Farmfit Intelligence

IDH Primary Data Collection Methodology Document

Version 1 (January 2020)

Data Collection Methodology

1. Background and Rationale

Farmfit is a program funded by Melinda and Bill Gates Foundation (BMGF) and the UK Department for International Development (DfID), implemented by IDH, that seeks to transform inclusive agricultural markets. It's target is to increase the income of 1.1 million smallholder farmers by at least 30%, in five years.

Farmfit assumes a market based approach: companies that offer services to farmers - such as those selling fertilizers or purchasing crops for export - request Farmfit/IDH to help them analyze and improve the services they offer to smallholder farmers. IDH has developed a standardized analysis tool to determine the efficiency, effectiveness, sustainability and scalability of smallholder engagement models, which is referred to as Service Delivery Model (SDM). The analysis is done by working with the companies to gather 1) financial company data on the services delivered to smallholder farmers and 2) farm economics of their producer base.

For the second component, the farm economics of the producer base, accurate information on smallholder farmers is needed. From July to December 2019 a standardized approach for collecting primary data from smallholder farmers was developed. IDH uses the collected data as: a) input for the SDM model, b) as input for the Farmfit M&E system and c) as a resource for learning .

IDH contracted a third party (hereon named 'the data collector') for a six month pilot period to facilitate the design of the data collection instruments, carry out the data collection, deliver the data to the IDH Farmfit team, and document the methodology. The data collector was contracted to collect farm level data for 10 SDM analyses (hereon named 'cases') within the Farmfit program. At the moment of writing this document 8 out of the 10 cases were completed. The jointly developed methodology is explained in this document.

2. Objective

The objective of the pilot was to generate primary farm level data that would:

1. Provide descriptive and numerical input for the SDM model
2. Generate baseline data for Farmfit indicators
3. Generate comparable aggregate data that allows comparison and learning across companies/SDMs

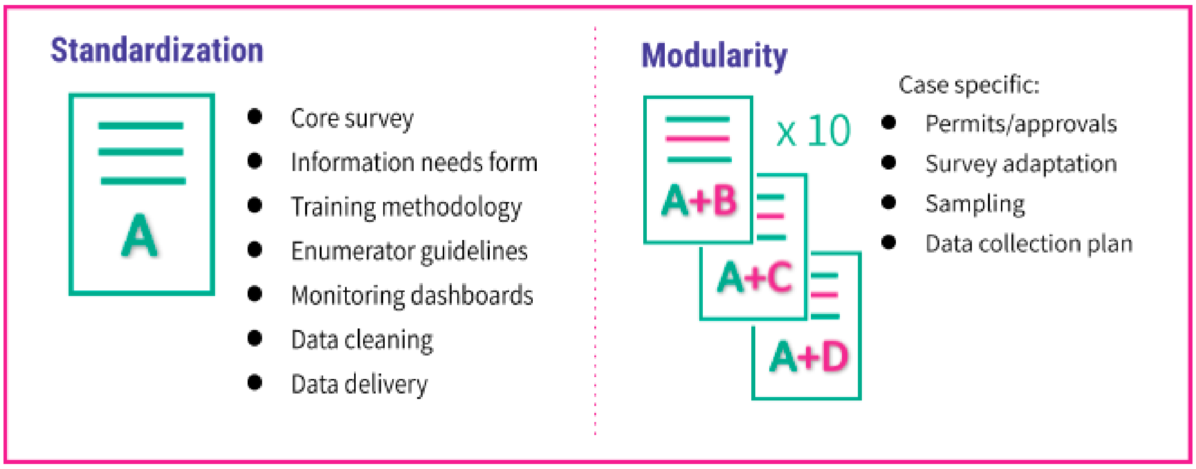
3. Areas of exploration/research

The research topics guiding the design of the household survey were refined during a design workshop at the start of the pilot and took into account the Farmfit program's Theory of Change, monitoring framework, the farm-level data needs for the SDM analysis and the insights from key IDH Farmfit staff. Data collected through the household survey seeks to explore the following categories:

1. Household demographics/ farmer profiles (incl. age, #of household members, technology use, access to finance)
2. Farm characteristics (land size, crops, practices used (e.g. labour, inputs,)
3. Farm(er) economics (revenue, costs to ultimately calculate net income)
4. Farmer resilience (incl. gender dynamics, FSN and climate)
5. Farmer service (incl. farmer-company interaction, customer satisfaction)

4. Approach

Key principles during the design of the primary data collection methodology were standardization and scalability. The primary data collection is envisioned to become an integral part of the SDM analysis and thus needs to be cost-effective and easy to replicate. Due to the different nature of each case, a primary data collection is a combination between standardized and modular approach:



4.1 The Survey

After defining the research topics and indicators for the primary data collection, the data collector reviewed smallholder farmer surveys from content experts as input for designing the survey, building on the experience and recommendations of other organisations[1][2][3][4][5]. The survey questions were then sorted into question groups representing the following topics:

General

1. Introduction
2. Informed consent

Farm characteristics & economics

3. Farm
4. Crop revenue
5. Other farm income
6. Off-farm income
7. Labour cost
8. Equipment cost
9. Inputs
10. Climate resilience

Farmer service

11. Customer satisfaction

Household demographics

12. Household

Farmer resilience

13. Food security
14. Gender
15. PPI
16. Money and mobile
17. Loans
18. Negative liquidity / cash flow

The questions were split into two categories: standard questions (to be used in each survey) and case-specific questions (based on the context). All questions are stored in an excel database. Case specific questions are added to the database and clustered to create modules for specific topics, like ‘rice’ or ‘financial service provider’ so they can be reused for new cases. A full list of questions and their characteristics can be found in annex 3.

After five primary data collections, the questions were carefully evaluated based on their purpose and scalability. This led to removing and revising a significant number of questions and to rephrase questions for further standardization.

4.2 The Sample

Each case had a different population size and (geographic) distribution of associated farmers. Available information on the smallholder farmer population varied per case as well. The available population information and sampling frame we on a case-by-case basis and a corresponding sampling strategy

[1] https://www.agrilinks.org/sites/default/files/ftf-indicator-handbook-march-2018-508.pdf
[2] https://www.agrilinks.org/post/data-suggest-feed-future-reaching-benefiting-and-empowering-women
[3] https://www.fantaproject.org/sites/default/files/resources/MAHFP_June_2010_ENGLISH_v4.pdf
[4] https://www.povertyindex.org/
[5] http://www.fao.org/sustainable-development-goals/indicators/231/en/

was developed to at a representative sample, aiming for 95% confidence level and a confidence interval of +/- 5%.

The survey is a multiple indicator survey, therefore the key indicator chosen to calculate sample size was initially farmer income. The formula to be used was:

$$ss = \frac{\text{Z-score}^2 * \text{st.dev.}^2}{(\text{Margin of Error})^2}$$

Z-score= 1.96

St.dev = Standard deviation

Margin of error

However, a reliable standard deviation for farmer income proved difficult to obtain for the cases so a more generic approach was adopted based on the formula noted below. A 20-30% adjustment was added to the resulting sample size to allow for errors in the sampling frame and potential non-reponse.

$$ss = \frac{Z^2 (p) * (1-p)}{c^2}$$

ss = Sample size

Z = Z value 1.96

p = 0.5

C = confidence interval (+/- .05)

The replacement strategy relied on the type of sampling applied and varied per case according the information available about the population. No stratification was considered as this would significantly increase the sample size, which was out of the scope of the assignment.

4.3 Data delivery

Data from the survey was delivered in three ways: raw, cleaned and aggregated. To comply with the European Data protection and Privacy law GDPR, only a limited number of people have access to the raw data. The data analysts do not need to know farmer names, phone numbers or addresses and receive a cleaned and anonymized data sheet. Furthermore, to lighten the workload of the analysts working on the SDM model, aggregates of the results are constructed and delivered with the data.

4.3.1 Data quality measures

IDH and the data collector together developed a 'information need sheet' (see Annex 3) that was used to collect case-specific information from the SDM company prior to the data collection. This information provided the data collector with ranges on farmgate prices and other data that is of use in validating the primary data against contextual information. The information was also used to define answer options for case-specific survey questions contributing to data quality and minimizing outliers.

Farmfits main indicator is net farmer income. However, net farmer income over a period of 12 months is notoriously hard to measure and can be subject to recall bias. To assess income data quality the data collector proposed to triangulate income information with the Poverty Probability Index (PPI)[6], which is an indication of wealth of the farmer by other means than income. The PPI is different for each country and is therefore part of the modular approach.

Built-in software features of the mobile data collection tool, like skip-logic, mandatory questions, double entering of numerical values and adding min. and max. values to numerical questions, prevent the enumerator from making mistakes. Exhaustive answer options, for example "I don't know" and "I prefer not to answer this question" were added as answer options for multiple choice questions to minimize empty values or incorrect data.

During data collection, incoming data was visualised near real-time in a data collection tracking dashboard. This allowed the data collection supervisor to get an immediate overview of the GPS location of data collection, surveyal time, as well as the number of data points collected by each enumerator. Furthermore, error-prone questions were visualised in plots (for example: acres of SDM crop vs. kilograms of SDM crop produced) to scout outliers. In case of doubts of data quality, the supervisor would call enumerators to ask them for an explanation and/or to clarify the question.

Apart from the data quality dashboard, the data collector's data science team performed spot checks during the data collection, in order to review data quality. Conducting a thorough review of the data quality by looking at question dependencies, common enumerator mistakes and questions that have not been answered, mistakes were detected early on. This allowed the data collection supervisor to mitigate data quality issues.

[6] PPI questions vary per country. It was decided to only use PPI questions that were updated within the past 10 years, as older PPIs are outdated and won't lead to reliable data for triangulation.

4.3.2 Data cleaning

Different measures were taken to arrive at a clean data set and minimize errors. First of all, outliers were removed from the dataset. To determine outliers for the numerical questions of the survey, a cut off of three standard deviations from the corresponding mean was used. Higher or lower values than this cut off, were set to '9997' and not incorporated in further analysis.

While constructing the survey, open text questions were avoided as much as possible, as it leads to ambiguity and inefficiency. As each case is highly contextual, answer options may not always be exhaustive. Therefore, the option 'other' including a free text field was added for most multiple choice answers to not lose important information. These answers were cleaned with standard text cleaning, such as removing punctuation and setting all uppercase words or letters to lowercase, in order to recognise recurring answers.

In the Farmfit survey two types of measurement units are asked: land size and crop measurements (e.g. 100 kg bags). As the commonly used measurements for both these types can differ between countries and crop, these values are converted to a single measurement after the data is collected. By default, in the data delivery all land size measurements are converted to acres and all crop measurements to kilograms.

4.3.3 Descriptive statistics

For numerical questions the minimum value, maximum value, the average and the standard deviation are determined. The standard deviation is calculated by taking the square root of the sample variance, which we determine with the following formula (Stats package, R core team):

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

For categorical questions the frequency of the options is determined and compared to the number of farmers that filled out the question by means of a percentage.

4.3.4 Delivery format

Data is analyzed using R, and R markdown is used for the delivery of the indicators to the IDH analysts. R markdown incorporates R code with text in a clean and orderly way. This makes it easy for analysts to reuse the code while making it easy for people who do not use R to read and understand the transformations. A standardized R script is run by the data collector to perform all data cleaning steps (see annex 4).

5. Operational Plan

The data collector assigned one staff member per case, who plans and oversees the data collection on location. This staff member is responsible for planning, logistics and training, and is supported by a dedicated team that designs the sampling strategy, validates data quality remotely, and helps with data delivery. Furthermore, there are one or more dedicated staff members from both IDH and the company who provide input on the survey design, who share lists of farmers and help appoint extension workers or field staff that help to locate the farmers.

5.1 Timeline

Each primary data collection followed a sequence of steps to ensure timely data collection in a cycle of eight weeks from the moment of the kick-off meeting to the company visit of IDH. In practice, however, the timelines may be significantly shorter than eight weeks.

	Weeks							
Activity	1	2	3	4	5	6	7	8
1. Information needs form filled out	(IDH)							
2. Kick-off meeting with IDH & SDM Company	🟩							
3. Adjusting Survey		🟩						
4. Sample design finalised (based on database)		🟩						
5. Enumerator selection			🟩					
6. Data collection plan			🟩					
7. Training enumerators				🟩				
8. Real-time monitoring dashboard				🟩				
9. Primary data collection				🟩	🟩			
10. Spot checks primary data collection				🟩	🟩			
11. Data cleaning and delivery						🟩		
12. Country visit IDH								(IDH)

5.2 Enumerator selection

The number of enumerators recruited for each case depended on the sample size and the time available for data collection (usually 5 to 6). Each enumerator on average collects five data points a day. On average the survey takes 45 minutes per farmer taking, but enumerators need enough time to reach the next interviewee and conditions in the field can be difficult. The following was used for enumerator recruitments:

*Number of data points needed / (5 days * 5 data points) = number of enumerators*

When the SDM company had used enumerators before, these enumerators were contacted, because their knowledge of the area and the local farmers support a smooth data collection process. If the SDM company had no prior experience with enumerators, a ToR was shared at the nearest university that offers a specialization in agriculture. This would ensure recruitment of well-educated enumerators with enough agricultural knowledge to understand the survey well.

The following factors were considered while selecting enumerators:

- Prior experience with mobile based data collection
- Background in agriculture
- Fluency in local language/dialect
- Familiarity with the geographic area of data collection

5.3 Data collection plan

Before sending enumerators to the farmers, a data collection plan was created detailing which enumerators visit which farmers and on which days. Creating a data collection plan remotely without knowing the context / geographic area proved difficult. Therefore, the data collection plan is usually designed together with a representative from the SDM company and/or local extension officers before the start of the training workshop.

The following factors were considered while designing the data collection plan:

- Is there an equal distribution in travel time between enumerators?
- An extra margin of farmers is needed for each day, in case farmers are not be home.
- Farmers need to be informed about the interview at least one day in advance.
- Formation of enumerator groups, in case they need each other's help.
- Field guides for each enumerator group to help them locate selected farmers

5.4 Enumerator Training

Once enumerators were identified, the data collector facilitated a one and a half day training. The training workshop consisted of learning how to use the data collection application on the smartphone, understanding the survey, practicing interviewing techniques, and learning how to troubleshoot during field data collection.

Planning	Module	Description
0,5 day	Mobile data collection	1. Downloading mobile data collection application 2. Getting to know all the features of the application 3. Calibrating GPS signal on the phone 4. Battery saving options in the field
0,5 day	Understanding the survey	1. Background of primary data collection 2. Going through survey question by question 3. Survey best practices and informed consent
0,5 day	Practicing (in the field)	1. Practicing the survey in the field or amongst each other (if field is too far from training location).

At the end of the training workshop, the enumerators would have group discussions on challenges encountered in the field and had the opportunity to clarify questions about the survey. IDH and the data collector jointly developed an enumerator field guide that explains agricultural and financial concepts and that they can fall back on during data collection if they have difficulties with answering specific questions (see annex 2).

6. Scope

Data collection is organized per each of the SDM cases selected by IDH Farmfit. A case is related to one company that the program has chosen to support in the process of improving services. Each company offers different types of services to farmers and vary significantly in the number of farmers it serves and in the types of crops associated with them. To be able to deliver within budget and timeframe, the data collector and IDH agreed on the following scope:

Scope of data collection		
Factor	**Scope**	**Implication**
Population definition	Smallholder farmers[7]	Medium and large scale farmers that are associated with company are not taken into account in SDM analysis.
Sample size	Up to 375 farmers per case,	Stratification of farmers is out of scope

[7] Smallholders are defined by the SDM companies, as definition of smallholder varies per country and crop.

	depending on population size	
Geography	1 to 2 areas (e.g. provinces, districts…)	SDM company which operates nation-wide needs to choose 1 or 2 geographic areas that are representative of their portfolio.
SDM Crop	1 to 2 SDM crops	SDM company which focuses on various SDM crops needs to choose 1 or 2 crops to focus on.
Time	The last 12 months	The data only paints a picture of the past 12 months.

7. Ethical Considerations

The data collector abides by the principle of do no harm. Farmers are given an explanation on how the data will be used and are given the option to participate in the survey or not through an informed consent question that is built into the survey for each interviewee. (i.e. farmer or female decision maker in household).

Furthermore, the data in the survey tool is only accessible to data collector team members that either built the survey or analyse the data. The analysed data is shared with IDH in two formats: raw, only accessible to the program manager and data specialist from IDH, and in an anonymised and cleaned version that is accessible to the Farmfit team. The raw data is not shared with the SDM company.

Data is stored in its original format on a cloud based database as a service from the data collector. After cleaning of the data the data is uploaded to SamePage (IDH's document storage system), where it is made available for only two staff members of IDH. An anonymized data delivery file is shared with the SDM team on samepage. Farmer names are taken out of the file. The data collector checks the dataset on any data that can reveal the identity of the farmers. Once uploaded, the data collector will contact IDH, who will perform a second check on the anonymisation of the dataset.

The data collector has signed a non-disclosure agreement and a data processing agreement with IDH. The data collector is not allowed to share any of the data nor the farmer databases of the SDM company with any third party and does not have ownership of the data at any point in time. Enumerators sign a non-disclosure agreement with the data collector in which they state they will not share any data with third parties and will delete all information related to the farmers after the end of the assignment. If IDH chooses to stop working with the data collector, the data in the cloud based database will be deleted.

8. Annex 2: Enumerator Guideline

Informed consent

Hello. My name is __________________________________. I am working with *Data collector* and IDH. We are conducting a survey about farming services and other topics in *Location X*[8]. IDH is an international organization that seeks to improve service that farmers receive. The information we collect will help IDH plan activities with companies that offer services to farmers. Your household was selected for the survey, based on your relationship with *Company Z*. I would like to ask you some questions about your household and your farm. The questions usually take about 45 minutes. All of the answers you give will be anonymous (your name will not be tied to the answers) and will not be shared with anyone other than members of our survey team and IDH – Coscharis will not receive your answers. You don't have to participate in the survey, but we hope you will agree to answer the questions since your views are important and could help improve the services farmers receive. If I ask you any questions you don't want to answer, just let me know and I will go on to the next question or you can stop the interview at any time.

Correspondent

Who to interview? Interview the person in charge of the farm. This would be the person who owns the land or is the caretaker (in the absence of the actual owner).

Farmer

- **A small-holder farmer:** *2 hectares*[9] of land and below
- If a farmer farms on more than *2 hectares*, please **stop** the interview and continue to the next farmer.

Farm

The size of the farm is the size the farmer has grown his/her crops only.

Multiple farm plots: When asking about farmsize ask the small-holder farmer if he has multiple plots and add up all plots to get to total farmsize. If it's bigger than *2 hectares* of land, the farmer is not considered a small-holder farmer.

Measurement units:

Plot size: *Hectares*

Weight: *Kilograms (kg)*

Most common packaging unit for rice seeds and rice grains: *Bags of 100 KG or 300 KG*

Contextual information:

Farmgate price per *Packaging type (e.g. 30 kg bags)***:** min. *100 currency X*, max. *200 currency X*

[8] All values in red should be replaced based on the information from the information needs form in Annex 1

[9] There is no standard definition for a smallholder farmer. The maximum number of acres or hectares for a farmer to be considered as smallholder is determined on a case by case basis by asking the company what they consider a smallholder.

Farmgate price per *Packaging type (e.g. 50 kg bags)***:** min. *100 currency X*, max. *200 currency X*

Number of harvest seasons in a year: *One* harvest season

Yield per acre of crop: min. *1500 kg/hectare* max. *3500 kg/hectare*

Reasons for crop loss: *Flooding, non-certified seeds*

Note:
All questions about labour, equipment and inputs refer to the SDM crop. We don't need to know labour, equipment and inputs for other crops the farmer grows.

Unit of measurement for the crop: In case the farmer says the amount of kilograms of *crop* per e.g. bag varies, ask for the minimum and maximum kilograms and answer by taking the average
Example: Farmer answers that the kg per bag vary from a minimum of 100 kilogram to a maximum of 120 kilogram. You answer by taking the average, which is 110 kilograms per bag.

FARM

Size of the farm: Be sure to double-check in which unit of measurement the farmer is answering the question (acres, hectares, etc.). In *Country Z*, farms are usually measured in *hectares.*
The answers can be: 0.5 - 2 hectares for small-holder farmers

> ***The tool tip of this question****: Make sure not to include the house of the farmer. Enter the number '9999' if the answer is interpreted as 'I don't know' and '9998' if 'I prefer not to answer.*

CROP REVENUE

Calculation of harvest seasons: Crops can have multiple harvest seasons per year. If the farmer has 2 harvest seasons, please make sure to repeat this question group.

Size of farm dedicated to crop: Use the same unit of measurement as the farmer used to answer the total size of the farm. In case the farmer does not know the size:

1) Ask to give an estimation by probing: was it a quarter of the total size of the land, half of the land, etc.?
2) Check the question on total size of the land (second question under 'Farm') and do the maths.

Example: Farmer says half of the land was dedicated to rice. Under question group 'Farm' it was answered that the total size of the farm is 2 acres. Thus, you fill in that 1 acre of land was dedicated to rice. Note: Farm size dedicated to SDM crop can't exceed total farm size.

Lost crop: For each season, the farmer needs to indicate how much of their main crop has been lost. In case the farmer is not able to answer directly on the loss, probe by asking about drought, floods, birds, disease, and pests.

OTHER CROP INCOME | LIVESTOCK/POULTRY INCOME | OFF-FARM INCOME

Calculating additional income for last 12 months: The questions on non-SDM crops, livestock and off-farm income will be answered in a **time frame of 12 months** instead of (harvest) seasons. For this case, you need to ask additional income from *October 2018* to *September 2019*. In case the farmer does not know exactly, help the farmer to calculate per month/quarter/season and add it up to a total of 12 months. Fill in the numbers from the local currency value.

LABOUR COST

Number of people working on activities: For each activity, make sure you note the answer of number of people involved **excluding** the farmer him/herself.

Number of days and payment per person per activity: You will find that farmers in Vietnam often hire people per day. In that case, ask for the number of days one person works on an activity per day and multiply.

For all the "how many days questions" ensure clarity that we are asking how many days, an avg hired person spent working for you throughout the year for this purpose.

How much did you pay the people you hired per person per month?: Enumerators will input the amount in numbers of local currency

OFF-FARM INCOME

Sources of income that do not relate to crop or livestock: Next to income from farming activities, some farmers will have income generated through other means:

Example remittances: Family member, relative or friends sends money to complement the farmer's income.

Example gift: Farmer receives a (one time) sum of money that he/she does not have to re-pay.

LOANS

Example in-kind loan: Farmer buys seeds on a loan from a seed company. The loan will be deducted when the seed companies buys products from the farmer.

9. Annex 4: R Script

These are the R-scripts used for the data delivery. The versions below are stripped from case specifics, in a normal data delivery *SDM company* is replaced by the actual company name. The first script is a R-markdown script, which means it consists of blocks of code and blocks of text. When run it creates an HTML document containing text and, if applicable, code. The gray highlighted parts are parts of the code, the blue highlighted parts are comments. What remains is text. The comments explain what happens in the code, but are not R-code, so they do not influence the data. The second script is the R script, which is a standardised script aimed to be used for every case which little adjustments needed.

[START SCRIPT]

~~~
---
title: "Data delivery - sdm company"
author: "Akvo - data science"
date: "11/25/2019"
output: html_document
---

```{r setup, include=FALSE}
library(knitr)
knitr::opts_chunk$set(echo = FALSE, message=FALSE, warning=FALSE, cache=FALSE)
knitr::read_chunk('Data delivery bare bones.R')
options(knitr.kable.NA = '-')

sdm_company = 'sdm company'
country = 'Kenya'
sdm_crop = 'maize'

```


```{r readdata}```

```{r repeated groups}```

```{r PPI}```

```{r Numerical descriptives}```
~~~

~~~
```{r Categorical descriptives}```

```{r write}```

# `r sdm_company` {.tabset}

## Data Cleaning Notes

#### Introduction

This document contains details of the `r sdm_company` primary data collection
and data cleaning steps. Please read the PDC process guide (SamePage: Primary
Data Collection) before consulting this document.


##### Number of farmers removed from the set

1. Initially `r nr_participants_raw` farmens were targeted for this data
collection.
2. Of the `r nr_participants_raw`, `r nr_participants_ic` farmers
participated.
3. Of the `r nr_participants_ic`, `r nr_participants_fsize` of the farmers
had less than 10 acres of land, which is the cut off used to indicate small
holders.


##### Sample characteristics:

To get an accurate picture of the SDM farmers we take a random sample of the
farmer group. However, we often run into issues with the farmer list supplied
by the SDM company. Farmers for example might not have an address and we are
dependent on a local contact person to take us to the right farmer. We often
find the farmer is not the farmer that is part of the sample. As the random
component of sampling is very important for the reliability of our findings
we try to register whether the farmers we speak to were part of the original
sample.

The amount of farmers part of the original sample in case of `r sdm_company`
can be found below:
~~~

`r kable(table(Data$farmer_sample))`

## Data Cleaning Steps

#### Introduction

This document contains an overview of the different steps that are taken to clean the FarmFit data for the `r sdm_company`. These steps have been drawn up in cooperation with IDH-FarmFit analysts and will be discussed in the following order:

1. Removing Farmers from the Set
2. Text cleaning
3. Determining and handling outliers
4. Looking at missing values
5. Anonymizing
6. Repeated question groups

#### Removing Farmers from the Set

At the moment farmers are only removed if they refuse to participate with the survey. The only data we have from these farmers is the name, location and sometimes a phone number.

#### Text Cleaning

In order to make the FarmFit data more accessible a few general steps are taken to clean the data.

1. All columns and text values are set to lowercase
2. Flow sets spaces to points; we set them to '_'.
3. Dummy variables get the prefix 'X..OPTION…' by Flow, these are removed from the cleaned data set.
4. A few free text options that have been found often in the data are set to similar text in order to make them comparable. An example is: 'dont know', 'doesn't know', 'I am not sure' are all changed to: 'I don't know'.
5. In case the measurement of crop is supplied by farmers in bags, boxes, crates, etc. In the survey the farmer is asked about the number of kilograms

the applicable measurement contains. In the cleaned data the measurements are set to kilograms, which can be seen in the column heading (_kg).
6. A measurement of an area is generally supplied by farmers in acres, kilometers squared or hectares. In the cleaned data the measurements are set to acres, which can be seen in the column heading (_acre).
7. Some redundant columns are removed, for example columns with Flow details unimportant for the FarmFit analyses.

#### Determining and Handling Outliers

To determine outliers for the numerical questions of the survey, a cut off of three standard deviations from the corresponding mean is set. All values are compared to this cut off. When the value is either higher than three standard deviations above the mean or lower than three standard deviations below the mean, it is set to ‘9997’.

#### Looking at Missing Values

The structure of the FarmFit survey prevents having actual missing values. All multiple-choice questions have the options ‘I don’t know’ and ‘I prefer not to say’ and are mandatory. The numerical questions are also mandatory. Enumerators are instructed to answer them with ‘9999’ in case a farmer doesn’t know the answer, and ‘9998’ when the farmer doesn’t want to give the answer. This way all missing values are defined.
In case of numerical questions, these values are not usable in aggregations and will give incorrect descriptive values. Therefore, all values containing ‘9999’, ‘9998’ and ‘9997’, resulting from outlier
handling, are set to ‘NA’.

#### Anonymizing

In order to anonymize the data farmer names, phone numbers, geolocation (longitude and latitude) and location except from the district is removed from the set.

#### Repeated question groups

When recording the amount of crop produced, sold, lost or used for own consumption, we use 'repeated question groups'. This means we ask the same questions for every season and we save them per season. However, in the data

```
delivery we only present one number for these questions. For the amount
produced, sold, lost and used for their own consumption, we add the values of
every season to get an idea of what happens throughout the year. For the farm
gate price, we take the average of the farm gate price per season.
```

[END SCRIPT]

The standardised R script used for all cases:

[START SCRIPT]

```
##  ---- readdata ----
# Data delivery bare bones

# Change to applicable SDM case
data_filename <- "sdm company_04022020.xlsx"
survey_filename <- "SURVEY_FORM-sdm company.xls"

# Output file names
output_file = "04022020 sdm company ALL"
anonymized_output_file = "04022020 sdm company Anom"

# CROP
sdm_crop = "maize"
country = "Kenya"

# Packages used
library(rmarkdown); library(plyr); library(dplyr); library(tidyr);
library(tidylog); library(readr)
library(stringr); library(data.table) ;library(ggplot2)
library(knitr); library(openxlsx); library(Hmisc)
library(here); library(zoo); library(readxl); library(splitstackshape)

# FUNCTIONS

# DATA TYPE Set factors to integers
factor_to_int <- function(x){
  as.numeric(as.character(x))
}
```

```
# Find and replace outliers
outlier_detection <- function(x){
  ifelse(
      x > (mean(x, na.rm=TRUE) + sd(x, na.rm=TRUE)*3) |
      x < (mean(x, na.rm=TRUE) - sd(x, na.rm=TRUE)*3),
      9997,
      x
  )
}

# count values without NA
count_n <- function(x){sum(!is.na(x))}

## RAW DATA

# Read data from the data folder
Data_raw <- read_excel(
  here::here("data/Flow", data_filename))

# Get version for to make alterations
Data <- copy(Data_raw)

# Get original number of participants
nr_participants_raw <- length(unique(Data$Identifier))

# Read file with survey structure
survey <- read_excel(
  here::here("data/Flow/Survey", survey_filename),
  sheet ="Full Survey", skip=2)

# SURVEY ADJUSTMENTS

survey_questions <- survey %>%
  # Fill excel merged cells from the section
  mutate(Title = na.locf(Title, na.rm = FALSE)) %>%
  # Select right columns
  select("Title", "Variable name", "Text",
      "Question type", "Options",
      "Allow multiple") %>%
  # Trim trailing spaces
  mutate_if(is.character, str_trim) %>%
  # Rename columns
  rename("section" = "Title",
      "variable" = "Variable name",
      "question" = "Text",
      "type" = "Question type",
      "options" = "Options",
      "multiple" = "Allow multiple") %>%
  # All variables to lowercase
  mutate_all(tolower) %>%
  # Filter the empty variables
  filter(!is.na(variable)) %>%
  # Remove crop specific string from survey
  mutate(variable = gsub(paste("_", sdm_crop, sep=""), '', variable))

# DATA ADJUSTMENTS

# Changes to data set:
Data <- Data %>%
  # Remove irrelevant columns
  select(-c(`Display Name`,`Device identifier`,`Instance`,
            `Submission Date`, `Submitter`, `Form version`)) %>%
  select(-contains("--option--")) %>%
  # All column names to lowercase
  rename_all(funs(tolower)) %>%
  # All variables to lowercase
  mutate_all(tolower) %>%
  # Remove to farmers that didn't participate
  filter(informed_consent == "accepted to participate") %>%
  # trailing spaces
  mutate_if(is.character, str_trim) %>%
  # Text changes
  mutate_all(funs(gsub("labour","labor",.))) %>%
  # farm size columns to numeric
  mutate_at(c("f_size", "f_sdm_size"), factor_to_int) %>%
  # Rename columns containing the crop type in the name -> changes when there
are two crops!
  rename_at(vars(contains(sdm_crop)), funs(sub(paste("_", sdm_crop, sep=""),
'', .)))
```

```
# Get new number of participants
nr_participants_ic <- length(unique(Data$identifier))

## Collect missing values for overview -> resulting from "I don't know" or "I
prefer not to say" answers
missing_9999 <- data.frame(
  apply(Data, 2, function(x) sum(x == "i don't know" | x == 9999,
na.rm=TRUE)))
missing_9999$Question <- rownames(missing_9999)
names(missing_9999) <- c("i don't know", "variable")
rownames(missing_9999) <- NULL

missing_9998 <- data.frame(apply(Data, 2, function(x) sum(x == "i prefer not
to say" | x == 9998, na.rm=TRUE)))
missing_9998$Question <- rownames(missing_9998)
names(missing_9998) <- c("i prefer not to say", "variable")
rownames(missing_9998) <- NULL

NA_values <- data.frame(apply(Data, 2, function(x) sum(is.na(x))))
NA_values$Question <- rownames(NA_values)
names(NA_values) <- c("missing due to skip logic", "variable")
rownames(NA_values) <- NULL

missings_raw <- missing_9999 %>%
  left_join(missing_9998) %>%
  left_join(NA_values) %>%
  filter(!variable %like% "*option*") %>%
  filter(!variable %in%
            c("identifier", "duration",
            "sdm_farmer", "informed_consent"))

missings_raw <- missings_raw %>%
  filter(!is.na(missings_raw$variable)) %>%
  left_join(survey_questions, by="variable") %>%
  select("section", "question", "variable",
      "missing due to skip logic", "i don't know",
      "i prefer not to say")

# Variables which contain measurements that should be set to kgs
```

```
columns_measurement_to_kg <- c("f_produced", "f_sold", "f_own_consumption",
"f_lost")

# Change columns data type remove missing and change different measurements
to one
Data <- Data %>%
  # From factor to numeric:
  mutate_at(columns_measurement_to_kg, factor_to_int) %>%

  # All variables that are 9999/"i don't know" or 9998/"i prefer not to say"
are set to NA
  mutate_if(is.numeric, list(~na_if(., 9999))) %>%
  mutate_if(is.numeric, list(~na_if(., 9998))) %>%
  mutate_if(is.character, list(~na_if(., "i don't know"))) %>%
  mutate_if(is.character, list(~na_if(., "i prefer not to say"))) %>%

  # Set measurement of farm size to acres
  mutate(f_size = ifelse(f_unit_land == "hectares", f_size*2.471, f_size))
%>%
  mutate(f_sdm_size = ifelse(f_unit_land == "hectares", f_sdm_size*2.471,
f_sdm_size)) %>%

  # PRODUCED
  unite("f_produced_measurement",
        c(f_produced_other_measurement, f_produced_measurement),
        na.rm = TRUE, remove = FALSE, sep=" ") %>%
  mutate(f_produced_measurement_kilograms =
parse_number(f_produced_measurement, na="NA")) %>%
  mutate(f_produced_measurement_kilograms =
replace(f_produced_measurement_kilograms,
is.na(f_produced_measurement_kilograms), 1)) %>%
  mutate(`f_produced (kilograms)` =
f_produced*f_produced_measurement_kilograms) %>%

  # SOLD
  unite("f_sold_measurement",
        c(f_sold_other_measurement, f_sold_measurement),
        na.rm = TRUE, remove = FALSE, sep=" ") %>%
  mutate(f_sold_measurement_kilograms = parse_number(f_sold_measurement,
na="NA")) %>%
  mutate(f_sold_measurement_kilograms = replace(f_sold_measurement_kilograms,
is.na(f_sold_measurement_kilograms), 1)) %>%
  mutate(`f_sold (kilograms)` = f_sold*f_sold_measurement_kilograms) %>%

  # LOST
  unite("f_lost_measurement",
      c(f_lost_other_measurement, f_lost_measurement),
      na.rm = TRUE, remove = FALSE, sep=" ") %>%
  mutate(f_lost_measurement_kilograms = parse_number(f_lost_measurement,
na="NA")) %>%
  mutate(f_lost_measurement_kilograms = replace(f_lost_measurement_kilograms,
is.na(f_lost_measurement_kilograms), 1)) %>%
  mutate(`f_lost (kilograms)` = f_lost*f_lost_measurement_kilograms) %>%

  # OWN CONSUMPTION
  unite("f_own_consumption_measurement",
      c(f_own_consumption_other_measurement, f_own_consumption_measurement),
      na.rm = TRUE, remove = FALSE, sep=" ") %>%
  mutate(f_own_consumption_measurement_kilograms =
parse_number(f_own_consumption_measurement, na="NA")) %>%
  mutate(f_own_consumption_measurement_kilograms =
replace(f_own_consumption_measurement_kilograms,
is.na(f_own_consumption_measurement_kilograms), 1)) %>%
  mutate(`f_own_consumption (kilograms)` =
f_own_consumption*f_own_consumption_measurement_kilograms) %>%

  # Rename the columns that changed values to indicate changes:
  rename("f_size (acre)" = "f_size",
      "f_sdm_size (acre)" = "f_sdm_size")

## ---- repeated groups ----

# remove lines without variable name
survey_questions <- survey_questions[!is.na(survey_questions$variable),]

# Make same changes in survey question data
for(kg in columns_measurement_to_kg){
  survey_questions$variable <- str_replace(
      survey_questions$variable, paste0(kg, "\\b"), paste0(kg, ' (kilograms)')
  )
}

for(acre in c("f_size", "f_sdm_size")){
  survey_questions$variable <- str_replace(
      survey_questions$variable, paste0(acre, "\\b"), paste0(acre, ' (acre)')
  )
}

# Collect the different variables, numerical questions:   --- ERROR HERE
numerical_columns <- as.vector(
  survey_questions[survey_questions$type=="number", "variable"][[1]])
Data <- Data %>%
  mutate_at(vars(numerical_columns), funs(as.numeric))

# Replace values that are more than 3 sd from the mean with 9997 - AND ERROR
HERE
Data <- Data %>%
  mutate_at(vars(numerical_columns), funs(outlier_detection))

# Collect outliers for overview
outliers <- data.frame(apply(Data, 2, function(x) sum(x == 9997,
na.rm=TRUE)))
outliers$Question <- rownames(outliers)
names(outliers) <- c("outliers", "variable")
rownames(outliers) <- NULL
outliers <- outliers %>% left_join(survey_questions, by="variable") %>%
  select("section", "question", "variable", "outliers") %>%
  arrange(desc(outliers))

# Replace 9997 with NA after outliers are registered
Data <- Data %>%
  mutate_if(is.numeric, list(~na_if(., 9997)))

# Split the repeated question group from the data
revenue_cols <- as.vector(
  survey_questions[
      !is.na(survey_questions$variable) &
      survey_questions$section %like% 'crop revenue', "variable"
      ][[1]]
)
```

```
# Split the revenue columns from the farm and household data  - AND ERROR
HERE
revenue <- Data[,c("identifier", "repeat no", revenue_cols)]
revenue <- revenue %>% mutate_if(is.factor, as.character)

# Select the farm and household data and remove missing rows due
# to repeated questions (harvests in this case)
farm <- Data[, !names(Data) %in% revenue_cols]
farm <- farm[farm$`repeat no` ==1, !names(farm) %like% 'repeat no']

revenue_num <- revenue %>%
  dplyr::select(-contains("price")) %>%
  group_by(identifier) %>%
  select_if(is.numeric) %>%
  summarise_each(funs(sum))

revenue_price <- revenue %>%
  group_by(identifier) %>%
  dplyr::select(contains("price")) %>%
  summarise_each(funs(mean))

# Combine the strings by making them into a list
revenue_char <- revenue %>%
  group_by(identifier) %>%
  select_if(is.character) %>%
  summarise_each(funs(first))

# combine the grouped dataframe back into one
revenue_summed <- revenue_num %>% full_join(revenue_price)
revenue_summed <- revenue_summed %>% full_join(revenue_char)

# Combine the revenue with the farm data back into one dataframe
Data <- farm %>% left_join(revenue_summed, by= "identifier")

# Remove farmer with too large farms
Data <- Data %>%
  filter(`f_size (acre)` < 10)
nr_participants_fsize <- length(unique(Data$identifier))
```

```
## ---- PPI ----

# THIS NEEDS TO BE CHECKED FOR EVERY DELIVERY

# Load the PPI scorecard (excel file constructed from the official scorecard)
scorecard <- read.xlsx(here("data/PPI","PPI_scorecard.xlsx"), sheet=country)

# Import the look up table for PPI
PPI_lookup <- read.xlsx(here("data/PPI","PPI_look_up_table.xlsx"),
sheet=country)

# Select data
PPI_data <- Data %>%
  select(c("identifier", "hh_gender", unique(scorecard$variable))) %>%
  gather(ppi_question, ppi_answer,
unique(scorecard$variable)[1]:unique(scorecard$variable)[10]) %>%
  filter(!is.na(ppi_answer)) %>%
  mutate_if(is.character, tolower)

# Replace the numeric answers with answers corresponding to the PPI scorecard
PPI_data <- PPI_data  %>%
  mutate(ppi_answer = replace(ppi_answer, ppi_answer == 1, "one")) %>%
  mutate(ppi_answer = replace(ppi_answer, ppi_answer == 2, "two")) %>%
  mutate(ppi_answer = replace(ppi_answer, ppi_answer == 3, "three")) %>%
  mutate(ppi_answer = replace(ppi_answer, ppi_answer == 4, "four")) %>%
  mutate(ppi_answer = replace(ppi_answer, ppi_answer > 4 & ppi_answer < 7,
"five or\tsix")) %>%
  mutate(ppi_answer = replace(ppi_answer, ppi_answer == 7, "seven")) %>%
  mutate(ppi_answer = replace(ppi_answer, ppi_answer == 8, "eight")) %>%
  mutate(ppi_answer = replace(ppi_answer, ppi_answer == "primary school",
"primary")) %>%
  mutate(ppi_answer = replace(ppi_answer, ppi_answer == "secondary school",
"secondary or post-primary, vocational")) %>%
  mutate(ppi_answer = replace(ppi_answer, ppi_answer == "college", "college
level or higher")) %>%
  mutate(ppi_answer = replace(ppi_answer, ppi_answer == "they did not attend
school", "pre-primary, none, or other"))

# Replace the last value (9 or higher)
PPI_data$ppi_answer <- gsub("^[0-9]+$","nine or more", PPI_data$ppi_answer)
```

```r
# Change data type
PPI_data$ppi_question <- as.character(PPI_data$ppi_question)

# Set all values to lowercase (to make comparison easier)
scorecard <- data.frame(lapply(scorecard, function(x) tolower(x)))

# Change data type
scorecard$answer <- as.character(scorecard$answer)
scorecard$variable <- as.character(scorecard$variable)

# Trim possible trailing spaces from file
scorecard <- scorecard %>% mutate_if(is.character, str_trim)

# join the files to get the PPI scores per farmer
PPI_score <- PPI_data %>%
  left_join(scorecard, by=c("ppi_question" = "variable",
"ppi_answer"="answer"))

# reshape the data to combine the score to a total
PPI_score <- PPI_score[,c("identifier", "hh_gender","ppi_question", "score")]
%>%
  filter(!is.na(score)) %>%
  spread(ppi_question, score)

# Change data type
PPI_score[, sapply(PPI_score, class) == 'factor'] <- as.data.frame(
  lapply(
    PPI_score[, sapply(PPI_score, class) == 'factor'], as.numeric
  ), stringsAsFactors = FALSE
)

# Get a score total
PPI_score$total <- apply(PPI_score[,3:length(PPI_score)], 1, sum)

# Merge the probability to the PPI scores
PPI_score <- PPI_score %>%
  mutate(dummy=TRUE) %>%
  left_join(PPI_lookup %>% mutate(dummy=TRUE)) %>%
  filter(total == `PPI.Score`) %>%
  select(-dummy)

# Get descriptive values
ppi_var <- PPI_score %>%
  group_by(hh_gender) %>%
  summarise(`mean` = mean(`Likelihood.(%)`),
            `sd` = sd(`Likelihood.(%)`),
            `min` = min(`Likelihood.(%)`),
            `max` = max(`Likelihood.(%)`))  %>%
  mutate(section = "PPI") %>%
  mutate(variable = "Likelihood (%)") %>%
  rename(gender = hh_gender)

# REMOVE PRIVACY COLUMNS

# Select variables with private information
private_info <- survey_questions %>%
  filter(section %in% c("personal information & monitoring", "location - confidential")) %>%
  filter(variable != "geolocation") %>%
  pull(variable)

# Select Geolocation separate
geolocation <- names(Data)[names(Data) %like% "geolocation"]

# Remove columns from set
Data <- dplyr::select(Data, select = -c(private_info, geolocation))

## ---- Numerical descriptives ----
numerical_means <- Data %>%
  group_by(hh_gender) %>%
  select_if(is.numeric) %>%
  summarise_each(funs(round(mean(., na.rm = TRUE),2)))

numerical_sds <- Data %>%
  group_by(hh_gender) %>%
  select_if(is.numeric) %>%
  summarise_each(funs(round(sd(., na.rm = TRUE),2)))

numerical_minus <- Data %>%
  group_by(hh_gender) %>%
  select_if(is.numeric) %>%
  summarise_each(funs(min(., na.rm = TRUE)))

numerical_max <- Data %>%
  group_by(hh_gender) %>%
  select_if(is.numeric) %>%
  summarise_each(funs(round(max(., na.rm = TRUE),2)))

numerical_freq <- Data %>%
  group_by(hh_gender) %>%
  select_if(is.numeric) %>%
  summarise_each(funs(count_n))

numerical_descriptives <- melt(numerical_freq, value.name="n") %>%
  left_join(melt(numerical_means, value.name="mean")) %>%
  left_join(melt(numerical_sds, value.name="sd")) %>%
  left_join(melt(numerical_minus, value.name="min")) %>%
  left_join(melt(numerical_max, value.name="max")) %>%
  left_join(survey_questions) %>%
  select(c("section", "question", "variable","hh_gender","n","mean","sd",
"min","max")) %>%
  rename("gender" = "hh_gender")

numerical_descriptives <- numerical_descriptives %>%
  full_join(ppi_var)

## ---- Categorical descriptives ----

# Gender
gender_var <- melt(table(Data$hh_gender))
gender_var$percentage <-
round((gender_var$value/nr_participants_fsize)*100,1)
names(gender_var) <- c("gender", "count", "percentage")

## Education (Male vs Female, primary and secondary)
education_var <- melt(table(Data[,c("hh_gender", "hh_education_farmer")]))
names(education_var) <- c("gender", "answer", "frequency")
education_var <- education_var %>%
  left_join(gender_var[,c("gender","count")])
```

```
education_var$percentage <-
round((education_var$frequency/education_var$count)*100,1)
education_var$variable <- "hh_education_farmer"
education_var <- education_var[,c("gender", "variable", "answer",
"frequency", "percentage")]

## Household head
household_head_var <- melt(table(Data[,c("hh_gender", "hh_head_gender")]))
names(household_head_var) <- c("gender", "answer", "frequency")
household_head_var <- household_head_var %>%
  left_join(gender_var[,c("gender","count")])

household_head_var$percentage <-
round((household_head_var$frequency/household_head_var$count)*100,1)
household_head_var$variable <- "hh_head_gender"
household_head_var <- household_head_var[,c("gender", "variable", "answer",
"frequency", "percentage")]

categorical_descriptives <- household_head_var %>% full_join(education_var)

## FOOD SECURITY
foodshortage_var <- melt(
  table(Data[,c("hh_gender", "fs_shortage")]),
  value.name="frequency",
  varnames = c("gender","answer"))
foodshortage_var$variable <- "fs_shortage"

foodshortage_var <- foodshortage_var %>%
  left_join(gender_var[,c("gender","count")])
foodshortage_var$percentage <-
round((foodshortage_var$frequency/foodshortage_var$count)*100,1)
foodshortage_var <- foodshortage_var %>% dplyr::select(-count)

categorical_descriptives <- education_var %>% full_join(foodshortage_var)

## MOBILE MONEY
Data[,c("hh_mobile", "hh_mobile_functionality", "hh_mobile_money",
"hh_bank_account")] <-
```

```
  apply(Data[,c("hh_mobile", "hh_mobile_functionality", "hh_mobile_money",
"hh_bank_account")],
      2, as.character)
mobile_var <- melt(table(Data[,c("hh_gender", "hh_mobile")]),
                   value.name="frequency",
                  varnames = c("gender","answer"))
money_var <- melt(table(Data[,c("hh_gender", "hh_mobile_money")]),
                   value.name="frequency",
                  varnames = c("gender","answer"))
bank_var <- melt(table(Data[,c("hh_gender", "hh_bank_account")]),
                   value.name="frequency",
                  varnames = c("gender","answer"))

mobile_var$variable <- "hh_mobile"
money_var$variable <- "hh_mobile_money"
bank_var$variable <- "hh_bank_account"

mobile_money_var <- rbind(mobile_var, money_var, bank_var)

mobile_money_var <- mobile_money_var %>%
  left_join(gender_var[,c("gender", "count")])
mobile_money_var$percentage <- round(
  (mobile_money_var$frequency/mobile_money_var$count)*100,1)
mobile_money_var <- mobile_money_var %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
full_join(mobile_money_var)

## Mobile functionality
mobile_function_var <- Data %>%
  cSplit("hh_mobile_functionality","|") %>%
  dplyr::select(hh_gender, starts_with("hh_mobile_functionality")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

mobile_function_var <- melt(
  table(mobile_function_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

mobile_function_var <- mobile_function_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "hh_mobile_functionality") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(mobile_function_var)

## LOANS
loan_var <- Data %>%
  cSplit("hh_loan_presence","|") %>%
  dplyr::select(hh_gender, starts_with("hh_loan_presence")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

loan_var <- melt(
  table(loan_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

loan_var <- loan_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "hh_loan_presence") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>% full_join(loan_var)

## Loan Source
loan_source_var <- Data %>%
  cSplit("hh_loan_source","|") %>%
  dplyr::select(hh_gender, starts_with("hh_loan_source")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

loan_source_var <- melt(
  table(loan_source_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

loan_source_var <- loan_source_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "hh_loan_source") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
full_join(loan_source_var)

## LOAN USE --
loan_use_var <- Data %>%
  cSplit("hh_loan_use","|") %>%
  dplyr::select(hh_gender, starts_with("hh_loan_use")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

loan_use_var <- melt(
  table(loan_use_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

loan_use_var <- loan_use_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "hh_loan_use") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(loan_use_var)
```

```
## CASHFLOW presence
Data$cf_shortage <- as.character(Data$cf_shortage)

cashflow_var <- melt(table(Data[,c("hh_gender","cf_shortage")]))
cashflow_var <- cashflow_var %>%
  rename(gender=hh_gender, answer=cf_shortage, frequency=value) %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate(percentage = round((frequency/count)*100,1)) %>%
  mutate(variable = "cf_shortage") %>%
  select(-count)

categorical_descriptives <- categorical_descriptives %>%
  full_join(cashflow_var)

##  CASHFLOW MONTHS
cashflow_months <- Data %>%
  cSplit("cf_shortage_months","|") %>%
  dplyr::select(hh_gender, starts_with("cf_shortage_months")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

cashflow_months <- melt(
  table(cashflow_months[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

cashflow_months <- cashflow_months %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "cf_shortage_months") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage)) %>%
  filter(answer != "i don't know")

categorical_descriptives <- categorical_descriptives %>%
  full_join(cashflow_months)

# MAIN CROP -> FIRST
```

```
first_crop_var <- Data %>%
  unite("f_first_crop", c(f_first_crop, f_first_crop_other),na.rm = TRUE,
remove = FALSE, sep=" ") %>%
  select(hh_gender, f_first_crop) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

first_crop_var <- melt(
  table(first_crop_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

first_crop_var <- first_crop_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "f_first_crop") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(first_crop_var)

# MAIN CROP -> SECOND
second_crop_var <- Data %>%
  unite("f_second_crop", c(f_second_crop, f_second_crop_other),na.rm = TRUE,
remove = FALSE, sep=" ") %>%
  select(hh_gender, f_second_crop) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

second_crop_var <- melt(
  table(second_crop_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

second_crop_var <- second_crop_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "f_second_crop") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage)) %>%
  mutate(answer = replace(answer, answer=="", "no second crop"))

categorical_descriptives <- categorical_descriptives %>%
  full_join(second_crop_var)

# MAIN CROP -> THIRD
third_crop_var <- Data %>%
  unite("f_third_crop", c(f_third_crop, `f_third_crop--other--`),na.rm =
TRUE, remove = FALSE, sep=" ") %>%
  select(hh_gender, f_third_crop) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

third_crop_var <- melt(
  table(third_crop_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

third_crop_var <- third_crop_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "f_third_crop") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage)) %>%
  mutate(answer = replace(answer, answer=="", "no third crop"))

categorical_descriptives <- categorical_descriptives %>%
  full_join(third_crop_var)

## INPUTS
inputs_var <- Data %>%
  rename(f_input_types = f_inputs) %>%
  cSplit("f_input_types","|") %>%
  dplyr::select(hh_gender, starts_with("f_input_types")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")
```

```
inputs_var <- melt(
  table(inputs_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

inputs_var <- inputs_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "f_inputs") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(inputs_var)

## INPUTS Challenges
input_challenges_var <- Data %>%
  cSplit("f_inputs_challenges_type","|") %>%
  dplyr::select(hh_gender, starts_with("f_inputs_challenges_type")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

input_challenges_var <- melt(
  table(input_challenges_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

input_challenges_var <- input_challenges_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "f_input_challenges") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(input_challenges_var)
```

```
## EQUIPMENT

equipment_var <- Data %>%
  rename(f_equipment_type = f_equipment) %>%
  unite("f_equipment_type", c(f_equipment_type, `f_equipment_other`), na.rm =
TRUE, remove = FALSE, sep="|") %>%
  cSplit("f_equipment_type","|") %>%
  dplyr::select(hh_gender, starts_with("f_equipment_type")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

equipment_var <- melt(
  table(equipment_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

equipment_var <- equipment_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "f_equipment") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(equipment_var)

## CLIMATE ISSUES
climate_var <- Data %>%
  cSplit("cl_extreme_weather","|") %>%
  dplyr::select(hh_gender, starts_with("cl_extreme_weather")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

climate_var <- melt(
  table(climate_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

climate_var <- climate_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "cl_extreme_weather") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(climate_var)

## CLIMATE MITIGATION
mitigation_var <- Data %>%
  unite("cl_coping_mechanisms", c(cl_coping_mechanisms,
`cl_coping_mechanisms--other--`), na.rm = TRUE, remove = FALSE, sep="|") %>%
  cSplit("cl_coping_mechanisms","|") %>%
  dplyr::select(hh_gender, starts_with("cl_coping_mechanisms")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

mitigation_var <- melt(
  table(mitigation_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

mitigation_var <- mitigation_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "cl_coping_mechanisms") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(mitigation_var)

## GENDER

# Preliminaries

# Informed consent info
```

```
g_informed_consent <- melt(
  table(Data[,c("hh_gender", "g_informed_consent")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

g_informed_consent <- g_informed_consent %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "g_informed_consent") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(g_informed_consent)

# Education of female primary dec. mak.
g_education <- melt(
  table(Data[,c("hh_gender", "g_education")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

g_education <- g_education %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "g_education") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(g_education)

# Decision making in HH
g_reproductive_var <- Data %>%
  cSplit("g_reprod_activities","|") %>%
  dplyr::select(hh_gender, starts_with("g_reprod_activities")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")
```

```
g_reproductive_var <- melt(
  table(g_reproductive_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

g_reproductive_var <- g_reproductive_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "g_reprod_activities") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(g_reproductive_var)

## DECISION MAKING HH  --> PRODUCTIVE
g_productive_var <- Data %>%
  cSplit("g_prod_activities","|") %>%
  dplyr::select(hh_gender, starts_with("g_prod_activities")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

g_productive_var <- melt(
  table(g_productive_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

g_productive_var <- g_productive_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "g_prod_activities") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(g_productive_var)

## DECISION MAKING ACTIVITIES HH FARM -- Reproductive
```

```r
g_reproductive_input <- melt(
  table(Data[,c("hh_gender", "g_reprod_input_decision")]),
  value.name="frequency",
  varnames=c("gender","answer"))
g_reproductive_input$variable <- "g_reprod_input_decision"

g_reproductive_input <- g_reproductive_input %>%
  left_join(gender_var[,c("gender","count")])
g_reproductive_input$percentage <- round(
  (g_reproductive_input$frequency/g_reproductive_input$count)*100,1)
g_reproductive_input <- g_reproductive_input %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(g_reproductive_input)

# Decision making reproductive
g_reproductive_decision <- Data %>%
  dplyr::select(hh_gender, "g_reprod_resp_decision") %>%
  cSplit("g_reprod_resp_decision","|") %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

g_reproductive_decision <- melt(
  table(g_reproductive_decision[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

g_reproductive_decision <- g_reproductive_decision %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "g_reprod_resp_decision") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(g_reproductive_decision)
```

```
## DECISION MAKING ACTIVITIES HH FARM

g_productive_input <- Data %>%
  dplyr::select(hh_gender,
                starts_with("g_prod_input"),
                -contains("household")) %>%
  gather(position, answer, -hh_gender) %>%
  mutate(position = gsub("g_prod_input_", "input into ", position)) %>%
  mutate(position = gsub('_', " ", position))

g_productive_input <- melt(
  table(g_productive_input),
  value.name="frequency",
  varnames=c("gender","variable","answer"))

g_productive_input <- g_productive_input %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(g_productive_input)

# Decision making productive
g_productive_decision <- Data %>%
  dplyr::select(hh_gender,
                starts_with("g_prod_decision")) %>%
  cSplit(2:8,"|") %>%
  gather(position, answer, -hh_gender) %>%
  mutate(position = gsub("g_prod_decision_", "decision making ", position))
%>%
  mutate(position = gsub('_', " ", position)) %>%
  mutate(position = gsub('[0-9]', "", position))

g_productive_decision <- melt(
  table(g_productive_decision),
  value.name="frequency",
  varnames=c("gender", "variable","answer"))
```

```
g_productive_decision <- g_productive_decision %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage)) %>%
  filter(frequency != 0)

categorical_descriptives <- categorical_descriptives %>%
  full_join(g_productive_decision)

## CUSTOMER SATISFACTION
customer_var <- Data %>%
  cSplit("cs_services","|") %>%
  dplyr::select(hh_gender, starts_with("cs_services")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

customer_var <- melt(
  table(customer_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

customer_var <- customer_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "cs_services") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(customer_var)

## RECOMMENDATIONS
recommendations_var <- Data %>%
  cSplit("cs_recommendation","|") %>%
  dplyr::select(hh_gender, starts_with("cs_recommendation")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

recommendations_var <- melt(
  table(recommendations_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

recommendations_var <- recommendations_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "cs_recommendation") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(recommendations_var)

## LIVESTOCK
livestock_var <- Data %>%
  cSplit("f_other_crop_livestock","|") %>%
  dplyr::select(hh_gender, starts_with("f_other_crop_livestock")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

livestock_var <- melt(
  table(livestock_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

livestock_var <- livestock_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "f_other_crop_livestock") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(livestock_var)
```

```
## OFF -FARM INCOME
off_farm_var <- Data %>%
  cSplit("f_offfarm_labour","|") %>%
  dplyr::select(hh_gender, starts_with("f_offfarm_labour")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

off_farm_var <- melt(
  table(off_farm_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

off_farm_var <- off_farm_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "f_offfarm_labour") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(off_farm_var)

# LAND OWNERSHIP
land_ownership_var <- Data %>%
  cSplit("f_ownership","|") %>%
  dplyr::select(hh_gender, starts_with("f_ownership")) %>%
  gather(position, answer, -hh_gender) %>%
  select("hh_gender", "answer")

land_ownership_var <- melt(
  table(land_ownership_var[,c("hh_gender", "answer")]),
  value.name="frequency",
  varnames=c("gender", "answer"))

land_ownership_var <- land_ownership_var %>%
  left_join(gender_var[,c("gender","count")]) %>%
  mutate_at(vars(c("count", "frequency")), as.numeric) %>%
  mutate(percentage =  round((frequency/count)*100,1)) %>%
  mutate(variable = "f_ownership") %>%
  dplyr::select(-count) %>%
  arrange(gender, desc(percentage))

categorical_descriptives <- categorical_descriptives %>%
  full_join(land_ownership_var)

# Add n to categorical descriptives
cat_desc_n <-categorical_descriptives %>%
  group_by(gender, variable) %>%
  summarise(n = sum(frequency))

categorical_descriptives <- categorical_descriptives %>%
  left_join(cat_desc_n) %>%
  left_join(survey_questions) %>%
  select("section", "question", "variable", "gender",
      "n", "answer","frequency", "percentage")

## ---- write ----

## WRITE data to excel
sets <- list(
  "Code book" = survey_questions,
  "Cleaned Data" = Data,
  "Raw Data" = Data_raw,
  "Missing values in the data" = missings_raw,
  "Outliers" = outliers,
  "Numerical descriptives" = numerical_descriptives,
  "Categorical descriptives" = categorical_descriptives
)

## WRITE data to excel
sets_anonymized <- list(
  "Code book" = survey_questions,
  "Cleaned Data" = Data,
  "Missing values in the data" = missings_raw,
  "Outliers" = outliers,
  "Numerical descriptives" = numerical_descriptives,
  "Categorical descriptives" = categorical_descriptives)
```

```
write.xlsx(sets_anonymized, file = here::here("data/output",
paste0(anonymized_output_file, ".xlsx")))
write.xlsx(sets, file = here::here("data/output", paste0(output_file,
".xlsx")))
```

[END SCRIPT]