3 - Descriptive statistics and tables

R labs - Manage Successful Field Research

The World Bank | WB Github

June 2025

Before we begin

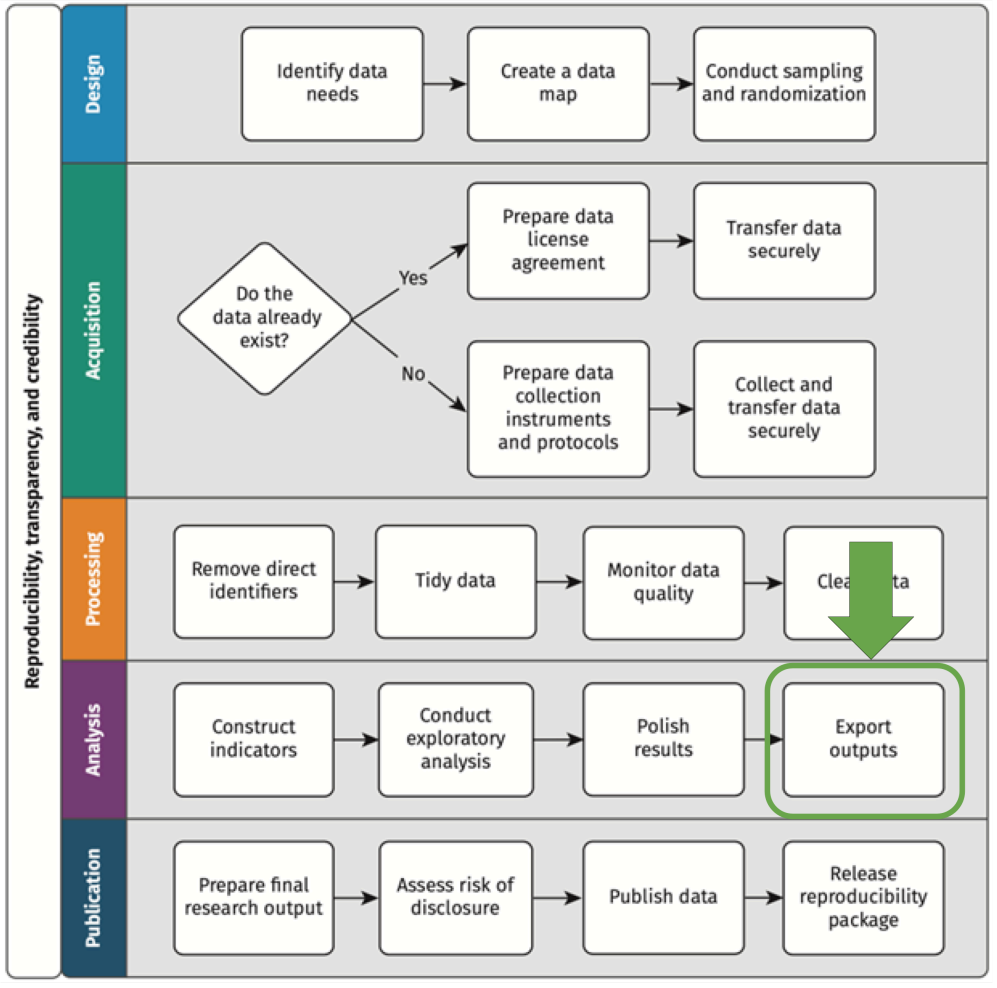
Go to http://bit.ly/msfr25_materials and download

3-descriptive-tables.pdf

Table of contents

About this session

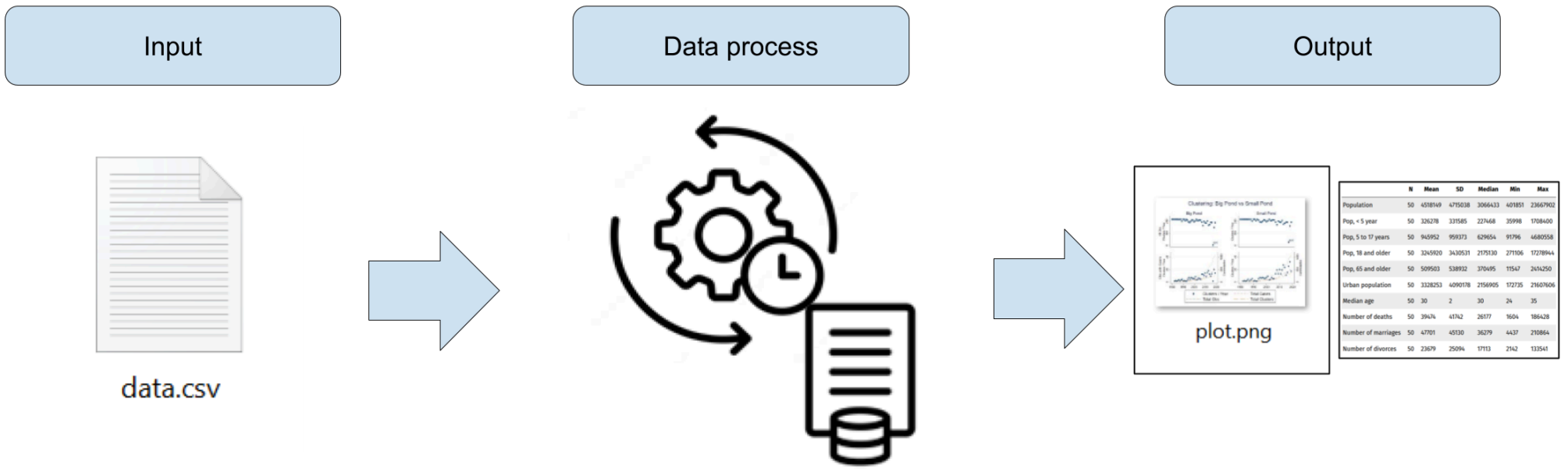
About this session



Creating outputs

Creating outputs

- Until now, we've seen how to produce dataframes or data table file outputs (`.Rds` and `.csv` files)
- Today we'll see how to produce file outputs with more finalized results, namely descriptive statistics, balance, and regression tables
- The objective of this is that the concept of reproducibility can also be also be applied not only in your data work but in your reports as well (though reports is something we're not covering in this course)



Creating outputs

Not reproducible

Anything that requires

- Copy-pasting
- Manual formatting after exported

Creating outputs

Not reproducible

Anything that requires

Copy-pasting

Manual formatting after exported

Reproducible

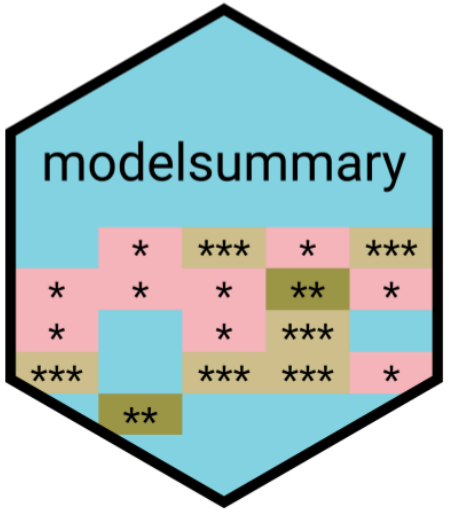
R Markdown: dynamic document containing code and text that is exported directly from R into PDF, HTML, Word, Power Point and other formats

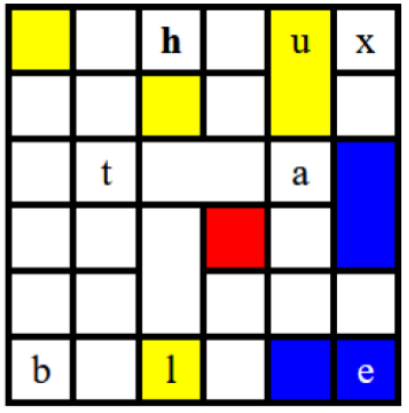
LaTeX: typesetting system used for scientific publications that automatically reloads tables and figures every time the document is rendered

Creating outputs

As usual, there are several options for exporting tables in R. Today we'll use:

- **modelsummary:** a package for creating descriptive statistics and regression tables
- **huxtable:** a package for creating HTML, Latex, and Excel tables from R





More badges for your collection!

Creating outputs

Exercise 1: Install and load packages for generating outputs

- Install `modelsummary` and `huxtable` with:

```
install.packages("modelsummary")
install.packages("huxtable")
install.packages("openxlsx") # this is a dependency for huxtable to export results to Excel
```

- Load all the libraries we'll use with:

```
library(here)
library(haven)
library(dplyr)
library(forcats)
library(janitor)
library(modelsummary)
library(huxtable)
```

Creating outputs

Exercise 2: Read the data

1. Use `here()` and `read_stata()` to read the file in `DataWork/Data/Raw/LWH-households-clean.Rds`. Apply `as_factor()` on the result to transform labeled values into factors.

```
path  <- here("DataWork", "Data", "Raw")
df_hh <- read_stata(here(path, "TZA_CCT_baseline.dta")) %>% as_factor()
```

2. Inspect the dataframe with `View()`

Data exploration

Data exploration

Knowing your data better

- Before starting to produce outputs, it's useful to explore your data so you will know what to export
- You already know `View()`, `nrow()`, and `colnames()`. Some other functions are:
  - `glimpse()`: prints a dataframe in the console (from `dplyr`)
  - `head()`: prints the first six observations of a dataframe
  - `tail()`: prints the last six observations of a datafrma
  - `dim()`: returns a size-two vector with the number of rows and columns in a dataframe

Data exploration

Exercise 3: Exploration with dataframe and column summaries

- Print the summary of `df_hh`

```
summary(df_hh)
```

- Print the summary of the column `hh_size` of `df_hh` (use the `$` operator)

```
summary(df_hh$hh_size)
```

Data exploration

```
# Summary of a dataframe
summary(df_hh)
```

```
##       vid              hhid           enid                     floor
##  Min.   : 1.00   Min.   :1001   Min.   :420.0   Mud/earth      :1655
##  1st Qu.:20.00   1st Qu.:2934   1st Qu.:619.0   Wood/plank     :   0
##  Median :41.00   Median :5026   Median :805.0   Tiles          :   0
##  Mean   :41.39   Mean   :5167   Mean   :716.6   Concrete/Cement: 105
##  3rd Qu.:62.00   3rd Qu.:7122   3rd Qu.:818.0   Grass          :   0
##  Max.   :82.00   Max.   :9922   Max.   :828.0   Other (specify):   0
##
##                roof                  walls                                 water
##  Thatch         :811   Mud/Mud brick  :1526   Uncovered Well                :446
##  Iron sheets    :616   Wood/Bamboo    : 153   River, lake, pond             :421
##  Mud            :333   Concrete/Cement:  81   Pipe bourne water treated     :336
##  Wood           :  0   Stone          :   0   Piped bourne water untreated:208
##  Concrete/Cement:  0   Burnt bricks   :   0   Bore hole/hand pump           :123
##  Roofing tiles  :  0   Iron sheets    :   0   Unprotected spring            : 98
##  (Other)        :  0   (Other)        :   0   (Other)                       :128
##                energy                rel_head            female_head      hh_size
##  Kerosine/paraffin:1320   Muslim         :1067   No             :1072   Min.   : 1.000
```

Data exploration

```
# Summary of a dataframe column
summary(df_hh$hh_size)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   3.976   5.000  17.000
```

Data exploration

Tabulations

- `summary()` is useful to explore quantitative variables
- However, it's not great for character or factor variables
- We'll use `tabyl()` in these cases, from the package `janitor`
- `tabyl()` generates frequency tables from dataframe columns

Exercise 4: Exploration with tabulations

- Tabulate the variable `energy` (main source of energy for lightning) of `df_hh` with:

```
df_hh %>% tabyl(energy)
```

- Cross-tabulate the variables `energy` and `floor` with:

```
df_hh %>% tabyl(energy, floor)
```

Data exploration

```
# Tabulating district
df_hh %>% tabyl(energy)
```

energy	n	percent
Kerosine/paraffin	1320	0.7500000
Gas	2	0.0011364
Main electricity	11	0.0062500
Solar panels/private generator	1	0.0005682
Battery	30	0.0170455
Candles	0	0.0000000
Firewood	393	0.2232955
Other (specify)	3	0.0017045

Data exploration

```
# Cross-tabulating sector and district
df_hh %>% tabyl(energy, water)
```

energy	Pipe bourne water treated	Piped bourne water untreated	Bore hole/hand pump	Covered Well	Uncovered Well	Protected spring	Unprotected spring	Rain water	River, lake, pond	Truck, vendor	Other (specify)
Kerosine/paraffin	264	120	101	76	331	5	70	1	325	27	0
Gas	1	0	0	0	1	0	0	0	0	0	0
Main electricity	8	1	1	1	0	0	0	0	0	0	0
Solar panels/private generator	1	0	0	0	0	0	0	0	0	0	0
Battery	7	7	0	1	8	0	2	0	4	0	1
Candles	0	0	0	0	0	0	0	0	0	0	0
Firewood	55	79	21	8	105	0	26	0	91	6	2
Other (specify)	0	1	0	0	1	0	0	0	1	0	0

Descriptive tables

Descriptive tables

What if you want to...

- ...export the summary statistics to another software?
- ...customize which statistics to display?

Descriptive tables

What if you want to...

- ...export the summary statistics to another software?
- ...customize which statistics to display?

Well, then you will need a few more packages

- There are many packages that can be used both for displaying and exporting summary statistics
- Today we will show you a combination of two packages: `modelsummary` and `huxtable`
- We chose this combination because together, they can perform all the tasks we are interested in
- In fact, `modelsummary` can perform most of them by itself -- with the exception of exporting formatted tables to Excel

Descriptive tables

`modelsummary` contains a family of functions called *datasummary* which can be used to create different types of summary statistics tables. These include:

- `datasummary_skim()`, to create descriptive statistics tables
- `datasummary_balance()`, to create balance tables
- `datasummary_correlation()`, to create a correlation table
- `datasummary_crosstab()`, to create a twoway tabulation
- `datasummary()`, to create customized descriptive statistics tables

Descriptive tables

- `datasummary_skim()` produces a quick summary table of numerical or categorical variables

```
datasummary_skim(
  data,
  type = "numeric",
  output = "default",
  ...
)
```

- It uses the following arguments:
  - `data`: the data set to be summarized, the only required argument
  - `type =`: type of variables in the dataframe to be described
  - `output =`: the type of output desired
  - `...`: additional options allow for formatting customization, such as including notes and titles
- Its most basic use is `datasummary_skim(df_name)` to describe numeric variables

Descriptive tables

Exercise 5: Producing a basic descriptive table

- Use `datasummary_skim()` to create a descriptive statistics table for `df_hh`

```
df_hh %>%datasummary_skim()
```

Descriptive tables

	Unique	Missing Pct.	Mean	SD	Min	Median	Max
anonymous village ID	80	0	41.4	23.9	1.0	41.0	82.0
Household ID	1758	0	5166.8	2531.4	1001.0	5026.0	9922.0
Enumerator ID	12	0	716.6	129.5	420.0	805.0	828.0
Number of members in the household	16	0	4.0	2.6	1.0	3.0	17.0
Number of childeren below or equal to 5 years of age	6	0	0.5	0.8	0.0	0.0	5.0
Number of children between 6 to 17 years of age	11	0	1.2	1.4	0.0	1.0	10.0
Number of Adults (18-59 years)	9	0	1.1	1.2	0.0	1.0	8.0
Number of Elderly (> 60 years)	5	0	1.3	0.7	0.0	1.0	4.0
What is the age of the member [FIRST]?	102	0	45.9	29.4	0.0	57.0	105.0
In the past 12 months, how many times has the member attended the clinic? [FIRST	31	0	2.5	5.0	0.0	1.0	106.0
No. of days in the last 4 weeks the member suffered from the heath problem? [FIR	28	69	10.9	8.9	1.0	7.0	28.0
How much did it cost? [FIRST]	55	75	2816.8	15053.8	-88.0	1000.0	300000.0
No. of days member was unable to perform dailt acitvities due to illness? [FIRST	97	36	29.3	47.6	0.0	14.0	365.0
What is the age of the member [SECOND]?	99	17	39.2	28.6	0.0	35.0	99.0
In the past 12 months, how many times has the member attended the clinic? [SECON	29	17	2.7	4.6	0.0	1.0	73.0
No. of days in the last 4 weeks the member suffered from theheath problem? [SECO	27	77	10.1	8.7	1.0	7.0	28.0
How much did it cost? [SECOND]	50	82	3689.5	14430.5	-88.0	1000.0	200000.0
No. of days member was unable to perform dailt acitvities due to illness? [SECON	96	49	29.1	50.2	0.0	13.0	365.0
Annual food consumption value	965	0	638718.4	513109.5	5096.0	517400.0	4903600.0
Annual non-food consumption value	1328	0	146788.5	250899.1	0.0	69933.0	4108400.0
How much land do you farm? AREA	42	8	1.3	1.1	0.1	1.2	32.4
How many livestock does the HH own today?	38	0	3.0	6.5	0.0	0.0	160.0
How many livestock did the HH own 12 months ago?	54	0	6.6	10.2	0.0	3.0	150.0

Descriptive tables

- `datasummary_skim()` summarizes only numeric variables by default
- To summarize categorical variables, use the argument `type = "categorical"`

```
# Selecting only one variables so table fits
df_hh %>% select(energy) %>%
  datasummary_skim(type = "categorical")
```

energy	N	%
Kerosine/paraffin	1320	75.0
Gas	2	0.1
Main electricity	11	0.6
Solar panels/private generator	1	0.1
Battery	30	1.7
Candles	0	0.0
Firewood	393	22.3
Other (specify)	3	0.2

Descriptive tables

- Other than `datasummary_skim()`, we can use `datasummary()` to customize the variables and statistics to include using a formula

```
datasummary(
  var1 + var2 + var3 ~ stat1 + stat2 + stat3 + stat4,
  data = data,
  ...
)
```

- The arguments of `datasummary()` are:
  - `formula`: a two-sided formula to describe the table: rows ~ columns
  - `data=`: the data set to be summarized
  - `...`: additional options allow for formatting customization

Descriptive tables

Exercise 6: Producing a table with more information

- Create a table showing the number of observations, mean, standard deviation, minimum, maximum, and median value for the variables `hh_size`, `n_child_5`, and `n_elder` in `df_hh` with the code below

```
datasummary(
  hh_size + n_child_5 + n_elder ~ N + Mean + SD + Median + Min + Max,
  data = df_hh
)
```

Descriptive tables

```
datasummary(
  hh_size + n_child_5 + n_elder ~ N + Mean + SD + Median + Min + Max,
  data = df_hh
)
```

	N	Mean	SD	Median	Min	Max
hh_size	1760	3.98	2.61	3.00	1.00	17.00
n_child_5	1760	0.47	0.80	0.00	0.00	5.00
n_elder	1760	1.26	0.72	1.00	0.00	4.00

Descriptive tables

- The package `modelsummary` doesn't offer an option to export tables to Excel
- To do that, we will first store the result of `datasummary()` into an object of type "huxtable"
- This will allow us to use the function `quick_xlsx()` from the package `huxtable` to export the table

Descriptive tables

Exercise 7: Exporting descriptive tables to Excel

1. Save the last table you created to and object called `descriptives_HH_members` and add the argument `output = "huxtable"`
2. Export this object to Excel with `quick_xlsx()`

```
# Storing summary table into a huxtable object
descriptives_income <-
  datasummary(
    hh_size + n_child_5 + n_elder ~ N + Mean + SD + Median + Min + Max,
    data = df_hh,
    output = "huxtable"
  )

# Exporting
quick_xlsx(
  descriptives_income,
  file = here("DataWork", "Outputs", "descriptives_HH_members.xlsx")
)
```

Descriptive tables

The result in Excel will look like this:

	N	Mean	SD	Median	Min	Max
hh_size	1760	3.98	2.61	3	1	17
n_child_5	1760	0.47	0.8	0	0	5
n_elder	1760	1.26	0.72	1	0	4

Descriptive tables

The result in Excel will look like this:

	N	Mean	SD	Median	Min	Max
hh_size	1760	3.98	2.61	3	1	17
n_child_5	1760	0.47	0.8	0	0	5
n_elder	1760	1.26	0.72	1	0	4

This might be okay. But if we want to add further customizations to make it truly look like a finalized output, at least two things are missing:

1. Variable labels
2. Some table formatting

We'll add these in the next exercise

Exercise 8: Exporting a formatted table to Excel

1. Create a new dataframe changing the variable names of `hh_size`, `n_child_5`, and `n_elder` with the following code:

```
df_table <- df_hh %>%
  select(
    `Household size`                    = hh_size,
    `Number of members aged 5 or below` = n_child_5,
    `Number of elderly (>60 years)`     = n_elder
  )
```

Note that the "labels" are enclosed in backticks. For R, they are not variable labels but the actual variables names of `df_table`. Variable names in R **can have space characters**. Whent they do, you enclose them in backticks so R understands where the variable name starts and ends.

Exercise 8: Exporting a formatted table to Excel

2. Use this new dataframe to produce a datasummary and add `theme_basic()` on top of it

```
desc_income <-
  datasummary(
    All(df_table) ~ N + Mean + SD + Median + Min + Max,
    data = df_table,
    output = "huxtable"
  ) %>%
  theme_basic()
```

3. Export the result with `quick_xlsx()`

```
quick_xlsx(
  desc_income,
  file = here("DataWork", "Outputs", "descriptives_HH_members_formatted.xlsx")
)
```

Descriptive tables

Now the result will look like this:

	N	Mean	SD	Median	Min	Max
Household size	1760	3.98	2.61	3	1	17
Number of members aged 5 or below	1760	0.47	0.8	0	0	5
Number of elderly (>60 years)	1760	1.26	0.72	1	0	4

Descriptive tables

Now the result will look like this:

	N	Mean	SD	Median	Min	Max
Household size	1760	3.98	2.61	3	1	17
Number of members aged 5 or below	1760	0.47	0.8	0	0	5
Number of elderly (>60 years)	1760	1.26	0.72	1	0	4

- We used the theme function `theme_basic()` for this table.
- `datasummary()` allows to apply multiple customization and themes to the results. You can explore them here. You can also manually apply changes directly to the Excel output.

Balance tables

Balance tables

- The library `modelsummary` has a function for easily exporting balance tables: `datasummary_balance()`

```
datasummary_balance(
  ~ balance_variable,
  data  = data
  stars = FALSE,
  title = "Table title",
  note  = "Table footnote",
  ...
)
```

- The basic arguments of `datasummary()` are:
  - `balance_variable`: name of the variable defining groups (i.e.: treatment variable)
  - `data=`: the data set to be summarized

Balance tables

```
datasummary_balance(
  ~ balance_variable,
  data  = data
  stars = FALSE,
  title = "Table title",
  note  = "Table footnote",
  ...
)
```

- Additional options:
  - `stars=` Logical value (`TRUE`, `FALSE`) for the inclusion of statistical significance stars
  - `title=` String with table title
  - `note=` String with table footnote
  - `...`: additional options allow for formatting customization

Exercise 9: Export a formatted balance table to Excel

1. Create a new dataframe keeping only HHs with "Yes" or "No" for `female_head` and changing the variable names of `hh_size`, `n_child_5`, and `n_elder` with the following code:

```
df_balance <- df_hh %>%
  mutate(female_head = droplevels(female_head)) %>% # removes unused levels of factor variable
  select(
    female_head,
    `Household size`                    = hh_size,
    `Number of members aged 5 or below` = n_child_5,
    `Number of elderly (>60 years)`     = n_elder
  )
```

Balance tables

Exercise 9: Export a formatted balance table to Excel

2. Generate a formatted balance table with `datasummary_balance()`:

```
balance_table <-
  datasummary_balance(
    ~ female_head,
    data   = df_balance,
    stars  = TRUE,
    title  = "Balance by sex of HH head",
    note   = "Includes HHs with observations for baseline and endline",
    output = "huxtable"
  ) %>%
  theme_basic()
```

Balance tables

Exercise 9: Export a formatted balance table to Excel

3. Export the result with `quick_xlsx()`:

```
quick_xlsx(
  balance_table,
  file = here("DataWork", "Outputs", "balance_table.xlsx")
)
```

```
## Warning in file.create(to[okay]): cannot create file
## 'C:/WBG/repos/manage-successful-field-research/2025/R/Data/DataWork/Outputs/balance_table.xlsx',
## reason 'No such file or directory'
```

Balance tables

Balance by sex of HH head (Female = 1)						
	No (N=1072) / Mean	**No (N=1072) / Std. Dev.**	**Yes (N=688) / Mean**	**Yes (N=688) / Std. Dev.**	**Diff. in Means**	**Std. Error**
Household	4.5	2.7	3.2	2.2	-1.3***	0.1
Number of	0.5	0.8	0.4	0.7	-0.1***	0
Number of	1.4	0.8	1	0.6	-0.4***	0
Includes HHs with observations for baseline and endline						

Regression tables

Regression tables

- `modelsummary` has a function called `modelsummary()` for exporting regression tables
- However, we'll use `huxreg()` from `huxtable` as it involves using only one package and it's a complete solution
- That said, there are several other R packages that export regression tables. We recommend checking `stargarzer`, though it should be noted that it only exports tables in text, HTML, and LaTeX formats (no Excel)

Regression tables

Exporting regression tables also involves a two-step process, similar to how we first (1) generated a balance table with `datasummary_balance()` and (2) exported it with `quick_xlsx()`.

1. First, estimate your regressions. You can use R's base command `lm()` (short for *linear model*) or functions from regression estimation libraries.

```
model1 <- lm(y ~ x1,           data = df)
model2 <- lm(y ~ x1 + x2,      data = df)
model3 <- lm(y ~ x1 + x2 + x3, data = df)
```

2. Then, export the regression results into a table with `huxreg()`

```
huxreg(model1, model2, model3) %>%
  quick_xlsx(file = "my-regresion.xlsx")
```

Regression tables

Regressions using `lm()` follow this syntax

```
lm(
  formula,
  data,
  ...
)
```

- `formula` specifies the regressed variable and covariates. It follow the structure: `y ~ x1 + x2`
  - `y`: regressed variable
  - `x1`, `x2`: covariates (separated by a `+`)
  - if one of the covariates is a factor, then R interprets it's a fixed effects variable
- `data`: dataframe to use
- `...`: additional options such as weights, how to treat NAs, and others

Regression tables

Exercise 10: Estimate regression models

1. Estimate three regressions of `livestock_now` on (1) `hh_size` only; (2) `hh_size` and `livestock_before`; (3) `hh_size`, `livestock_before`, and `energy` (note that this last variable is a factor). Let's not focus on the correctness of proposing these models for now.

```
m1 <- lm(livestock_now ~ hh_size,                          data = df_hh)
m2 <- lm(livestock_now ~ hh_size + livestock_before,       data = df_hh)
m3 <- lm(livestock_now ~ hh_size + livestock_before + energy, data = df_hh)
```

2. Check the results of `m3` using `summary()`.

```
summary(m3)
```

Regression tables

```
summary(m3)
```

```
##
## Call:
## lm(formula = livestock_now ~ hh_size + livestock_before + energy,
##     data = df_hh)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -51.007  -1.199  -0.370   0.484 127.815
##
## Coefficients:
##                                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)                          -0.04414    0.26460  -0.167   0.8675
## hh_size                               0.20723    0.05207   3.980 7.18e-05 ***
## livestock_before                      0.32929    0.01320  24.945  < 2e-16 ***
## energyGas                             7.09280    3.80675   1.863   0.0626 .
## energyMain electricity               11.64309    1.62722   7.155 1.22e-12 ***
## energySolar panels/private generator -1.73625    5.36515  -0.324   0.7463
## energyBattery                         0.10973    0.99184   0.111   0.9119
## energyFirewood                       -0.25835    0.31754  -0.814   0.4160
## energyOther (specify)                 0.53998    3.10417   0.174   0.8619
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.363 on 1751 degrees of freedom
## Multiple R-squared:  0.3317,    Adjusted R-squared:  0.3287
## F-statistic: 108.6 on 8 and 1751 DF,  p-value: < 2.2e-16
```

Regression tables

- Now the last step is to export the results with `huxreg()` and `quick_xlsx()`
- `huxreg()` takes the models as the first arguments and then uses named arguments for regression table customization
- Some of its most useful named arguments are:
  - `coefs`: labels for covariates
  - `omit_coefs`: covariates to omit
  - `stars`: levels for significance stars
  - `statistics`: which statistics to show at the bottom of the table
  - `add_rows`: adds rows with additional information

Regression tables

Exercise 11: Export a formatted regression table to Excel

Use the following code to export `m1`, `m2`, and `m3` to a regression table in Excel.

```
models <- list("Model A" = m1, "Model B" = m2, "Model C" = m3)

huxreg(
  models,
  omit_coefs = c("Intercept", "energy"),
  coefs = c(
    "Household size" = "hh_size",
    "Livestock owned 12 months ago" = "livestock_before"
  ),
  statistics = c(
    "N"  = "nobs",
    "R2" = "r.squared"),
  stars = c(`***` = 0.01, `**` = 0.05, `*` = 0.1),
  note = "Includes HHs with observations in baseline and endline"
) %>%
  add_rows(
    c("Energy FE", "No", "No", "Yes"),
    after = 5
) %>%
  theme_basic() %>%
  quick_xlsx(
    file = here("Datawork", "Outputs", "regression_table.xlsx")
)
```

Regression tables

	Model A	Model B	Model C
Household size	0.608 ***	0.236 ***	0.207 ***
	(0.058)	(0.052)	(0.052)
Livestock owned 12 months ago		0.335 ***	0.329 ***
		(0.013)	(0.013)
Energy FE	No	No	Yes
N	1760	1760	1760
R2	0.059	0.310	0.332
Includes HHs with observations in baseline and endline			

Thanks! Gracias! Asante!