 8]	0 1 0 1 0 1 0 1 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 ...
wt	double [2101]	0.00102 0.00102 0.00102 0.00102 0.00102 0.00102 ...
n.compl	integer [1]	1125
wt.rs	double [9]	0.8408 0.4425 0.1703 0.1685 0.0737 0.0499 ...
wt.rel.rs	double [9]	0.4655 0.2450 0.0943 0.0933 0.0408 0.0276 ...
converged	character [1]	'Converged!'

Step 3b. Assess fit with diagnostic results.

There are four main diagnostics that must be examined thoroughly to assess model fit: infit, outfit, Rasch reliability, and principal component analysis (PCA) on the residual matrix. This step provides a brief description and guidance along with code examples on how to assess each test result. The information provided for each test is not extensive, and analysts may wish to consult other materials given the complexities of working with Rasch models. More thorough background can be found in the following article by FAO outlining statistical validation of the FIES for the Sustainable Development Goals: https://elearning.fao.org/pluginfile.php/491591/mod_scorm/content/5/story_content/external_files/SDG2.1.2_lesson3.pdf

Infit (and corresponding standard errors):

Description: The infit statistic is based on the chi-square statistic with each observation weighted by the overall model variance. As a result, infit statistics are more sensitive to unexpected patterns of observations by households on items close to their cutoff.

Fit: If the infit statistic is 1.0 for each item, all items discriminate equally—that is, items are strongly or consistently related to a food insecurity condition—which is one of the main assumptions of the Rasch model. An adequate fit to the Rasch model is indicated by infit and outfit statistics of 0.7-1.3 for each item. Items with infit values of less than 0.7 or greater than 1.3 should not be used for scoring.

Guidance: If the item infit is less than 0.7 or greater than 1.3 for any single item, a useful first step is assessing household-level (in R called **`person.infit`**) statistics to remove households with very low or very high infit patterns and unusual patterns on the items that have high infit scores, or where values do not follow the hierarchical logic of the item severity. If item infit scores remain low or high, consider removing the item from the model and rerunning the Rasch model without it.

Every time a household (row) or item (column) is removed, the Rasch model should be re-run. If there is an item infit value less than 0.7 or greater than 1.3, the item should be dropped from the model and re-run. If there is more than one item with an infit value of greater than 1.3 or less than 0.7, the item with the highest/lowest value should be removed and the model should be re-run. Please note that an analyst should not remove more than three items from the Rasch model.

R code:

```
round(FIES_midline_rasch$infit) # to see infit scores for each
     item/question

round(FIES_midline_rasch$infit.person) # to see individual infit
     scores
```

Outfit (and corresponding standard errors):

Description: Outfit is an outlier-sensitive fit statistic, based on the conventional chi-square statistic. Compared to infit, outfit is more sensitive to unexpected observations for questions that are very easy or very hard to answer (i.e., a "NO" response to item 1 [WORRIED] and "YES" to all other items).

Fit: An item outfit of greater than 2 is considered high and may be useful for identifying where unusual patterns are located.

Guidance: Unlike item infit, a high item outfit score is typically not a sufficient reason to remove an item. Nevertheless, item infit and outfit should be examined together. As with infit, assessing household-level outfit statistics (in R called **`person.outfit`**) and potentially removing problematic response patterns can help improve measurement quality. If time permits, examining household-level outfit scores by enumerator can determine if any patterns emerge based on who collected the data.

R code:

```
round(FIES_rasch$outfit, digits = 2)

round(FIES_midline_rasch$outfit.person, digits = 2) # to see
     individual outfit scores
```

Rasch reliability (and SD):

Description: Rasch reliability is the proportion of total variance in the ZOI population that is explained by the Rasch model. The flat Rasch reliability statistic can be thought of similarly to how R-squared is used to assess the fit of a linear regression model. The flat Rasch reliability statistic is used because it assumes that each raw score has an equal number of cases and thus provides a comparable measure of model fit, where as the standard Rasch reliability statistic can be influenced by the total number of cases across the raw scores because it is weighted by the frequency of each raw score.

Fit: Like R-squared, the closer that the reliability statistic is to 1, the better the model fits the data. As a rule of thumb, the reliability statistic should be greater than or equal to 0.7 and the SD should be greater than 1.5.

Guidance: Usually, a low reliability score occurs because the sample size is too small to establish a reproducible item severity hierarchy. If reliability is lower than the suggested threshold (0.7), a first step is to examine missing (NA) values because they can adversely impact precision and reliability. If one individual item has 10 percent or more missing values, the item can be dropped entirely. If the threshold still is not met, it is possible that removing household observations with missing values on multiple FIES items can improve the reliability. If additional improvements are needed, removing households with high household-level outfit values (i.e., respondents who respond "YES" to items out of line with the severity of the items) can strengthen a hierarchical pattern among the items and increase the reliability score.

R code:

```
round(FIES_midline_rasch$reliab.fl, digits = 2) # reliability
stat

round(sd.p(FIES_midline_rasch$b), digits = 2) # calculate pop.
Standard deviation of item scores.
```

PCA on residual matrix:

Description: Examining residual variance using PCA provides information on the structure of the variance not explained by the model.

Fit: The residual variance should not be explained by only one or two principal components. Ideally, there should be a gradual drop-off in the residual variance. The scree plot should not be in an 'L' or hockey stick shape; this would suggest that the model is capturing factors that are not explained by the FIES items.

Guidance: If the PCA results show a gradual drop-off in the residual variance, analysts can be confident in their results. If the PCA results show a sharp drop-off in residual variance after one or two principal components, the analyst cannot do anything to make the data "better," but the results should be documented as part of the data quality assessment process.

R code:

```
plot(prcomp(FIES_midline_rasch$mat.res)$sd^2)
```

Final considerations:

- Although it is not one of the four main diagnostics included in this Guide, reviewing the residual correlation between items can be useful to see if the items successfully capture different aspects of food insecurity. A residual correlation between a pair of items is considered high if it is greater than 0.3. If two items have high residual correlation, it may be necessary to drop one of them and re-run the model.
- Any item or household removed from the Rasch model should be re-run one by one. For example, if there are two items with an infit scores greater than 1.3, the item with the highest score should be removed and the other item with a score greater than 1.3 left in. This is because removing the highest item can lead to a changes in the scores of other items. If the other items are still greater than 1.3, then it is suggested that the other item with a score greater than 1.3 be dropped as well.
- A minimum of five items are required to proceed with analysis; therefore, do not remove more than three items.
- Removal of items or households can be done by calling the column or row number in brackets with the data frame. In the brackets, rows are placed first (before the comma) and columns second (after the comma). Place a minus sign in front of the column or row to be removed.

```
# item (column) removal - WORRIED
FIES_midline <- FIES_midline[, -1]
FIES_midline_rasch <- RM.w(as.matrix(FIES_midline),write.file =
      TRUE, country = "FTF ZOI SURVEY COUNTRY")
# row (household) removal - HH ar row number 30
FIES_midline <- FIES_midline[-30, ]
FIES_midline_rasch <- RM.w(as.matrix(FIES_midline), write.file =
      TRUE, country = "FTF ZOI SURVEY COUNTRY")
```

Step 4. Use equating processes to adjust the midline data so that they are on the same local scale as the baseline data. The local scale will be the baseline because it serves as the anchor for the analysis. Creating a common scale is crucial to ensuring that prevalence estimates are accurate. A meteorologist would not use two different temperature measurement scales to compare the weather of two different regions. If analyzing only midline data without comparing to baseline data, skip to Step 5.

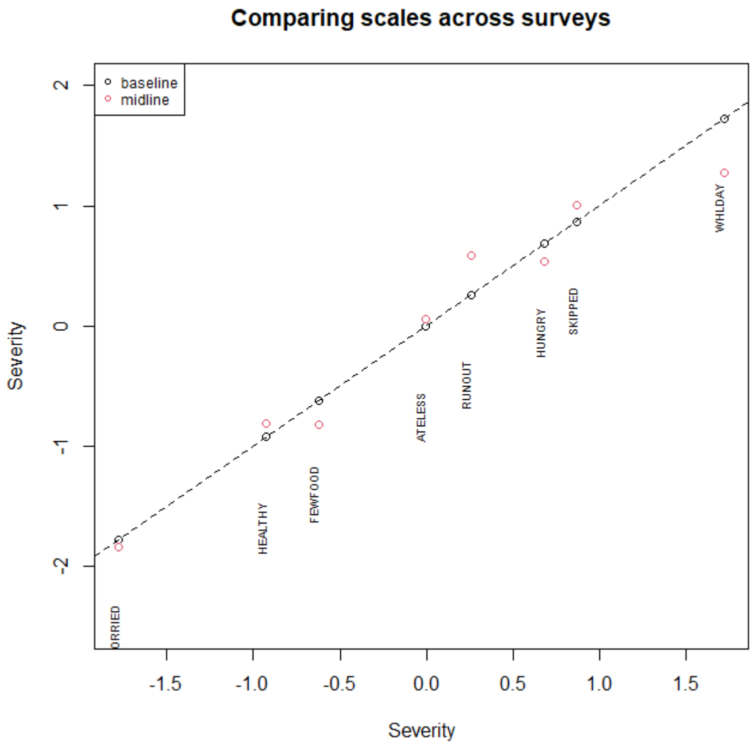
Before continuing with this step, be sure to create a baseline Rasch model output stored in a list (`FIES_baseline_rasch`), following the same steps used to create the midline Rasch object (`FIES_midline_rasch`).

Step 4a. Create a graph to compare baseline and midline item severity scores divided by the item SD.

```
{
  x   <- FIES_baseline_rasch$b/sd.p(FIES_baseline_rasch$b)
  y_bl <- FIES_baseline_rasch$b/sd.p(FIES_baseline_rasch$b)
  y_ml  <- FIES_midline_rasch$b/sd.p(FIES_midline_rasch$b)
  plot(x,y_bl, col = 1, ylim = c(-
     2.5,2),ylab="Severity",xlab="Severity")
  points(x, y_ml, col = 2)
  text(x+0.04,x-0.55,colnames(FIES),cex=0.6,pos=2,srt=90)
  abline(0,1,lty = 2)
  legend("topleft",c("baseline","midline"),pch = 1, col = c(1,2),
     cex = 0.75)
  title(main = "Comparing scales across surveys")

}
```

This code produces a graph similar to this:



Look at the difference between midline and baseline item severity scores for each item.

```
abs(round(FIES_baseline_rasch$b/sd.p(FIES_baseline_rasch$b) -
     FIES_midline_rasch$b/sd.p(FIES_midline_rasch$b), 2))
```

Example output:

```
WORRIED HEALTHY FEWFOOD SKIPPED ATELESS  RUNOUT  HUNGRY  WHLDAY
   0.06    0.11    0.19    0.14    0.06    0.36    0.15    0.44
```

To equate the baseline and midline data, identify the items that will be included in the common scale. To determine common items, a threshold of 0.35 is frequently used. That is, items with a midline-baseline item severity score difference of 0.35 or less will be considered common items and included in the common scale. A minimum of five common items are required. In the example above, item 6 (**RUNOUT**) and item 8 (**WHLDAY**) have item severity score differences greater than 0.35, and in the graph, the midline values (in red) are farther away from the dotted line, which runs through the baseline items. Therefore, items 6 and 8 will be excluded from the common scale.

Step 4b. Create vectors with common items to be used to calculate the mean and SD of the common item severity scores for both baseline and midline.

```
# identify items in common – one where scores are similar
not_common  <-  c(“WHLDAY”,”RUNOUT”)
common  <- setdiff(colnames(FIES_midline),not_common)
# mean of common item scores
m.bl <- mean(res_bl$b[common])
m.ml  <- mean(res_ml$b[common])
# sd of common item scores
s.bl  <- sd.p(res_bl$b[common])
s.ml  <- sd.p(res_ml$b[common])
```

Step 4c. Adjust the midline item severity scores (**FIES_midline_rasch$b**) and raw score severity scores (**FIES_midline_rasch$a**) so they are on the same scale as the baseline scores. To obtain the adjusted midline scores, the common item severity scores of the midline are subtracted by the mean of the common item severity scores. This is divided by the product of the population SD of the common item severity scores for the baseline and midline plus the mean of the baseline common item severity scores. This calibration process is outlined in greater detail in Section 15.1.

```
# adjust midline scale to the baseline metric
# assign Midline rasch object to Adjusted Midline rasch object
FIES_midline_rasch_adj  <- FIES_midline_rasch
# adjusted mean of item severity scores
FIES_midline_rasch_adj$b  <- (FIES_midline_rasch$b –
     m.ml)/s.ml*s.bl + m.bl
# adjusted mean of raw score severity scores
FIES_midline_rasch_adj$a  <- (FIES_midline_rasch$a –
     m.ml)/s.ml*s.bl + m.bl
# adjusted standard deviation of raw scores
FIES_midline_rasch_adj$se.a  <-
     FIES_midline_rasch_adj$se.a/s.ml*s.bl
```

Step 4d. Create a matrix with the weighted distributions of baseline and midline raw scores to use for the prevalence calculations.

```
RS_table  <- t(cbind(
  “Baseline” = aggregate(ftfzoi_baseline$wt,
     list(FIES_baseline_RS), FUN=sum, na.rm=TRUE)$x
     /sum(ftfzoi_baseline$wt [!is.na(FIES_baseline_RS)]),
  “Midline” = aggregate(ftfzoi_midline$wt, list(FIES_midline_RS),
     FUN=sum, na.rm=TRUE)$x /sum(ftfzoi_midline$wt
     [!is.na(FIES_midline_RS)])
))
```

Step 5. Map the global scale onto the local baseline scale to obtain thresholds for moderate and severe food insecurity, ensuring that measurement is consistent across datasets.

To calculate moderate and severe food insecurity as defined by the international standards of FAO, households must be assigned to a level of food insecurity as defined by set thresholds on the item severity scale. FAO has created a global standard of item parameter values based on survey results from more than 140 countries covered by the Gallup World Poll in 2014-2016, whereby the threshold for moderate food insecurity is set at the severity of the fifth item (**ATELESS**) and the threshold for severe food insecurity is set at the severity of the eighth item (**WHLDAY**) (FAO, 2015).

The calibration process is similar to the process used to equate baseline and midline item severity scores.

Note: In this step, the baseline scale is the local scale used. As such, the midline was adjusted to the baseline scale in Step 4. If a baseline is not included in the analysis, the midline can be used as the local scale.

Step 5a. Define the local (baseline) and the global scales.[6]

```
loc_st <- FIES_baseline_rasch$b
glob_st  <- c(“WORRIED”= -1.2230564, “HEALTHY”= -0.847121,
     “FEWFOODS”= -1.1056616, “SKIPPED” = 0.3509848, “ATELESS” =
     -0.3117999, “RUNOUT” = 0.5065051, “HUNGRY” = 0.7546138,
     “WHLDAY” = 1.8755353)
```

Step 5b. Obtain the absolute value of the difference of the standardized versions of the local and global scales.

```
# standardized version of both scales – item severity score
     divided by the population SD
```

[6] The values of the global scale and additional details on their construction can be found at: https://doi.org/10.1016/j.measurement.2017.10.065

```
abs(round(loc_st/sd.p(loc_st) – glob_st/sd.p(glob_st), 2))
```

Example output:

```
WORRIED HEALTHY FEWFOOD SKIPPED ATELESS  RUNOUT  HUNGRY  WHLDAY
  0.55    0.08    0.28    0.31    0.52    0.25     0.07   0.34
```

To calibrate the local and global scales, identify the items that will be included in the common scale. To determine common items, use a threshold of 0.35 as was done for equating the baseline and midline data—that is, items with a local-global item severity score difference of 0.35 or less will be considered common items and included in the common scale. In the example above, items 1 (`WORRIED`), and 5 (`ATELESS`) have item severity score differences greater than 0.35 and would, therefore, be excluded from the common scale.

Step 5c. Calculate the mean and SD of the common item severity scores for both the global and local scales.

```
# produce mean and sd for each scale among common items used in
    equating
# columns 2,3,4,6,7,8
glob_st.m <- mean(glob_st[c(2:4, 6:8)])
glob_st.s <- sd.p(glob_st[c(2:4, 6:8)])
m.bl  <- mean(loc_st[c(2:3, 6:8)])
s.bl  <- sd.p(loc_st[c(2:3, 6:8)])
```

Step 5d. Adjust the global scale to fit the local scale. The thresholds are defined as items 5 and 8 of the adjusted scale based on FAO's thresholds for moderate and severe food insecurity.

```
# mapping the thresholds from the global scale onto the local
     (baseline) scale
glob_st_adj  <- (glob_st – glob_st.m)/(glob_st.s * s.bl  + m.bl)
newthres  <- glob_st_adj[c(5,8)]
```

Step 6. Calculate the prevalence of moderate and severe food insecurity, severe food insecurity, and moderate food insecurity.

For this, the `pnorm` function uses the normal distribution to assess the probability of a household being over the moderate and severe thresholds given the threshold level, mean raw score severity score, and SD of the raw score severity scores.

Step 6a. Assign a probability to each household that it is beyond the moderate food insecurity threshold and then calculate the prevalence of moderate and severe food insecurity using matrix multiplication of each household's score probabilities against the frequency of raw scores in the sample:

```
# moderate+severe FI
# midline
```

```
glo_probs_ml_mod_sev  <- 1-pnorm(newthres[1], mean =
     FIES_midline_rasch_adj$a, sd = FIES_midline_rasch_adj$se.a)

glo_prev_ml_adj_mod_sev  <- glo_probs_ml_mod_sev %*%
     FIES_midline_RS[2,]

#baseline (if needed)
glo_probs_bl_mod_sev  <- 1-pnorm(newthres[1], mean =
     FIES_baseline_rasch$a, sd = FIES_baseline_rasch$se.a)

glo_prev_bl_mod_sev  <- glo_probs_bl_mod_sev %*%
     FIES_baseline_RS[1,]
```

Step 6b. Follow a similar process to calculate the prevalence of severe food insecurity.

```
# severe FI
# midline
glo_probs_ml_sev  <- 1-pnorm(newthres[2], mean =
     FIES_midline_rasch_adj$a, sd = FIES_midline_rasch_adj$se.a)

glo_prev_ml_adj_sev  <- glo_probs_ml_sev %*% FIES_midline_RS[2,]

#baseline (if needed)
glo_probs_bl_sev  <- 1-pnorm(newthres[2], mean =
     FIES_baseline_rasch$a, sd = FIES_baseline_rasch$se.a)

glo_prev_bl_sev  <- glo_probs_bl_sev %*% FIES_baseline_RS[1,]
```

Step 6c. Calculate the prevalence of moderate food insecurity by obtaining the difference between the moderate and severe food insecurity prevalence and severe food insecurity prevalence.

```
# create moderate only  <- mod+sev – sev
# prevalence
glo_prev_bl_mod <- glo_prev_bl_mod_sev – glo_prev_bl_sev
glo_prev_ml_adj_mod <- glo_prev_ml_adj_mod_sev –
     glo_prev_ml_adj_sev

# probability of moderate food insecurity at each raw score
glo_probs_bl_mod <- glo_probs_bl_mod_sev - glo_probs_bl_sev
glo_probs_ml_mod <- glo_probs_ml_mod_sev - glo_probs_ml_sev
```

Step 6d. Add the prevalence results to a common object and assign probabilities to each household in the original data frame read in Step 1g.

```
# put into common object
```

```
glo_prev_bl <- c(glo_prev_bl_mod_sev, glo_prev_bl_mod,
     glo_prev_bl_sev)
glo_prev_ml <- c(glo_prev_ml_adj_mod_sev, glo_prev_ml_adj_mod,
     glo_prev_ml_adj_sev)

#Attaching probabilities to each case/HH
ftfzoi_baseline$prob_mod_sev  <- NULL
ftfzoi_midline$prob_mod_sev  <- NULL

ftfzoi_baseline$prob_mod  <- NULL
ftfzoi_midline$prob_mod  <- NULL

ftfzoi_baseline$prob_sev  <- NULL
ftfzoi_midline$prob_sev  <- NULL

for (rs in 0:8) {
  ftfzoi_baseline$prob_mod[ftfzoi_baseline$RS == rs] =
     glo_probs_bl_mod[rs+1]
  ftfzoi_midline$prob_mod[ftfzoi_midline$RS == rs] =
     glo_probs_ml_mod[rs+1]
  ftfzoi_baseline$prob_mod_sev[ftfzoi_baseline$RS == rs] =
     glo_probs_bl_mod_sev[rs+1]
  ftfzoi_midline$prob_mod_sev[ftfzoi_midline$RS == rs] =
     glo_probs_ml_mod_sev[rs+1]
  ftfzoi_baseline$prob_sev[ftfzoi_baseline$RS == rs] =
     glo_probs_bl_sev[rs+1]
  ftfzoi_midline$prob_sev[ftfzoi_midline$RS == rs] =
     glo_probs_ml_sev[rs+1]
}
table(ftfzoi_baseline$prob_mod, RS, useNA = “ifany”)
table(ftfzoi_baseline$prob_mod_sev, RS, useNA = “ifany”)
table(ftfzoi_baseline$prob_sev, RS, useNA = “ifany”)
```

Step 7. Now that each household has a probability assigned for each level of severity, an optional step is to assign binary variables to the different food insecurity levels. This can be helpful when trying to use food insecurity as a disaggregate in the analysis of other variables. To do this, use an **`ifelse`** statement that assigns 1 if the unit is .5 or over and 0 if the unit is below.

```
Ftfzoi_midline$fi_mod_sev_bin <-
     ifelse(ftfzoi_midline$prob_mod_sev <= .5, 1, 0)
ftfzoi_midline$fi_mod_bin <- ifelse(ftfzoi_midline$prob_mod <=
     .5, 1, 0)
ftfzoi_midline$fi_sev_bin <- ifelse(ftfzoi_midline$prob_sev <=
     .5, 1, 0)
```

```
ftfzoi_baseline$fi_mod_sev_bin <-
     ifelse(ftfzoi_baseline$prob_mod_sev <= .5, 1, 0)
ftfzoi_baseline$fi_mod_bin <- ifelse(ftfzoi_baseline$prob_mod <=
     .5, 1, 0)
ftfzoi_baseline$fi_sev_bin <- ifelse(ftfzoi_baseline$prob_sev <=
     .5, 1, 0)
```

Step 8. Calculate the baseline and midline MOE and use them to detect statistically significant changes in the food insecurity estimates over time.

The MOE is calculated using the **moe** function, which was read into R in Step 1f. When calling the function, specify the confidence level (the default is 90 percent; in the example that follows, 95 percent is specified), and define the primary sampling unit and strata variables.

```
Modsev_bl_moe_95 <-
     moe(ftfzoi_baseline$prob_mod_sev,ftfzoi_baseline$RS,ftfzoi_
     baseline$wt * 10^6, conf.level = .95, psu =
     ftfzoi_baseline$psu, strata = ftfzoi_baseline$strata)$moe *
     100

modsev_ml_moe_95 <-
     moe(ftfzoi_midline$prob_mod_sev,ftfzoi_midline$RS,ftfzoi_mi
     dline$wt * 10^6, conf.level = .95, psu =
     ftfzoi_midline$psu, strata = ftfzoi_midline$strata)$moe *
     100
```

To assess statistically significant changes in food insecurity prevalence over time, from baseline to midline, subtract the baseline estimate from the midline estimate and compare the difference to the midline MOE. If the difference is greater than the midline MOE, the change is considered statistically significant at the p<.05 level.

To assess statistically significant changes in food insecurity prevalence over time, from baseline to midline, subtract the baseline estimate from the midline estimate and compare the difference to the midline MOE. If the difference is greater than the midline MOE, the change is considered statistically significant at the p<.05 level.

```
Fies_diff <- glo_prev_bl[1] - glo_prev_ml[1] # calculate
     difference
ifelse(fies_diff > modsev_ml_moe_95, TRUE, FALSE) # output will
     indicate whether statement is TRUE or FALSE
```

The MOE for each food insecurity level and survey can be calculated and the value can be stored into an object as follows.

```
# Baseline
modsev_moe_bl_95 <-
     moe(ftfzoi_baseline$prob_mod_sev,ftfzoi_baseline$RS,ftfzoi_
     baseline$wt * 10^6, conf.level = .95,
                       psu = ftfzoi_baseline$psu, strata =
     ftfzoi_baseline$strata)$moe * 100
mod_moe_bl_95 <-
     moe(ftfzoi_baseline$prob_mod,ftfzoi_baseline$RS,ftfzoi_base
     line$wt * 10^6, conf.level = .95,
                    psu = ftfzoi_baseline$psu, strata =
     ftfzoi_baseline$strata)$moe * 100
sev_moe_bl_95 <-
     moe(ftfzoi_baseline$prob_sev,ftfzoi_baseline$RS,ftfzoi_base
     line$wt * 10^6, conf.level = .95,
                    psu = ftfzoi_baseline$psu, strata =
     ftfzoi_baseline$strata)$moe * 100
moe_bl_95 <- c(modsev_moe_bl_95, mod_moe_bl_95, sev_moe_bl_95)

# Midline
modsev_moe_ml_95 <-
     moe(ftfzoi_midline$prob_mod_sev,ftfzoi_midline$RS,ftfzoi_mi
     dline$wt * 10^6, conf.level = .95,
                       psu = ftfzoi_midline$psu, strata =
     ftfzoi_midline$strata)$moe * 100
mod_moe_ml_95 <-
     moe(ftfzoi_midline$prob_mod,ftfzoi_midline$RS,ftfzoi_midlin
     e$wt * 10^6, conf.level = .95,
                    psu = ftfzoi_midline$psu, strata =
     ftfzoi_midline$strata)$moe * 100
sev_moe_ml_95 <-
     moe(ftfzoi_midline$prob_sev,ftfzoi_midline$RS,ftfzoi_midlin
     e$wt * 10^6, conf.level = .95,
                    psu = ftfzoi_midline$psu, strata =
     ftfzoi_midline$strata)$moe * 100
moe_ml_95 <- c(modsev_moe_ml_95, mod_moe_ml_95, sev_moe_ml_95)
```

Step 9. Store all results (i.e., the baseline and midline weighted and unweighted number of observations, the estimates for moderate and severe food insecurity, moderate food insecurity, and severe food insecurity, and the MOE and 95 percent CI for each estimate) in a data frame, **AGG_df**.

```
#### Calculate aggregate prevalence levels and put in df ###### #

# create empty DF
AGG_df <- matrix(NA, nrow = 2, ncol = 17)
# add column and rownames
```

```
colnames(AGG_df) = c(“Moderate+Severe_Food_Insecurity”,
      “Moderate_Food_Insecurity”, “Severe_Food_Insecurity”,
                      “N”,”WN”,
                      “MSFI_MoE”, “MFI_MoE”,”SFI_MoE”,
                      “MSFI_CI_Low”, “MFI_CI_Low”, “SFI_CI_Low”,
                      “MSFI_CI_High”, “MFI_CI_High”,
      “SFI_CI_High”,
                      “MSFI_Sig”, “MFI_Sig”, “SFI_Sig”)
rownames(AGG_df) = c(“Baseline”,”Midline”)
# add in values
AGG_df[1, c(1,2, 3)] <- glo_prev_bl * 100
AGG_df[1, c(4)] <- nrow(FIES_baseline_rasch$XX)
AGG_df[1, c(5)] <- round(sum(ftfzoi_baseline$wt * 10^6), 2)
AGG_df[1, c(6, 7, 8)] <- moe_bl_95
AGG_df[1, c(9, 10, 11)] <- (glo_prev_bl * 100) – moe_bl_95
AGG_df[1, c(12, 13, 14)] <- (glo_prev_bl * 100) + moe_bl_95
AGG_df[1, c(15, 16, 17)] <- c(NA, NA, NA)

AGG_df[2, c(1,2, 3)] <- glo_prev_ml * 100
AGG_df[2, c(4)] <- nrow(res_ml$XX)
AGG_df[2, c(5)] <- round(sum(FIES_midline_rasch$wt * 10^6), 2)
AGG_df[2, c(6, 7, 8)] <- moe_ml_95
AGG_df[2, c(9, 10, 11)] <- (glo_prev_ml * 100) – moe_ml_95
AGG_df[2, c(12, 13, 14)] <- (glo_prev_ml * 100) + moe_ml_95
AGG_df[2, c(15, 16, 17)] <- ifelse(((glo_prev_ml * 100) –
      (glo_prev_bl * 100)) > moe_ml_95, “T”, “F”)

# Convert matrix to dataframe and convert row ID to column.
  AGG_df <- AGG_df %>%
         as.data.frame() %>%
         rownames_to_column(“Survey_Round”)

# view results

head(AGG_df)
```

Step 10. Calculate midline estimates for disaggregates.

Assessing food insecurity prevalence when considering key characteristics of households is an important component of analysis. In this step, the disaggregate variables read into R in Step 2 (i.e., residence, gendered household type, shock severity, and wealth quintile) are used to disaggregate moderate and severe food insecurity at midline. A similar process can be repeated for the baseline data, different classifications of food insecurity, or additional disaggregates.

Step 10a. Define the disaggregates to be included in this analysis.

If more disaggregates are needed, create additional groups (`group7`, `group8`, etc.), as done with `group1` to `group6`.

```
# Computing prevalence and MoEs by groups
# survey round
# urban rural
group1 = ftfzoi_midline$urbanrural

group2 = ftfzoi_midline$genhhtype_dj
group2 <- factor(group4, levels = c(1,2,3,4), labels = c(“De jure
     male and female adults”,”De jure female, no male”, “De jure
     male, no female”, “De jure children only”))

group3 <-
     ftfzoi_midline$shock_sev[!is.na(ftfzoi_midline$shock_sev)]
group3 <- factor(group5, levels = c(1,2,3,4), labels = c(“None”,
     “Low”, “Medium”, “High”))

group4 <-
     ftfzoi_midline$awiquint[!is.na(ftfzoi_midline$awiquint)]
group4 <- factor(group6, levels = c(1,2,3,4,5), labels =
     c(“Poorest”, “Second”, “Middle”, “Fourth”, “Wealthiest”))
```

Step 10b. Add the disaggregate variables to the `group_list` and `groups` objects so that they are included in the loop function.

```
Group_list <- list(group1, group2, group3, group4)

groups <- c(unique(as.character(group1)),
     unique(as.character(group2)),
unique(as.character(group3)), unique(as.character(group4)))
```

Step 10c. Create an empty data frame (`mod_sev_fi_ml`) and use a loop function to calculate prevalence, unweighted and weighted number of observations, and the MOE for each disaggregate at midline and store the results in the data frame.

```
# create empty data frame for results
mod_sev_fi_ml <- data.frame()
# loop function
for (I in 1:length(group_list)) {
    for (dis in unique(groups)) {
      if(!(dis %in% group_list[[i]])) next # skip if disaggregate
     not in group
      disag = dis # store disaggregate name
```

```
   fltr = which(group_list[[i]] == dis) #define rows/HHs to
   calc. values
   prob_mods = ftfzoi_midline$prob_mod_sev[fltr] # define
   probabilities
   wt = ftfzoi_midline$wt[fltr]*10^6 # define weights
   rs = ftfzoi_midline$RS[fltr] # define raw scores
   psu = ftfzoi_midline$psu[fltr]  # define primary sampling
   units for MOE
   strata = ftfzoi_midline$strata[fltr] # define strata for
   MOE
# results
   output_1 = disag
   output_2 = length(fltr)
   output_3 = sum(wt)
   output_4 = wtd.mean(prob_mods,wt) * 100
   output_5 = moe(prob_mods,rs,wt,psu=psu,strata=strata,
   conf.level = .95)$moe * 100
   tot_output = c(output_1, output_2, output_3,
   round(output_4, 1), round(output_5, 1)) # assign results to
   vector
   mod_sev_fi_ml <- rbind(mod_sev_fi_ml, tot_output) # assign
   vector to data frame row
  }
 colnames(mod_sev_fi_ml) =
   c“"Disaggregat”",“"Midline_”"”"Midline_W”"”"Midline_MSF”"”"
   Midline_Mo”")  # assign column names
}

# view results
head(mod_sev_fi_ml)
```

Step 10d. If also analyzing baseline data, customize the code in Steps 10a-c to run with the baseline data. Then ensure that the column names have the survey round in each name so that they are unique as shown above (“`Midline_`”, “`Baseline_`”) and merge the two data frames using either the `left_join` or `bind_rows` function so that the midline and baseline estimates can be compared and changes over time assessed.

```
# create empty data frame for results
mod_sev_fi_bl <- data.frame()
# loop function
for (i in 1:length(group_list)) {
      for (dis in unique(groups)) {
      if(!(dis %in% group_list[[i]])) next # skip if
   disaggregate not in group
           disag = dis # store disaggregate name
```

```
                fltr = which(group_list[[i]] == dis) #define
     rows/HHs to calc. values
                prob_mods = ftfzoi_baseline$prob_mod_sev[fltr] #
     define probabilities
                wt = ftfzoi_baseline$wt[fltr]*10^6 # define
     weights
                rs = ftfzoi_baseline$RS[fltr] # define raw scores
                psu = ftfzoi_baseline$psu[fltr]  # define primary
     sampling units for MOE
                strata = ftfzoi_baseline$strata[fltr] # define
     strata for MOE
                # results
                output_1 = disag
                output_2 = length(fltr)
                output_3 = sum(wt)
                output_4 = wtd.mean(prob_mods,wt) * 100
                output_5 =
     moe(prob_mods,rs,wt,psu=psu,strata=strata, conf.level =
     .95)$moe * 100
                tot_output = c(output_1, output_2, output_3,
     round(output_4, 1), round(output_5, 1)) # assign results to
     vector
                mod_sev_fi_bl <- rbind(mod_sev_fi_bl, tot_output)
     # assign vector to data frame row
  }
colnames(mod_sev_fi_bl) =
     c(,“"Disaggregat”",“"Baseline_”"”"Baseline_W”"”"Baseline_MS
     F”"”"Baseline_Mo”")  # assign column names
}

# view results
head(mod_sev_fi_bl)

# create variable identifying survey
mod_sev_fi_ml$Survey <- “Midline”
mod_sev_fi_bl$Survey <- “Baseline”
# join into combined disaggregate data frame
mod_sev_fi_combined <- left_join(mod_sev_fi_bl, mod_sev_fi_ml, by
     = c(“Disaggregate”, “Survey”)
```

Step 10e. After combining the baseline and midline data frames, determine whether the change over time is statistically significant for each disaggregate category by calculating the difference between the midline and baseline estimates and then running an `ifelse` statement as done for the aggregate level in Step 8.

```
# calculate difference- midline-- baseline
mod_sev_fi_combined$Diff <- mod_sev_fi_combined$ Midline--
mod_sev_fi_combined$Baseline_MSFI
# significance test
mod_sev_fi_combined$Sig_test <- ifelse(mod_sev_fi_combined$Diff >
mod_sev_fi_combined$Midline_MoE, “T”, “F”)
```

Step 11. Export the midline results to two CSV files—one with the aggregate results and one with the disaggregate results—using the **`write_csv`** function. The data can then be used for further analysis in another statistical software program or for generating tables. Adjust the code and repeat the step if the baseline results also must be exported to CSV files.

```
# define working directory to save results in a separate folder
setwd“"FIES\\FTF ZOI Survey [COUNTRY] [YEAR] NAME\\Result”")

write_csv(as.data.frame(AGG_df),“"prevalence_food_insecurity_aggr
     egate.cs”")
write_csv(as.data.frame(mod_sev_fi),“"prevalence_mod_sev_food_ins
     ecurity_disaggregates.cs”")
```

Step 12. Export the results to two Stata data files—one with household-level results and one with ZOI-level results—using the **`write_dta`** function.

Step 12a. Ensure that all variables are labeled. This can be done with the **`var_label`** function of the ‘**`labelled`**’ library, which is a subpackage of the ‘**`haven`**’ library. For example:

```
var_label(ftfzoi_midline$prob_mod_sev) <- “Probability of HH
     being moderate or severe food insecure”
var_label(ftfzoi_midline$fi_mod_sev_bin) <- “Binary variable for
     moderate & severe food insecurity”
```

Step 12b. Ensure that all values are labeled using the **`val_label`** function. For example:

```
val_label(ftfzoi_midline$fi_mod_sev_bin, 1) <- “Yes”
val_label(ftfzoi_midline$fi_mod_sev_bin, 0) <- “No” # set yes/no
     labels for the binary variable of moderate/severe food
     insecurity
```

Step 12c. Include the following variables in the **`ftfzoi_midline`** data frame and export to a Stata data file, *FTF ZOI Survey [COUNTRY] [YEAR] FIES*, to be added to the larger household-level data file.

```
Core variables:
     survey – “Survey round”
     hhnum – “Household ID number”
     psu – “Primary sampling unit”
     wt – “HH sampling weight”
       hhsize_dj – “Number of members in household”
     strata – “Stratum”
```

```
      fcluster – “Cluster ID number”

FIES items:
      WORRIED – “Past 12 months: ever worried not enough food”
      HEALTHY – “Past 12 months: ever unable to eat healthy
            foods”
      FEWFOOD – “Past 12 months: ever limited variety of food”
      SKIPPED – “Past 12 months: ever skipped a meal”
      ATELESS – “Past 12 months: ever ate less than should”
      RUNOUT – “Past 12 months: ever did’'t have food”
      HUNGRY – “Past 12 months: ever hungry but did not eat”
      WHLDAY – “Past 12 months: ever did’'t eat for a whole day”
      RS – “Raw score: sum of “yes” responses to FIES Items
      prob_mod – “Probability of being moderately food secure”
      prob_mod_sev – “Probability of HH being moderately or
            severely food insecure”
      prob_sev – “Probability of HH being severely food insecure
      fi_mod_sev_bin – “Moderately or severely food insecure –
            disaggregate”
      fi_mod_bin – “Moderately food insecure – disaggregate”
      fi_sev_bin – “Severely food insecure – disaggregate”

Disaggregates (if calculated):
      ahtype – “Residency type (urban/rural)”
      genhhtype_dj – “Gender household type – de jure household
            members”
      awiquint – “Wealth quintile disaggregate”
      shock_sev – “Shock severity exposure disaggregate”

write_dta(ftfzoi_midline,“"FTF ZOI Survey [COUNTRY] [YEAR] FIES.dt”")
```

Step 12d. Include the following aggregate ZOI-level variables in the `AGG_df` data frame and export to a Stata data file, *FTF ZOI Survey [COUNTRY] [YEAR] NAME Food Insecurity Aggregate Results*.

```
Survey_Round – “Survey Round: Baseline/Midline”
Moderate+Severe_Food_Insecurity – “Estimate of Moderate & Severe
      Food Insecurity Rate”
Moderate_Food_Insecurity – “Estimate of Moderate Food Insecurity
      Rate”
Severe_Food_Insecurity – “Estimate of Severe Food Insecurity
      Rate”
N – “Unweighted N”
WN – “Weighted N”
MSFI_MoE – “Margin of Error – Moderate & Severe Food Insecurity”
MFI_MoE-- “Margin of Error – Moderate Food Insecurity”
SFI_MoE-- “Margin of Error – Severe Food Insecurity”
MSFI_CI_Low – “Lower Confidence Interval – Moderate & Severe Food
      Insecurity”
MFI_CI_Low-- “Lower Confidence Interval – Moderate Food
      Insecurity”
SFI_CI_Low-- “Lower Confidence Interval – Severe Food Insecurity”
```

```
MSFI_CI_High – “Upper Confidence Interval – Moderate & Severe
     Food Insecurity”
MFI_CI_High-- “Upper Confidence Interval – Moderate Food
     Insecurity”
SFI_CI_High – “Upper Confidence Interval – Severe Food
     Insecurity”
MSFI_Sig-- “Significance Test – Moderate & Severe Food
     Insecurity: TRUE/FALSE”
MFI_Sig – “Significance Test – Moderate Food Insecurity:
     TRUE/FALSE”
SFI_Sig – Significance Test – Severe Food Insecurity: TRUE/FALSE”

write_dta(AGG_df,“"FTF ZOI Survey [COUNTRY] [YEAR] NAME Food Insecurity
     Aggregate Results.dt”")
```

Step 12e. Include the following aggregate ZOI-level variables in the `mod_sev_fi_combined` data frame and export to a Stata data file, *FTF ZOI Survey [COUNTRY] [YEAR] NAME Food Insecurity Disaggregate Results*.. **Note:** The variables shown are for only moderate and severe food insecurity; be sure to include the same variables for the other two levels of food insecurity (moderate, and severe) for each disaggregate.

```
Disaggregate – “Name of disaggregate”
FI_Baseline – “Moderate and Severe Food Insecurity Rate,
     Baseline”
CI_Baseline – “Confidence Interval: Moderate and Severe Food
     Insecurity Rate, Baseline”
N_Baseline – “Unweighted N, Baseline”
FI_Midline-- “Moderate and Severe Food Insecurity Rate, Midline”
CI_Midline-- “Confidence Interval: Moderate and Severe Food
     Insecurity Rate, Midline”
N_Midline-- “Unweighted N, Midline”
Diff – “Difference between Midline and Baseline in Moderate and
     Severe Food Insecurity Rate”
Sig – “Significance Test from Midline to Baseline – Moderate and
     Severe Food Insecurity Rate”
MoE_Baseline – “Margin of Error-- Moderate and Severe Food
     Insecurity Rate, Baseline”
MoE_Midline – “Margin of Error-- Moderate and Severe Food
     Insecurity Rate, Midline”
write_dta(mod_sev_fi_combined,“FTF ZOI Survey [COUNTRY] [YEAR]
   NAME Moderate & Severe Food Insecurity Disaggregate
   Results.dta”)
```

15.3 References

Ballard, T., Kepple, A.W., & Cafiero, C. (2013). *The food insecurity experience scale: Development of a global standard for monitoring hunger worldwide*. Technical Paper. Rome: Food and Agriculture Organization of the United Nations. Available at: http://www.fao.org/fileadmin/templates/ess/voh/FIES_Technical_Paper_v1.1.pdf

Cafiero, C., Viviani, S., & Nord, M. (2017). *Food security measurement in a global context: The Food Insecurity Experience Scale*, Measurement. Available at: doi: https://doi.org/10.1016/j.measurement.2017.10.065

Cafiero, C., et al. (2016). *Methods for estimating comparable prevalence rates of food insecurity experienced by adults throughout the world*. Voices of the Hungry Technical Report No 1/2016. Rome: Food and Agriculture Organization of the United Nations. Available at: http://www.fao.org/3/a-i4830e.pdf

Food and Agriculture Organization of the United Nations (FAO). (2019). *Voices of the hungry: Applying the FIES*. Rome: FAO. Available at: http://www.fao.org/in-action/voices-of-the-hungry/using-fies/en/

Food and Agriculture Organization of the United Nations (FAO). (2015). *Modeling food insecurity in bivariate and regression analyses.* Rome: FAO. Available at: https://www.fao.org/3/bp091e/bp091e.pdf

INDDEX Project. (2018). *Food insecurity experience scale (FIES)*. Data4Diets: Building Blocks for Diet-related Food Security Analysis. Tufts University, Boston, MA. Available at: https://inddex.nutrition.tufts.edu/data4diets/indicator/food-insecurity-experience-scale-fies

Nord, M. (2014). *Introduction to item response theory applied to food security measurement: Basic concepts, parameters and statistics*. Rome: Food and Agriculture Organization of the United Nations. Available at: http://www.fao.org/3/a-i3946e.pdf

Onori, F., Viviani S., & Brutti P. (2021). *Towards global monitoring: equating the Food Insecurity Experience Scale (FIES) and food insecurity scales in Latin America.* Available at: https://arxiv.org/pdf/2102.10005.pdf

Viviani, S. (2016). *Manual for the implementation of the FAO Voices of the Hungry methods to estimate food insecurity: RM.weights package in R. Version 2.* Rome: Food and Agriculture Organization of the United Nations. Available at: http://www.sesric.org/imgs/news/1752_Manual_on_RM_Weights_Package_EN.pdf