The SurveyAMC Project

Manual

December 28, 2020

Claudia Saalbach *

https://www.survey.codes/surveyamc

https://gitlab.com/jojo_boulix/auto-multiple-choice

*Claudia Saalbach is a research associate at the Chair of Methods of Empirical Social Research at the University of Potsdam, Germany.

Abstract

The SurveyAMC project is targeted to create machine-readable questionnaires for self-administered paper surveys in high typesetting quality. SurveyAMC is available as an option of the Auto-Multiple-Choice LaTeX package. It provides new commands for generating survey questions, formatting the answer options and the general page layout, adding completion and filter instructions, and importing the questionnaire's content from a metafile. Since SurveyAMC is part of the Auto-Multiple-Choice software[1], the scanned answer sheets can be automatically processed into a dataset and therefore analyzed with standard statistical software.

Contents

[1] https://gitlab.com/jojo_boulix/auto-multiple-choice

List of Figures

1 Introduction

Despite the general availability of web surveys, self-administered paper-and-pencil surveys are still a major mode of survey research. Self-administered paper-and-pencil questionnaires are, for example, used for populations that are difficult to reach with web surveys, or in combination with other modes in mixed mode approaches (de Leeuw, 2018; Couper, 2011; Don A. Dillman, Smyth, & Christian, 2009). Self-administered paper-and-pencil surveys are also often preferred to web surveys due to the many documented quality problems of the latter, including sampling problems (Bethlehem, 2015), very low response rates (Don A Dillman, Reips, & Matzat, 2010), and sloppy answering behavior (Zhang & Conrad, 2014).

The web mode merges the data collection and the data entry into one single step performed by the respondents. This leads to an immense decrease in survey costs, both in terms of time and money. However, this advantage of the web mode can be reduced to some extent by machine-readable paper questionnaires. Besides, SurveyAMC is a suitable tool for obtaining the best possible data quality. Compared with manual data entry methods, machine readability reduces the risk of processing errors. Further, SurveyAMC allows implementing the scientific principles of visual questionnaire design (Don A. Dillman, Smyth, & Christian, 2014), reducing measurement error if correctly applied. Thus, the SurveyAMC package addresses the demand for a combination of comprehensive machine-readability and high layout quality. More specifically, SurveyAMC offers a:

. . . very high typesetting quality of the paper questionnaires (see Figure 2.1 for an example) by providing a comprehensive toolbox. It should be thereby noted that the layout of a questionnaire is not just an aesthetic exercise, but has direct effects on the quality of the collected data (e.g., Christian & Dillman, 2004). Due to the quality of the questionnaires created with SurveyAMC, it might also be used for creating questionnaires for interviewer based survey modes since the questionnaires' usability affects interviewer behavior in a similar way to respondents'(Geisen & Romano Bergstrom, 2017).

... extensive machine-readability. The completed questionnaires can be scanned and processed by the freely available software Auto-Multiple-Choice[1]. The processed data is written into a fully structured file in a comma-separated format (a so-called CSV file).

... the use of structured metafiles. SurveyAMC provides commands for importing questionnaire contents (i.e. the text for questions and answers, variable names and values) from an external file in a spreadsheet format. One advantage of this is that the questionnaire contents can be created by survey experts without LaTeX skills using ordinary software such as MS Excel, or the freeware alternative LibreOffice or Open Office Calc. However, the key idea is to create questionnaires for different modes from one source. As it stands, an interface for creating web mode questionnaires for LimeSurvey[2] exists.

... extendable linkages. The use of structured metafiles allows creating various secondary products of the survey. This includes fully labeled data files for statistical packages such as Stata, as well as certain highly standardized survey products such as data dictionaries or table volumes. Various such extensions exist and are available via the survey.codes project[3].

... one-click processing. All survey products (paper-questionnaire, data dictionary, data label file) are created from one source file just by one click. The additional open source software ClickAMC bundles default files for generating these survey products. ClickAMC is freely accessible via a web application and GitLab[4]

This document contains a short Tutorial (Chapter 2), as well as a detailed Manual (Chapter 3). The Tutorial provides the most vital information to program the first questionnaire. The Manual explains the LaTeX commands in greater detail. It intends to give you a deeper understanding of the SurveyAMC package option so that you will be able to use all tools in its complete flexibility. As you will notice, the SurveyAMC LaTeX commands are not shortened to its' minimum. Even if it is a bit more to type, the more extended commands are deliberately designed that way. The SurveyAMC LaTeX commands give you the flexibility to adjust many parameters of the questions' layout. For example, the width of the answer options, their vertical or horizontal spacing, or their alignment, to name

[1] For more information see https://www.auto-multiple-choice.net and https://gitlab.com/jojo_boulix/auto-multiple-choice
[2] For more information see https://www.limesurvey.org
[3] For more information see https://survey.codes
[4] For more information see https://survey.codes/click and https://gitlab.com/CSaalbach/clickamc-project

a few. Since questionnaire contents can differ significantly, this flexibility is necessary to achieve the best possible typesetting quality for every thinkable paper questionnaire. The manual is structured as follows:

- It starts with a detailed description of the layout options for each of the three most common question types. Single-Choice Questions: Question text with two or more answer options. Respondents are allowed to tick only one answer (Section 3.2). Multiple-Choice Questions: Question text with two or more answer options. Respondents are allowed to tick more than one answer (see Section 3.3). Matrix Questions: A general lead-in question followed by several lines that define variants of the lead-in question (see Section 3.4).[5] Respondents are allowed to tick one answer in each line.

- Introducing different question types is followed by a bunch o examples that show how to add visual design elements to those questions. Graphical Add-ons (Section 3.5) may be useful if a question contains completion or filter instructions. Further, closed-ended questions may include an answer option with an open text field for respondents to write in an alternative answer.[6] Further, graphical elements can be used to connect the answer boxes of a matrix question with horizontal lines to support the respondents in their task to search for the right answer not column by column but row by row.

- The Overall Questionnaire Layout is one of the first things respondents perceive. This first impression may influence the respondents' decision on whether to fill in the questionnaire or not. Therefore, the overall layout of a questionnaire is essential, and survey methodologists like Dillman and colleagues (2014) have made a significant contribution to this field by developing guidelines for designing appealing questionnaire layouts. You will learn what options are available for designing an appealing layout in Section General Settings (3.1).

- How to use Structured Metafiles for programming questionnaires is explained in Section 3.6. You will learn abou the metafiles' structure and how to import the questionnaire's metdata to LaTeX. This procedure may be useful, since besides the paper questionnaire also the web questionnaire, or the data documenting files can be created from one source. This section will also find some information about importing the metafile to web questionnaire software and the paper-questionnaire generator ClickAMC.

[5] It should be noted that matrix questions are frequently used but often discouraged by survey methodologists.

[6] Note, however, that Auto-Mulitple-Choice is not able to process handwritten open answers.

- Since the functionality of the SurveyAMC LaTeX package relies on the third party software Auto-Mulitple-Choice, some information about this software is given as well. Section 3.7 briefly describes how you can Generate the Answer Dataset.

2 Tutorial

Karl Marx: Workers' Inquiry

In 1880 Karl Marx intended to investigate the living and working conditions of the french working class through a survey of workers. It is considered one of the first works of empirical social research. Marx developed a questionnaire with over a hundred questions to which he seeks respondance by publishing it in the magazine "Revenue Socialiste" and by distributing a large number of copies throughout France (Weiss, 1936).

Since he designed all questions as open-ended questions, respondents had to formulate their answers with their own words and write them down. As we know today, the results of this survey were never published, maybe due to a horrifying small response rate (ibid.) or due to the tremendous amount of work to code and analyze the handwritten answers. But what if Marx could have used SurveyAMC to create a standardized machine-readable questionnaire?

The questionnaire follows on the next page.

+1/1/60+

Question 1: Does the shop in which you work belong to a capitalist or to a limited company?

capitalist ☐

limited company ☐

don't know ☐

Question 2: What is the youngest age at which children are taken on?

< 13 years	13 years	14 years	15 years	16 years	> 16 years	don't know
☐	☐	☐	☐	☐	☐	☐

Question 3: Is the shop in a town, or in a village?

town ☐ ⟶ go on with question 5

village ☐ ⟶ go on with question 4

Question 4: If your shop is in the country, is there sufficient work in the factory for your existence, or are you obliged to combine it with agricultural labor?

sufficient work in the factory ☐

combine it with agricultural labor ☐

Question 5: Are safety measures to prevent accidents applied to the engine, transmission and machinery? Please tick all that apply.

engine ☐

transmission ☐

machinery ☐

+1/2/59+

Question 6: How satisfied are you with the hygienic conditions in the workshop regarding the following aspects?

	very satisfied	somewhat satisfied	somewhat dissatisfied	very dissatisfied	don't know
size of the rooms space	☐	☐	☐	☐	☐
ventilation	☐	☐	☐	☐	☐
temperature	☐	☐	☐	☐	☐
plastering	☐	☐	☐	☐	☐
lavatories	☐	☐	☐	☐	☐
general cleanliness	☐	☐	☐	☐	☐
noise of machinery	☐	☐	☐	☐	☐
metallic dust	☐	☐	☐	☐	☐
dampness	☐	☐	☐	☐	☐

Question 7: Mention the accidents which have taken place to your personal knowledge.

Question 8: State the number of holidays in the course of a year.

☐☐

Figure 2.1: Karl Marx Wokers' Inquiry

2.1 Getting Started

Installation

Before Marx can get started with his questionnaire, he has to make some preparations. Since he uses a Linux operating system on his computer, he can install the Auto-Multiple-Choice software. To do this, he visits the GitLab web page (https://gitlab.com/jojo_boulix/auto-multiple-choice/), and studies the README[1]. Then Marx downloads the program and installs it on his computer. Now he can use the Auto-Multiple-Choice LaTeX package with the new option for survey programming. Marx opens a new TEX file via a LaTeX editor and loads the package within the document's preamble. In order to be able to use the commands for programming survey questions, he adds the option [survey] when loading {automultiplechoice}:

Figure 2.2: Package Implementation & Questionnaire Environment

```
\documentclass[11 pt, a4paper]{report}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[english]{babel}
\usepackage[survey]{automultiplechoice}

\begin{document}
    \begin{Questionnaires}{1}
        my first questionnaire
    \end{Questionnaires}
\end{document}
```

Marx intends to work closely with Engels in developing the questionnaire. However, Engels uses a proprietary operating system on which Auto-Multiple-Choice cannot be installed. However, since Marx only wants to share his LaTeX code with Engels, he finds a solution. He recommends his friend to use the style file (Auto-Multiple-Choice LaTeX package) outside of the software. For this, Engels only has to copy the sty file into his tex tree or put it into the same folder where the TEX document with the questionnaire code is located. Engels can now also work on the questionnaire, although he does not use a Linux operating system. However, only Marx can process the answer sheets into a dataset since only he has installed the complete software.

[1] https://gitlab.com/jojo_boulix/auto-multiple-choice/-/blob/master/README

Questionnaire Environment

Having finished the preamble, Marx creates a Questionnaires environment in which he will later program the questions. At this point, he also indicates the number of individualized questionnaires he would like to create by typing the appropriate number within the first curly brackets[2] of the Questionnaires environment. Since he plans to create a draft version first, he types a 1 into the first argument.

Marx wants to know whether he has done everything right so far, so he compiles the TEX file. The result looks good: The header of the PDF document contains a barcode. This barcode is necessary for the Auto-Multiple-Choice software to identify the answer sheets and process them into a dataset.

Default Messages

Marx also realizes the message "draft" printed across the entire page. He figures that he could turn off this watermark by putting the command \AMCtext{draft}{} in the preamble, but he decides to keep the message to prevent him from using the draft version for actual data collection. Additionally, there is a footnote on the questionnaire page. Marx replaces the default message with the questionnaires' title by typing \AMCtext{message}{Worker's Inquiry} in the preamble:

Figure 2.3: Questionnaire Footnote

```
\documentclass[11 pt, a4paper]{report}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[english]{babel}
\usepackage[survey]{automultiplechoice}
\AMCtext{message}{Worker's Inquiry}

\begin{document}
  \begin{Questionnaires}{1}
    my first questionnaire
  \end{Questionnaires}
\end{document}
```

[2]Note that further in this document, curly brackets that follow a command or an environment are called argument, and square brackets are called option.

2.2 The First Question

Question Text

Marx rolls up his sleeves and starts programming the first question (Question 1, Figure 2.1). He stipulates that asking about the workplace owner is a suitable way to start the questionnaire. Marx writes the question text into the TEX file as he would write it into any other text processing program. He knows that it is a good idea to number the questions because it helps the respondents to find the question they should start with and follow the intended sequence.

Variable-Single Environment

First, Marx thinks about the appropriate question type. In essence, he is interested in whether a capitalist or a limited company owns the workplace. Since he wants respondents to choose one of the two options or alternatively don't know, he programs a single-choice question. For this he uses the variable-single environment (Figure 2.4).

Figure 2.4: Single-Choice Question

```
\begin{document}
  \begin{Questionnaires}{1}
    Question 1: Does the shop in which you work belong to a capitalist or to a limited company?

    \begin{variable-single}{shop}{99}
      \answer{capitalist}{0,3}{vallab-sc}{checkbox-sc}\scoring{b=3}
      \answer{limited company}{0,2}{vallab-sc}{checkbox-sc}\scoring{b=2}
      \answer{don't know}{0,1}{vallab-sc}{checkbox-sc}\scoring{b=1}
    \end{variable-single}
  \end{Questionnaires}
\end{document}
```

Because the answers to the questionnaire should ultimately be written to a dataset for some statistical software, he must also define a variable name at this point. Marx chooses shop. Finally, he decides that persons who do not answer the question should be coded with a missing value and writes 99 into the second argument of the variable-single environment.

Answer & Scoring Command

Now, Marx starts programming the answer options. For each answer option, he defines a value label, a related value, and the answer options' position within the layout. The programming starts with the \answer command, followed by four arguments and the \scoring command (Figure 2.4). The \answer commands' first argument contains the value label, the second argument the graphical position, the third argument the style information for the value label, and the fourth argument a style for the answer box. The \scoring command assigns a value to the answer option.

- For the first answer option, Marx uses \scoring{b=3} and as value label {capitalist}. Therefore, the dataset that is created in the end will assign 3 to this answer category.
- The second answer option he labels with {limited company} and gives the value 2.
- The third answer option he offers respondents who do not know an answer to this question. Accordingly, Marx writes {don't know} in the first argument of the \answer command and assigns the value 1 using the \scoring command.

Positioning the Answer Options

Marx then starts to wonder about the layout position of the answer options. He learns that he can move each of the answer options along an imaginary x- and y-axis by defining their coordinates within the third argument of the \answer command (Figure 2.4). The third arguments' first number corresponds to the value on the x-axis, the second number to the value on the y-axis: {x,y}, Note that the coordinate values are comma-separated. If you want to use a decimal value, the decimal separator has to be a dot.

- Because he wants {capitalist} to be on top of the answer list, he sets the y-coordinate for this answer option to the highest value {0,3}.
- He chooses the coordinates {0,2} to position {limited company} in the middle.
- The {don't know} option he places at the bottom of the answer list by writing {0,1} within the third argument of the last \answer command.

Marx already found out that the third and fourth arguments are for designing labels and answer boxes. But he will take care of that later sometime (Section 2.3 and 2.5).

2.3 Rotate Answer Lists

Marx's second question concerns the age of children that work in factories (Question 2, Figure 2.1). As with the first question, he chooses a single-choice format. Since he wants to save space on the questionnaire, he displays the answer options horizontally. Marx proceeds the same as before, but then he adjusts the layout position and the answer styles (Figure 2.5).

Figure 2.5: Single-Choice Question—Horizontal Display

```
\begin{document}
  \begin{Questionnaires}{1}
    Question 2: What is the youngest age at which children are taken on?
      \begin{variable-single}{childage}{99}
        \answer{< 13 \\ years}{0,0}{vallab-sc, align=center}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=1}
        \answer{13 \\ years}{2,0}{vallab-sc, align=center}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=2}
        \answer{14 \\years}{4,0}{vallab-sc, align=center}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=3}
        \answer{15 \\years}{6,0}{vallab-sc, align=center}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=4}
        \answer{16 \\years}{8,0}{vallab-sc, align=center}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=5}
        \answer{> 16 \\years}{10,0}{vallab-sc, align=center}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=6}
        \answer{don't \\know}{13,0}{vallab-sc, align=center}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=7}
      \end{variable-single}
    \end{Questionnaires}
  \end{document}
```

Position the Answer Text

In order to display the answers horizontally, Marx varies the x-coordinate. After some trial and error, he ended up separating the answer text by `2` units on the imaginary x-axis. Since, he wants to put the {don't know} option somewhat apart, he decides to increase the horizontal spacing to `3` for that category (Figure 2.5).

Move the Answer Box

In the default setting (`checkbox-sc`) the answer box is aligned right to the answer text. Marx learns that he can change the default setting by adding `below=of lab\thecsvrow`[3] to the fourth argument of the `\answer` command (Figure 2.5). He wonders what `lab\thecsvrow`

[3] You can also use `right`, `left` and `above`.

means because it sounds a little bit complicated to him. He finds out that it is the name of the tikz node containing the value label. He also gets to know that the command \thecsvrow is essential when it comes to importing questionnaire information from a structured metafile. Since he does not want to try this now, he decides to leave it at that and takes care of it later[4].

Align the Answer Text

Finally, Marx wants to center the answer text. For this, he uses the third argument, which contains the style sets' name (vallab-sc). Per default, the style set aligns the text to the left. Marx overwrites the default setting by adding align=center to the third argument (Figure 2.5).

2.4 Add Filter Instructions

Marx is interested in whether the working conditions differ between urban and rural areas (Question 3, Figure 2.1). Since the answer options are mutually exclusive, Marx chooses the single-choice format one more time. He realized that programming this question went quickly, since he just copied the source code of the previous question and edited the necessary fields. While he is working on this question, the next question is coming up in his head. There, he wants to know if workers in a village earn enough to live from their work in the factory or if they have to do additional agricultural work (Question 4, Figure 2.1). Marx wants respondents who live in a village to answer the next question, while urban workers should skip that question. He this adds a filter instruction. To make sure that the instructions are recognized, he adds arrows, which lead the respondents' view to the correct question (Question 3, 2.1). Adding arbitrary text and graphical elements to a question is always possible by using tikz commands[5] within the variable-single environment (Figure 2.7).

Node Command

With a \node command you can place a field within an imaginary coordinate system — also called a tikzpicture (Figure 2.6). A \node command is structured as follows:

[4] For more information, see the SurveyAMC Manual (Chapter 3).

[5] tikz offers a great toolbox for designing a questionnaire. All the great things you can do with tikz are explained in more detail in the SurveyAMC Manual (Chapter 3). Apart from this you can take a look at the TikZ/PGF Manual at https://ctan.org/pkg/pgf.

```
\node[options] (name) at (x,y) {content};
```

First, you can name the node so that you can refer to it later. Second, you have to define an x- and a y-value to position the nodes' center. Finally, you can fill the node with content, for example, with text or an image. Within the square brackets at the beginning of the command, you can optionally style a node by adding node options (e.g., draw, fill=red, text width=5cm, xshift=1cm).

Figure 2.6: Single-Choice Question—Coordinate System of a tikzpicture

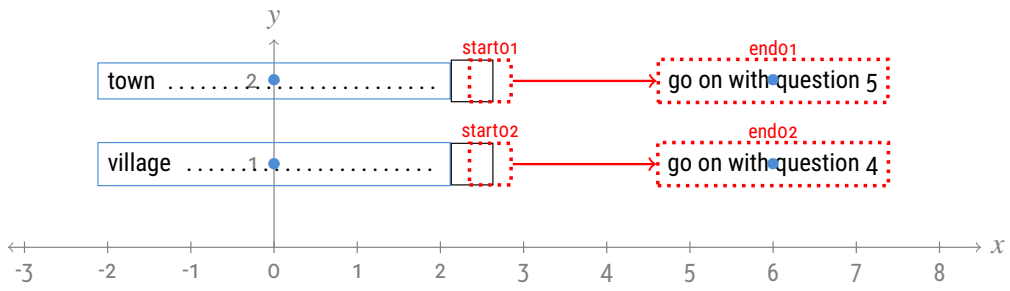


Figure 2.7: Single-Choice Question—Filter Instructions

```
\begin{document}
  \begin{Questionnaires}{1}
      Question 3: Is the shop in a town or in a village?

      \begin{variable-single}{shoplocal}{99}
            \answer{town}{0,2}{vallab-sc}{checkbox-sc}\scoring{b=2}
            \answer{village}{0,1}{vallab-sc}{checkbox-sc}\scoring{b=1}

               \node (start01) at (2.6,2) {};
               \node (start02) at (2.6,1) {};
               \node (end01) at (6,2) {go on with question 5};
               \node (end02) at (6,1) {go on with question 4};

                  \draw[->] (start01.east) -- (end01.west);
                  \draw[->] (start02.east) -- (end02.west);
      \end{variable-single}
  \end{Questionnaires}
\end{document}
```

Marx uses this information to add arrows to his textual filter instructions.

- First, he defines starting points for his arrows. By using the \node command (Figure 2.7), Marx plots invisible squares, each one near the respective answer box (Figure 2.6). He also names these two nodes with start01 and start02 to call them later when using the \draw command for drawing the arrows.

- Second, Marx uses the \node command to plot the textual filter instruction {go on with question 5} right to the answer option town and {go on with question 4} right to the answer option village. He names the nodes with end01 and end02 and by this, he is defining the ending points of the arrows.

- Now he connects the respective starting and ending points with the help of the \draw command. For example, he connects the eastern side of the node start01 with the western side of the node end01 by writing: \draw[->] (start01.east) -- (end01.west);

Later, Marx notices that there are several other options for adding filter instructions with TikZ. One of them he finds in the SurveyAMC Manual in Section 3.5.

2.5 Deal with Long Value Labels

To speed up the programming work, Marx copies the code from Question 3 and adapts it for Question 4 (Figure 2.1). He only changes the variable name and the value labels (Figure 2.8). However, when looking at the generated PDF document, he does not like the visual design of Question 4. Since the text width of the value labels is only 4cm in the default setting, longer text elements automatically break the line.

Figure 2.8: Single-Choice Question—Width of Value Labels

```
\begin{document}
  \begin{Questionnaires}{1}
      Question 4: If your shop is in the country, is there sufficiant work in the factory for your excistence, or are you obliged
      to combine it with agricultural labor?

      \begin{variable-single}{agrilabor}{99}
          \answer{sufficient work in the factory}{0,2}{vallab-sc, text width=6cm}{checkbox-sc}\scoring{b=2}
          \answer{combine it with agricultural labor}{0,1}{vallab-sc, text width=6cm}{checkbox-sc}\scoring{b=1}
      \end{variable-single}
  \end{Questionnaires}
\end{document}
```

Marx knows that line breaks within answer options can affect the readability of the question, so he wants to change that. He increases the value labels' text width by adding the node option text width=6cm to the third argument of the \answer command (Figure 2.8). While he's at it, he makes a note of other options that he may need later[6]:

font=\small\bfseries	➔	sets the nodes' font size to small and the font series to bold
minimum height=1cm	➔	defines the minimum size of the node
fill=white	➔	fills the node with a white background color
draw=red	➔	displays the node with a red colored border

2.6 Multiple-Choice Question

Marx wants to know more about the safety measures at the workplace. Specifically, he is interested if there are safety measures at the engine, the transmission, and the machinery (Question 5, Figure 2.1). Since respondents can tick any answer that applies, Marx programs a multiple-choice question (Figure 2.9). Unlike single-choice questions, the answer options of multiple-choice questions do not correspond to different values of a variable. Instead, each answer option of a multiple-choice question corresponds to a unique variable in the answer dataset.

Variable-Multi Environment

For programming the multiple-choice question Marx uses the variable-multi environment. Since Marx needs three answer options, he uses the variable-multi environment three times in a row (Figure 2.9). Each variable-multi environment has four arguments:

- Within the first argument Marx defines the variable names for each answer option: {safety1}, {safety2} and {safety3}.
- The second argument, he uses to type in the variable labels: {engine}, {transmission} and {machinery}. These are equivalent to the answer options, which respondents can select.
- The third argument contains the name of the tikzset to format the variable labels: {varlab-mc}. Marx likes the default layout, so he makes no changes here. But he

[6] He also tries to keep in mind that he has to separate node options with a comma.

keeps in mind that he can learn more about the styling options of multiple-choice questions in the SurveyAMC Manual.

- Within the fourth argument Marx specifies the missing value, which he needs to identify respondents who did not tick the respective answer option. Again, he chooses {99} for the missing value.

Figure 2.9: Multiple-Choice Question

```
\begin{document}
  \begin{Questionnaires}{1}
      Question 5: Are safety measures to prevent accidents applied to the engine, transmission, and machinery?
      \small Please tick all that apply. \normalsize

        \begin{variable-multi}{safety1}{engine}{varlab-mc}{99}
          \begin{values}
            \answer{}{2.5,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
          \end{values}
        \end{variable-multi}

        \begin{variable-multi}{safety2}{transmission}{varlab-mc}{99}
          \begin{values}
            \answer{}{2.5,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
          \end{values}
        \end{variable-multi}

        \begin{variable-multi}{safety3}{machinery}{varlab-mc}{99}
          \begin{values}
            \answer{}{2.5,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
          \end{values}
        \end{variable-multi}
  \end{Questionnaires}
\end{document}
```

Answer & Scoring Command

After defining the variables, Marx starts declaring the variables' values and their labels. To identify respondents who ticked the answer options (engine, transmission, machinery), he uses one \answer command within each variable-multi environment (see Figure 2.9). He remembers that the \answer command for programming the single-choice question looked

similar. But he notices a difference in the name of the tikzset that is used by default to style the answer box and the answer text. With multiple-choice questions, they have the ending mc instead of sc.

2.7 Reduce Code with a Loop

Marx is a bit worried by the amount of code to write for multiple-choice questions with many answer options. But then he notices that the answer options' code barely differs (Figure 2.9). Until now, he copied the variable-multi environment for each new answer option and adjusted the information for the variable name and the variable label. Everything else he left the same. So why not executing the code with a loop (Figure 2.10)?

Figure 2.10: Multiple-Choice Question—Loop

```
\begin{document}
  \begin{Questionnaires}{1}
      Question 5: Are safety measures to prevent accidents applied to the engine, transmission, and machinery?
      \small Please tick all that apply. \normalsize

          \foreach \label [count=\num from 1] in {engine, transmission, machinery}{

              \begin{variable-multi}{safety\num}{\label}{varlab-mc}{77}
                  \begin{values}
                    \answer{}{2.5,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
                  \end{values}
              \end{variable-multi}

          }
  \end{Questionnaires}
\end{document}
```

All he needs is a list of variable labels and names. With the \foreach command he writes a loop that does two things: First, the loop replaces the \label command with a string from a list on each run. Second, the loop replaces the \num command with a new number.[7] Marx uses the \num command to declare three different variable names:

[7] Note that you can choose every other description for the num and the label command as long as it is not assigned already.

{safety1}, {safety2}, and {safety3} (Figure 2.10).

2.8 Matrix Question

Marx is interested in the hygienic condition of the workplace (Question 6, Figure 2.1). He considers describing the hygienic conditions through nine dimensions: size of the rooms space, ventilation, temperature, plastering, lavatories, general cleanliness, noise of the machinery, metallic dust and dampness. Since he cannot measure the hygienic condition objectively, he decides to ask the workers how satisfied they are with each hygienic aspect. For this, he uses a matrix question. Similar as with multiple-choice questions, each answer option (hygienic aspect) corresponds to one variable in the answer dataset. But there are two differences: First, unlike multiple-choice questions, the variables of matrix questions have more than two values. Second, the value labels are displayed.

Variable-Multi Environment

Marx starts programming the first row of the matrix-question by using the variable-multi environment (Figure 2.11). For a variable name, he chooses {hygscale}, and for a variable label {size of the rooms space}. Since for this matrix question the text width of the variable labels is a little bit longer than usual, Marx changes the default setting by adding text width=5cm to third argument of the variable-multi environment. As missing value, he, again, chooses 99.

First Row

For the first row of the matrix question, Marx wants to display five value labels. Each of them, he programs with the \answer command (Figure 2.11). As with single-choice questions, Marx types the value label in the first argument and assigns the value with the \scoring command.

To position the answer options Marx uses the second argument of the \answer command. He keeps the y-coordinate fixed and only varies the x-coordinate. As a distance between the value labels {very satisfied}, {somewhat satisfied}, {somewhat dissatisfied} and {very dissatisfied}, he chooses 2 units on the imaginary x-scale: {3,0}, {5,0}, {7,0} and {9,0}. Only for the {dont know} option, he chooses a larger distance of 3 points to visually set this answer a bit off: {12,0}. By default, the answer options of multiple-choice questions are placed above the

answer boxes[8].

Figure 2.11: Matrix Question—Loop

```
\begin{document}
  \begin{Questionnaires}{1}
      Question 6: How satisfied are you with the hygienic conditions in the workshop regarding the following aspects?

      \begin{variable-multi}{hygscale}{size of the rooms space}{varlab-mc, text width=5cm}{99}
        \begin{values}
          \answer{very satisfied}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
          \answer{somewhat satisfied}{5,0}{vallab-mc}{checkbox-mc}\scoring{b=2}
          \answer{somewhat dissatisfied}{7,0}{vallab-mc}{checkbox-mc}\scoring{b=3}
          \answer{very dissatisfied}{9,0}{vallab-mc}{checkbox-mc}\scoring{b=4}
          \answer{don't know}{12,0}{vallab-mc}{checkbox-mc}\scoring{b=5}
        \end{values}
      \end{variable-multi}


      \foreach \label [count=\num from 1] in {ventilation, temperature, plastering, lavatories, general cleanliness,
      noise of machinery}{

        \begin{variable-multi}{hyg\num}{\label}{varlab-mc, text width=5cm}{99}
          \begin{values}
            \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
            \answer{}{5,0}{vallab-mc}{checkbox-mc}\scoring{b=2}
            \answer{}{7,0}{vallab-mc}{checkbox-mc}\scoring{b=3}
            \answer{}{9,0}{vallab-mc}{checkbox-mc}\scoring{b=4}
            \answer{}{12,0}{vallab-mc}{checkbox-mc}\scoring{b=5}
          \end{values}
        \end{variable-multi}

      }


  \end{Questionnaires}
\end{document}
```

[8] How this works exactly and how to change the default setting, is explained in the SurveyAMC Manual (Chapter 3.2).

Additional Rows

After programming the first row, Marx repeats the procedure for each further answer option of the matrix question (each additional row). The only difference is that he does not display the value labels. As practiced with the multiple-choice question before (Figure 2.10), Marx again uses a loop (Figure 2.11).

Loop

Marx defines a list of variable labels: {ventilation, temperature, plastering, lavatories, general cleanliness, noise of machinery, metallic dust, dampness}. By using the \label command, the loop inserts one label after the other at each run. Marx uses the \num command to create different variable names: {hyg1}, {hyg2}, {hyg3}, {hyg4}, {hyg5}, {hyg6}, {hyg7}, {hyg8}. Since Marx does not want to display the value labels for these answer options, he keeps the first argument of the \answer command empty (Figure 2.10).

2.9 Add Open Answer Questions

Text Question

At the questionnaires' end, Marx wants to ask about the risk of accidents at work. He has no idea what accidents can happen. For this reason, Marx is not able to create predefined response options. He decides to allow respondents to formulate the answer in their own words (Question 7, Figure 2.1).[9] To program the open answer question, Marx uses the rich toolbox of the tikz package (Figure 2.12).

Marx wants the respondents to describe their experienced work accidents in several sentences. He knows that the size of the answer field affects the answers' length (e.g., Israel, 2010). For this reason, he programs an answer field that spans several lines. He uses the tikzpicture environment and creates a 13cm wide and 7cm high field with the node command's help.

[9] Note that when you use open answer questions, the Auto-Multiple-Choice software cannot transfer them to the dataset. You have to process them manually.

Figure 2.12: Open Answer Question—Text

```
\begin{document}
  \begin{Questionnaires}{1}
      Question 7: Mention the accidents which have taken place to your personal knowledge.

        \begin{tikzpicture}
            \node[draw=black, minimum width=13cm, minimum height=7cm] at (3,0) {};
        \end{tikzpicture}

  \end{Questionnaires}
\end{document}
```

Numeric Question

With the last question (Question 8, Figure 2.1) Marx wants to know, what the average amount of holidays is usual for a worker within a typical year. Because Marx expects a high variation in the answers, a predefined answer list might become very long. For this reason, he decides to ask an open answer question. In this case, Marx is not interested in a detailed textual answer but precise information on the holidays' actual number. Because of that, he adjusts the answer field to two small boxes and hopes that this graphical presentation signals what kind of answer he expects.

Figure 2.13: Open Answer Question—Numeric

```
\begin{document}
  \begin{Questionnaires}{1}
      Question 8: State the number of holidays in the course of a year.

        \begin{tikzpicture}
            \node[draw=black, minimum width=5mm, minimum height=7mm] at (0,0) {};
            \node[draw=black, minimum width=5mm, minimum height=7mm] at (.75,0) {};
        \end{tikzpicture}

  \end{Questionnaires}
\end{document}
```

Again he uses the tikzpicture environment. Marx uses the \node commands' options to

define the answer fields' height (7mm) and width (5mm). To position the answer fields, he each sets an x- and a y-coordinate: (0,0) for the first box and (.75,0) for the second box.

2.10 Colored Background Boxes

Marx is happy with his questionnaire, but he is worried whether enough workers will fill out the form. He reads that the visual design of a questionnaire influences the willingness to answer it (Jenkins & Dillman, 1995). He gets to know that he can improve the general layout and, with that, the questionnaire's readability by displaying the questions and the respective answers within colored background boxes (Don A. Dillman, Gertseva, & Mahon-Haft, 2005).

Package Loading

Marx uses the tcolorbox package to display each question against a colored background[10]. He loads the package within the preamble of the TEX document by adding the \usepackage{tcolorbox} command (Figure 2.14).

Figure 2.14: Colored Background Box

```
\usepackage{tcolorbox}

\begin{document}
    \begin{Questionnaires}{1}

        \begin{tcolorbox}[colback=blue!10, colframe=gray, boxrule=1pt, arc=.5cm]

            Question 1: Does the shop in which you work belong to a capitalist or to a limited company?

            \begin{variable-single}{shop}{99}
                \answer{capitalist}{0,3}{vallab-sc}{checkbox-sc}\scoring{b=3}
                \answer{limited company}{0,2}{vallab-sc}{checkbox-sc}\scoring{b=2}
                \answer{don't know}{0,1}{vallab-sc}{checkbox-sc}\scoring{b=1}
            \end{variable-single}
        \end{tcolorbox}

    \end{Questionnaires}
\end{document}
```

[10]For detailed information see https://ctan.org/pkg/tcolorbox

Then, he switches to the document environment and inserts the first question into the tcolorbox environment.

Layout Options

Within the square brackets, Marx defines the layout options, also called the tcolorbox set. For a first try, he chooses a light blue background color by adding colback=blue!10 and a gray frame by adding colframe=gray. The option boxrule=1pt he uses to define the line width of the frame. And with arc=.5cm, he adjusts the frame corners' angle (Figure 2.14).

Style Sets

In the tcolorbox package documentation,[11] Marx reads that he can outsource the layout options by using the \tcbset command. He finds this useful because he wants to use the layout over and over again for all questions. By defining a \tcbset, he merely has to type its' name (e.g., myset) within the squared brackets of the tcolorbox environment instead of the detailed layout options (colback!10, colframe=gray, boxrule=1pt, arc=.5cm).

Marx is happy with his result. He can't wait to show his questionnaire to Engels.
We wish them every success with their survey!

[11] https://ctan.org/pkg/tcolorbox

3 User Manual

What if Marx could have used SurveyAMC as you can do? Maybe he would have created an appealing questionnaire like you have seen in the Tutorial (Figure 2.1). With SurveyAMC, you can create questionnaire layouts that meet scientific standards and therefore minimize the cognitive burden to fill in. Further, you do not have to manually type the answers into a dataset because the compatible software Auto-Multiple-Choice[1] processes them automatically to a CSV file. So if Marx could have used SurveyAMC, he likely could have reduced the respondents' burden and the burden for himself. The question about how burdensome a survey project is is not only a question of efficiency but also data quality. By reducing the respondents' burden and the survey data operators', the risk of nonresponse, measurement, and processing error decreases (Biemer & Lyberg, 2003). SurveyAMC supports the goal to achieve good data quality in self-administered survey projects because the software allows implementing the scientific principles of visual questionnaire design as well as a well-structured workflow that is traceable at every point for anyone working on the project.

The manual explains in great detail how to program different question types, like single-choice questions (Section 3.2), multiple-choice questions (Section 3.3) and matrix questions 3.4. The SurveyAMC LaTeX style package uses tikz pictures. For this reason, the manual explains the logic of tikz with the help of many examples[2]. Understanding the logic of tikz pictures is essential to take advantage of the several design possibilities offered by the tikz toolbox regarding question and answer formatting. Using tikz pictures for questionnaire programming makes it possible to add graphical elements (Section 3.5) like arrows for visual filter instructions (Section 3.5.1), open-answer fields (Section 3.5.2), and lines to connect answer boxes (Section 3.5.3).

Besides the formatting of individual questions, the manual explains how to design

[1] For more information see https://www.auto-multiple-choice.net/

[2] Note, that all question examples in this manual are taken from the source questionnaire of the European Social Survey, 2016. For a better illustration of the programming technique the original questions are partly shortened. For didactic reasons, the variable's names and values were also changed.

the general layout—for example, by using colored background boxes or a two-column page format (Section 3.1). The manual also holds a section on working with structured metafiles (Section 3.6). You will learn to create such a metafile by yourself and implement the metafiles' data with LaTeX and the online survey software, Limesurvey. This section also presents the accompanying software ClickAMC, which you can use to create paper-questionnaires automatically. Since the functionality of the SurveyAMC project relies on the third party software Auto-Mulitple-Choice, some information about this software is given as well in Section 3.7.

3.1 General Settings

3.1.1 Installation

The SurveyAMC project is part of the Auto-Mutiple-Choice software (AMC). AMC was developed for a Linux operating system. Though you can extract the AMC LaTeX package from the AMC software and run it on proprietary operating systems, it is recommended to install the complete software on Linux. This ensures that the software part used to create the questionnaire is always compatible with the software part needed to process the scanned answer sheets to a dataset.

The latest version of AMC is the developer version. It contains the [survey] option, which is necessary for programming a questionnaire. The developer version can be found at GitLab under https://gitlab.com/jojo_boulix/auto-multiple-choice/. The release version can be accessed via the web page of the AMC software (https://www.auto-multiple-choice.net/). The latter version does not yet contain the survey option.

If you want to install the developer version via GitLab, read the README beforehand. In short, it says, first download the repository, second, open the terminal (console) and change to the folder where you have saved the repository, then check for dependencies. For building the software, type make version_files, then make. Finally, type sudo make install, and the software will be installed.

More detailed information about accessing different versions and installation you will find on https://www.auto-multiple-choice.net/download.en.

After installing AMC you can load the LaTeX package within your TEX document and use the survey option for generating a machine readable paper questionnaire.

```
\usepackage[survey]{automultiplechoice}
```

3.1.2 General Page Layout

This section describes several options for designing the general page layout of the questionnaire. Figure 3.1 shows a code example. As with other TEX documents, first, a \documentclass is loaded; here {report}. By defining the document class also some options can be specified, like the font size (here 11 pt) and the page format (here A4). In this example, the document class also provides the option for a two-column format of the page; therefore, add twocolumn to the square brackets. Note, only choose a two-column layout if you are sure that the text width of your questions and answers will fit on half of the page width. For matrix questions or long answer texts, this is usually not the case. Besides the possibility of setting the column format globally via the document class, there is also the possibility of switching between single-column and multi-column format within the questionnaire environment. For this, use the {tcolorbox} package with its library {raster}. For detailed information about the tcolorbox toolbox, view the {tcolorbox} documentation[3].

After defining the document class, the usual settings follow, which are always necessary to control characters' input via the keyboard {inputenc} and the characters' output {fontenc} via the PDF document. Here the input is done as UTF-8 encoded characters and the output via a character set with Western European characters.

Figure 3.1 also shows how to change the font family. This example uses the SourceSansPro-LF from the sourcesanspro package as the default font. An overview of freely available LaTeX fonts provides the LaTeX Font Catalogue[4]. Analogous to the new definition of the switch command for roman fonts (\rmdefault), additionally, a font family of your choice can be assigned to the switch command for sans serif fonts (\sfdefault) and typewriter fonts (\ttdefault).

After choosing a suitable font for the questionnaire, the {automultiplechoice} package's [survey] option can be implemented. By loading the automultiplechoice package, several global settings can be defined by using AMC commands.

With the command \AMCtext{draft} a watermark can be defined, which is printed across the page; here {not for publication}. If there is no need for a watermark, leave the last bracket of the command empty. The \AMCtext{message} command works the same way, allowing to specify the contents of the footer; here {This is an example.}. Alternatively, use the \fancyfoot[C]{} command within the document environment since the automultiplechoice package loads the fancyhdr package.

[3] https://www.ctan.org/pkg/tcolorbox
[4] https://tug.org/FontCatalogue/

Figure 3.1: Options for the General Page Layout

```
\documentclass[11 pt, a4paper]{report}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{sourcesanspro}
\renewcommand{\rmdefault}{SourceSansPro-LF}
\usepackage[survey]{automultiplechoice}
\AMCtext{draft}{not for publication}
\AMCtext{message}{This is an example.}
\AMCboxColor{red}
\AMCboxDimensions{shape=oval}
\geometry{
    right=2cm, left=2cm, top=3cm, bottom=2cm,

}
\usepackage{tcolorbox}
\tcbuselibrary{raster}
\definecolor{myblue}{HTML}{58748f}
\definecolor{myorange}{HTML}{ff6100}
\tcbset{
    mystyle/.style={
        colback=myblue!10!white, coltext=myblue!40!black, colframe=myorange,
        arc=.1cm, boxrule=.75pt, boxsep=2.5mm,

    },

}
\newcounter{mycounter}
\def\myquest{\refstepcounter{mycounter}\themycounter}
% ..............................
\begin{document}
    \begin{Questionnaires}{1}
        \begin{tcolorbox}[mystyle]
            \AMCboxStyle{shape=oval}
            Question \myquest: Some people don't vote nowadays for one reason or another.
            Did you vote in the last national election?
            ...
        \end{tcolorbox}
    \end{Questionnaires}
\end{document}
```

However, do not write anything left [L] or right [R] in the footer; otherwise, the scan marks will disappear. These are important in order to be able to process the scanned answer sheets into a dataset. The \AMCboxColor command allows specifying the color of the answer boxes' border. For better illustration, the example uses a {red} border. If the questions and answers are displayed on a colored background that is dark enough, it may be useful to choose a white border for the answer box. With the \AMCboxDimensions command you can determine the shape of the answer boxes. By default, the answer boxes are square. However, you can also use round boxes (oval), or no boxes none.

The answer boxes' formatting not necessarily has to be defined globally in the preamble but can also be specified individually for each question within the questionnaire environment. The command for this is \AMCboxStyle. For example, size=2.5ex defines the box size, down=4ex moves the box down, and rule=5pt declares the line width for the box border. For more AMC commands view the documentation of the AMC Software.[5]

Since the {automultiplechoice} package loads the {geometry} package, you can set the documents' margins with the \geometry command. The default option is optimized for processing the answer sheets into one dataset. If you shorten the margins, as shown in this example (right=2cm, left=2cm, top=3cm, bottom=2cm), make sure that the scan marks do not move too close to the page margin, as this can cause problems with data processing.

Using the {tcolorbox} package's extensive toolbox, the questions can be displayed against a colored background or within a color-framed box. This can be useful to visually structure the questionnaire and make it easier for respondents to answer. Figure 3.1 shows an example of defining a style set (a so-called \tcbset) in the preamble and applying it within the questionnaire environment. The question here is to be displayed against a light blue background (colback=myblue!10!white) with an orange border (colframe=myorange) and a dark blue text color (coltext=myblue!40!black). The frame should have round corners (arc=.1cm) and a line width of .75pt. Furthermore, the distance between the text inside the colored box and the border should be (boxsep=2.5mm). The set is called (mystyle) and can be used again and again within the questionnaire environment. Of course, it is possible to create several sets. More information about the possibilities with the {tcolorbox} package provides the tcolorbox documentation.[6]

It may be useful to have the questions automatically numbered. This way, there has to be no renumbering when the order is changed or when questions are added or removed. Figure 3.1 shows an example of defining a counter in the preamble to use it within the

[5] https://www.auto-multiple-choice.net/doc.en
[6] https://www.ctan.org/pkg/tcolorbox

questionnaire environment to number the questions automatically; here \myquest.

After defining the general page layout in the preamble, the {Questionnaire} environment can be opened within the {document} environment to program the questionnaire. The last curved bracket of the questionnaire environment indicates how many questionnaires with individual barcodes and identifiers are to be generated when compiling the TEX document, here {1}.

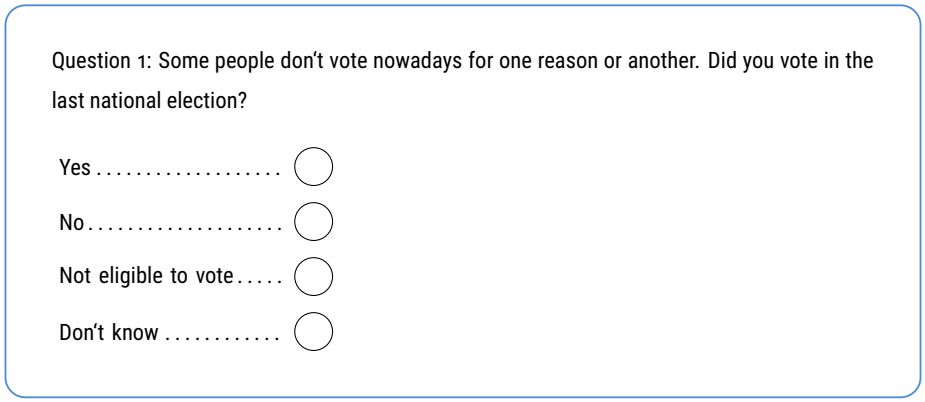
How the questions are now programmed is the subject of the following sections.

3.2 Single-Choice Question

3.2.1 Basics

Figure 3.2 presents an example[7] of a single-choice question. The question has four answer options from which respondents should select only one. In the dataset, this question corresponds to a variable that can take five values: four different values depending on which answer was chosen, and one value if the question was not answered (missing value).

Figure 3.2: Single-Choice Question—Picture



For programming this single-choice question, you need two environments: the variable-single environment and the values environment (Figure 3.3). Additionally, you can choose between squared and oval answer boxes. The default setting is square, and you can switch to round answer fields by using the command \AMCboxStyle{shape=oval}.

[7] Variable B13, European Social Survey, 2016

Variable-Single Environment

The variable-single environment indicates that a variable is defined, and it has two arguments: The first one specifies the variable name, in this example {vote}. The second argument specifies the missing value, here {99}.

Figure 3.3: Single-Choice Question—Code

```
Question 1: Some people don't vote nowadays for one reason or another. Did you vote in the last national election?

\AMCboxStyle{shape=oval}

\begin{variable-single}{vote}{99}
  \begin{values}
    \answer{Yes}{0,4}{vallab-sc}{checkbox-sc}\scoring{b=4}
    \answer{No}{0,3}{vallab-sc}{checkbox-sc}\scoring{b=3}
    \answer{Not eligible to vote}{0,2}{vallab-sc}{checkbox-sc}\scoring{b=2}
    \answer{Don't know}{0,1}{vallab-sc}{checkbox-sc}\scoring{b=1}
  \end{values}
\end{variable-single}
```

Values Environment

After specifying the variable name and the missing value, you can define the variables' values and value labels using the values environment. Within the values environment each answer option is specified by an \answer command. The \answer command has four arguments:

{1} The first argument includes the value label, here {Yes}, {No}, {Not eligible to vote} and {Don't know}.[8]

{2} Within the second argument, you can define the answer option's position by setting an x- and a y-coordinate, in this example {0,4}, {0,3}, {0,2} and {0,1}.

{3} The third argument is necessary for adjusting the answer options' layout. In Section 3.2.3, you can learn how this works exactly. Important here is that the third argument contains the name of the default layout setting for single-choice questions: vallab-sc.

[8] If you want to fill up the space between the end of the answer text and the beginning of the answer box with dots, you can write {Yes \dotfill}.

{4} Within the fourth argument, you can define the answer boxes' alignment. Per default, the answer boxes are aligned right to the answer text. The default setting is called by typing checkbox-sc within the fourth argument. How to change the answer boxes' alignment, you can learn in Section 3.2.2.

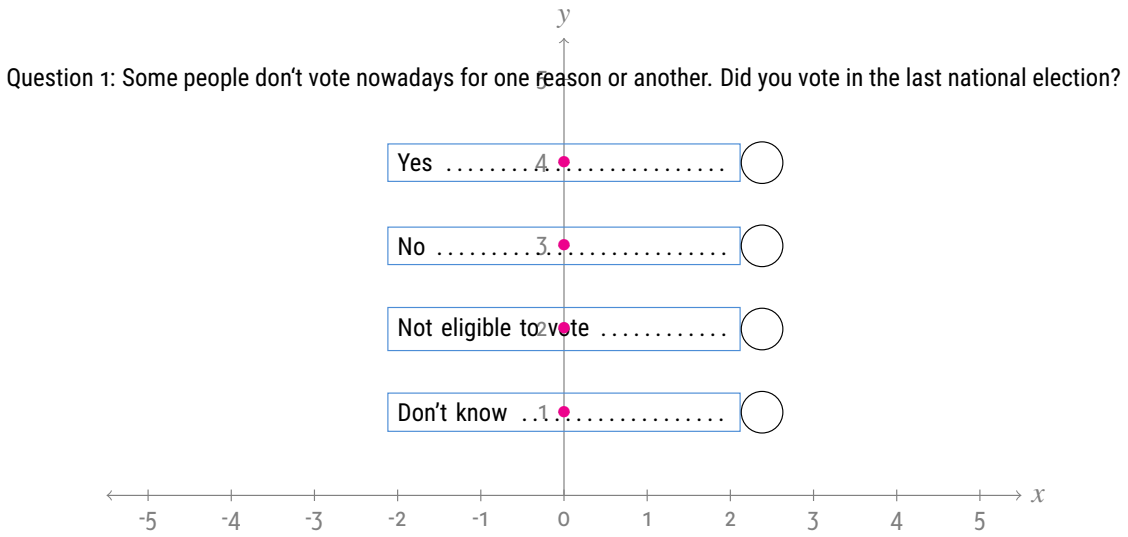
Scoring Command

At the end of each \answer command line has to be a \scoring command (Figure 3.3). With the \scoring command, you can define the variables' values. In this example, the \scoring command assigns the value {4} to the answer option {Yes}, {3} to {No}, {2} to {Not eligible to vote} and {1} to {Don't know}.

3.2.2 Positioning the Answer Options

Each answer options' position can be defined individually. The [survey] option of the automultiplechoice package uses the toolbox from the tikz package to create the answer options. Therefore, a so-called node contains the answer text. The answer box is automatically aligned to the right of the answer text[9]. As you can see in Figure 3.4, you can position these nodes within a imaginary coordinate system by defining an x- and a y-coordinate.

Figure 3.4: Single-Choice Question—Coordinate System



[9]How you can change the answer boxes' alignment you can learn in Section 3.2.4

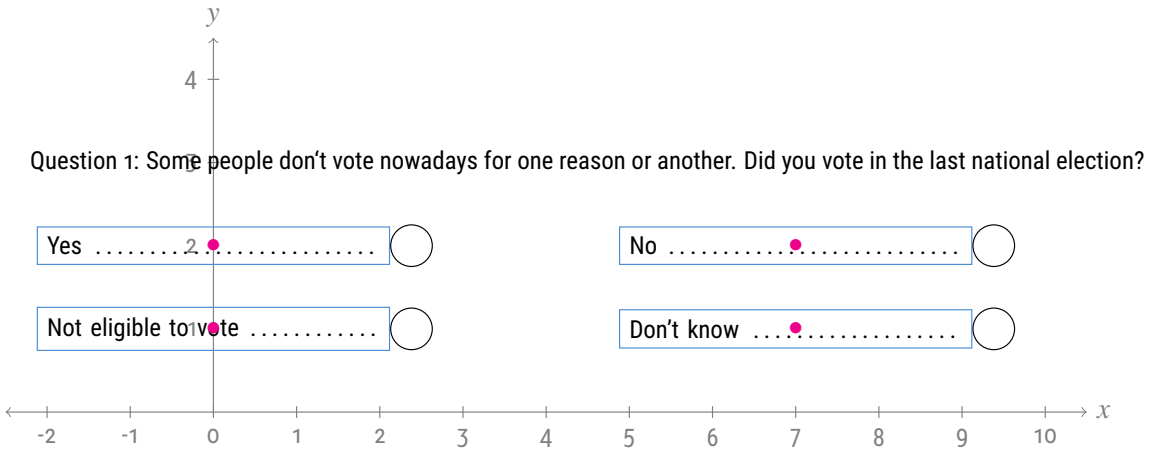
The coordinate point corresponds to the node's center, which is important because when you program the question, you do not see the nodes' frame, and you might mistakenly think that you are positioning the beginning of the answer text's node. For defining the coordinates of the answer options, you can use the second argument of the `\answer` command (Figure 3.3). In this example, the center of the node containing the answer {Yes} has the coordinates `x=0` and `y=4`. Respectively the center of the answer {No} is at `x=0` and `y=3`, {Not eligible to vote} at `x=0` and `y=2` and {Don't know} at `x=0` and `y=1`. By changing the coordinate points, you can move the answer options (text and box) along the x- and y-axis. Though by default the answer box is fixed to the right of the answer texts' node, you can change the box alignment, which is explained in Section 3.2.4.

Two-Column Format

Figure 3.5 shows an example of a two-column layout for a single-choice question. The only thing you have to change is the specification of the coordinate points (Figure 3.6).

The coordinates for the answer text nodes' center have changed to `x=0` and `y=2` for {Yes}, `x=7` and `y=2` for {No}, `x=0`, `y=1` for {Not eligible to vote} and `x=7` and `y=1` for {Don't know}.

Figure 3.5: Single-Choice Question—Two-Column Layout



For a better orientation, while positioning the answer options, you can display the nodes' frame by adding the node option `draw` to the third argument of the `\answer` command. Note that node options have to be comma-separated. More information about formatting the answer text you can find in Section 3.2.3.

Figure 3.6: Code for a Single-Choice Question—Two-Column Layout

```
Question 1: Some people don't vote nowadays for one reason or another. Did you vote in the last national election?


\AMCboxStyle{shape=oval}

\begin{variable-single}{vote}{99}
  \begin{values}
    \answer{Yes}{0,2}{vallab-sc}{checkbox-sc}\scoring{b=4}
    \answer{No}{7,2}{vallab-sc}{checkbox-sc}\scoring{b=3}
    \answer{Not eligible to vote}{0,1}{vallab-sc}{checkbox-sc}\scoring{b=2}
    \answer{Don't know }{7,1}{vallab-sc}{checkbox-sc}\scoring{b=1}
  \end{values}
\end{variable-single}
```

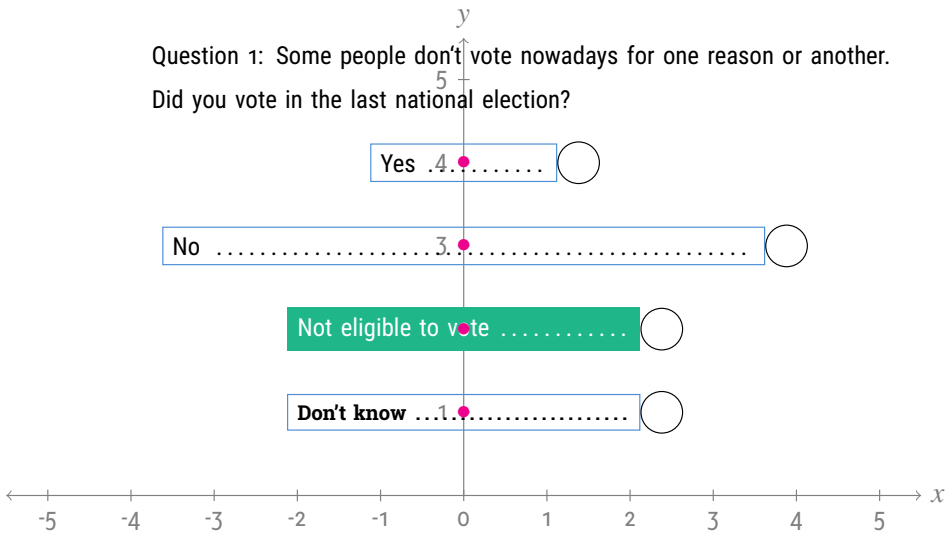
3.2.3 Formatting the Answer Text

The [survey] option of the automultiplechoice package uses tikzpictures to plot the answer options of a question. For single-choice questions, all answer options are created within one tikzpicture environment. Therefore, each answer text (value label) corresponds to one node, which you can format by using node options. The default layout for formatting the value labels is stored in the tikzset named vallab-sc, which is called by the \answer command's third argument (Figure 3.8). The default setting can be extended and overwritten by adding comma-separated node options.[10]

Figure 3.7 demonstrates different examples of how you can format the answer text. The code for these examples you can find in Figure 3.8. The first example in Figure 3.7 shows that you can decrease the text width of the value label {Yes} by adding text width=2cm to the third argument of the \answer command. Using the same node option, you can also increase the text width, as shown in the second example, where the text width of the value label {No} is changed to 7cm. The third example shows that you can change the answer texts' font and background color. By adding the node options text=white, fill=green to the third argument of the \answer command, {Not eligible to vote} appears with white font and green background.

[10] Another way is to define a new tikzset and replace the name of the default set vallab-sc with the new one.

Figure 3.7: Single-Choice Question—Answer Text Layout



Using node options like font=\sffamily\scriptsize\bfseries, you can also change the font family, font size, and font series of the answer text, as shown for the variable label {Don't know} in the example.

Figure 3.8: Code for a Single-Choice Question—Answer Text Layout

```
Question 1: Some people don't vote nowadays for one reason or another. Did you vote in the last national election?

\AMCboxStyle{shape=oval}

\begin{variable-single}{vote}{99}
  \begin{values}
    \answer{Yes}{0,4}{vallab-sc, text width=2cm}{checkbox-sc}\scoring{b=4}
    \answer{No}{0,3}{vallab-sc, text width=7cm}{checkbox-sc}\scoring{b=3}
    \answer{Not eligible to vote}{0,2}{vallab-sc, fill=green, text=white}{checkbox-sc}\scoring{b=2}
    \answer{Don't know}{0,1}{vallab-sc, font=\sffamily\scriptsize\bfseries}{checkbox-sc}\scoring{b=1}
  \end{values}
\end{variable-single}
```

Table 3.2.3 shows a selection of further options that might be useful for designing the answer text.

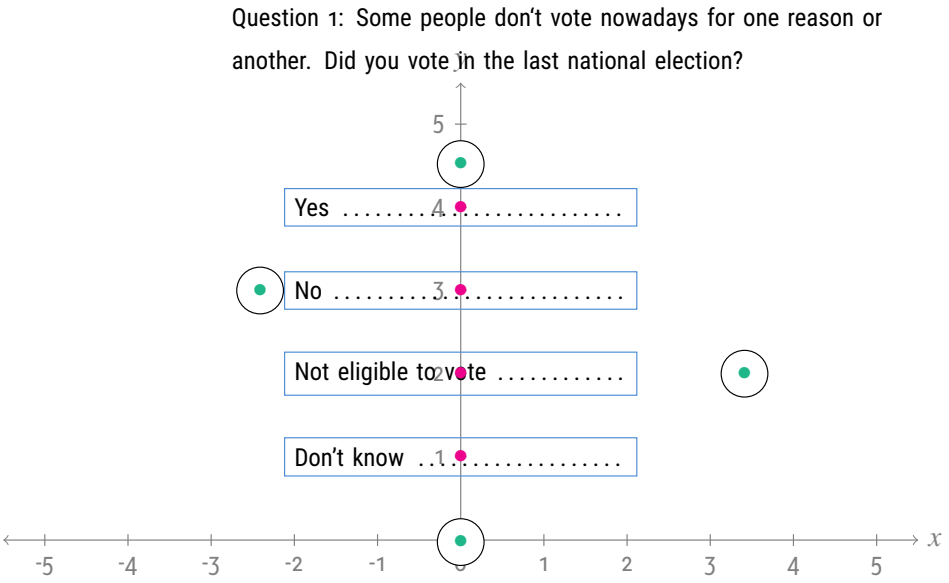
Table 3.1: Node Options

node option	function
minimum width=1cm	defines a minimum width for a node
minimum height=1cm	defines a minimum height for a node
inner sep=5pt	defines the distance of the nodes' frame to the nodes' content
draw=red	defines the color of the nodes' frame
fill=red	defines the color of the nodes' background
align=center	aligns the content within the node

3.2.4 Aligning the Answer Boxes

Since the answer options are plotted within tikzpictures, not only the answer texts' position (Section 3.2.3) but also the answer boxes' position can be controlled using node options within the fourth argument of the \answer command.

Figure 3.9: Single-Choice Question—Answer Box Alignment



In the default setting checkbox-sc the answer box is aligned right to the answer text. The [survey] option of the automultiplechoice package automatically names the answer texts' nodes with lab\thecsvrow. You can use this name to change the answer boxes' alignment. Figure 3.9 shows different examples of how you can rotate the answer box around the value labels' node, and Figure 3.10 shows the corresponding code for these examples.

The first example demonstrates how you can move the answer box above the value label {Yes} by adding the node option above= of lab\thecsvrow to the fourth argument of the \answer command. The second example uses the same node option as the first one, but exchanges above with left, so that the answer box appears left to the answer text {No}.

Figure 3.10: Code for a Single-Choice Question—Answer Box Alignment

```
Question 1: Some people don't vote nowadays for one reason or another. Did you vote in the last national election?

\AMCboxStyle{shape=oval}

\begin{variable-single}{vote}{99}
  \begin{values}
    \answer{Yes}{0,4}{vallab-sc}{checkbox-sc, above= of lab\thecsvrow}\scoring{b=4}
    \answer{No}{0,3}{vallab-sc}{checkbox-sc, left= of lab\thecsvrow}\scoring{b=3}
    \answer{Not eligible to vote}{0,2}{vallab-sc}{checkbox-sc, xshift=1cm}\scoring{b=2}
    \answer{Don't know}{0,1}{vallab-sc}{checkbox-sc, below= of lab\thecsvrow, yshift=-5mm}\scoring{b=1}
  \end{values}
\end{variable-single}
```

The example with the value label {Not eligible to vote} shows how you can control the space between the answer text and the answer box. Here, the node option xshift=1cm increases the distance. You also can combine node options. For example, the node option below= of lab\thecsvrow aligns the answer box below the answer text {Don't know}, and the option yshift=-5mm additionally moves the box 5mm down the y-axis.

3.3 Multiple-Choice Question

3.3.1 Basics

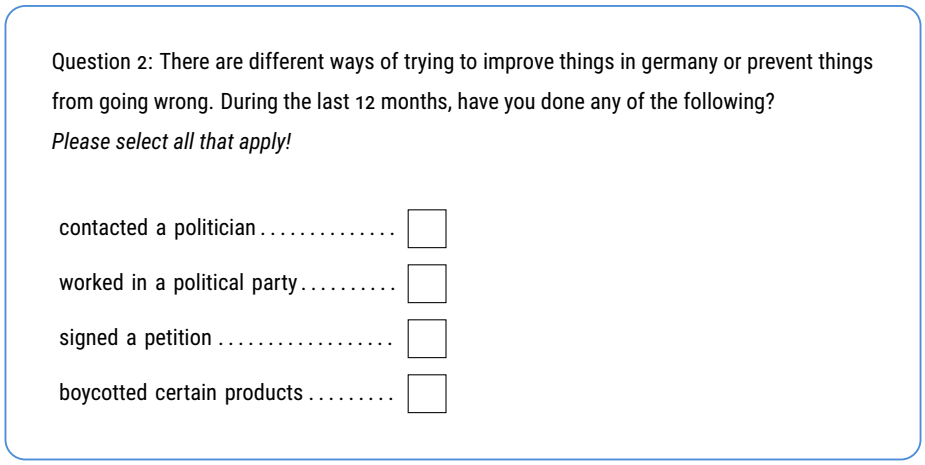
Figure 3.11 presents an example of a multiple-choice question[11] with four answer options from which respondents can select all that apply. In the answer dataset this question corresponds to four variables. Each variable can take two values: one if the answer option is selected and one if it is not selected (missing value).

[11] Variables B15, B16, B19 and B21, European Social Survey, 2016

Variable-Multi Environment

You can define a variable of a multiple-choice question by using the variable-multi environment. Since the example question in Figure 3.11 has four answer options, you need four variable-multi environments (Figure 3.12). The variable-multi environment has four arguments:

Figure 3.11: Multiple-Choice Question



{1} The first argument specifies the variable name, in this example: {improveo1}, {improveo2}, {improveo3}, {improveo4}.

{2} The second argument includes the variable label, here: {contacted a politician}, {worked in a political party}, {signed a petition} and {boycotted certain products}.

{3} Within the third argument you can format the variable labels. Unlike single-choice questions, the answer text for multiple-choice questions corresponds to the variable label and not to the value label (Section 3.2). The tikzset varlab-mc provides the default layout. How to change the variable labels' default layout, you can learn in Section 3.3.4.

{4} The fourth argument is for defining a missing value; in this example {99}.

Values Environment

Within the variable-multi environment, you can use the values environment to specify the variables' values and if necessary the value label.

Figure 3.12: Code of a Multiple-Choice Question

```
Question 2: There are different ways of trying to improve things in germany or prevent things from going wrong.
During the last 12 months, have you done any of the following?
\textit{Please select all that apply!}

\begin{variable-multi}{improveo1}{contacted a politician}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
  \end{values}
\end{variable-multi}

\begin{variable-multi}{improveo2}{worked in a political party}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
  \end{values}
\end{variable-multi}

\begin{variable-multi}{improveo3}{signed a petition}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
  \end{values}
\end{variable-multi}

\begin{variable-multi}{improveo4}{boycotted certain products}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
  \end{values}
\end{variable-multi}
```

Similar to programming single-choice questions, you can define each value and value label by using the **\answer** command. Also, with multiple-choice questions, the **\answer** command has four arguments, but there are some differences to those in single-choice questions:

{1} The first argument of the **\answer** command includes the value label. Since this example of a multiple-choice question displays no value label, the first argument remains empty.

{2} The second argument holds information about the positioning of the answer box. This gets important if you want to program more than one value, e.g., for matrix questions (Section 3.4).

{3} The third argument is for formatting the value label. The default layout is stored in the tikzset called vallab-mc. In Section 3.4 you can learn more about the available options for formatting the value labels within the variable-multi environment.

{4} The fourth argument is for moving the answer box. The default setting is stored in the tikzset called checkbox-mc. For more details, see Section 3.3.5.

Scoring Command

A \scoring command must follow at the end of each \answer command line (Figure 3.12). The \scoring command is for defining the variables' values. In this example, each variable has only two values: one if the respondent ticked the answer option, and one if not. As you can see from the code for this example (Figure 3.12), each variable can take the value 1, or the missing value {99}.

3.3.2 Using Loops

With single-choice questions, you can program several answer options within one variable-single environment. With multiple-choice questions, the programming of answer options works differently. Here, a new variable-multi environment is necessary for each answer option (Section 3.3). Quickly you will have to type much code (Figure 3.12). To avoid this, you can use a loop. As you can see in Figure 3.12, for each answer option, you only have to adjust two pieces of information—namely the variable name and the variable label within the arguments of each variable-multi environment. So, it is easy to iterate the code with a \foreach loop (Figure 3.13).

The \foreach loop repeats the code within the curly brackets for each item in the variable list: {contacted a politician, worked in a political party, signed a petition, boycotted certain products}. The variable list contains the variable labels for each answer option of the multiple-choice question. On each run, the loop inserts an item from the list at the position of the \label command. Additionally, the loop contains a counter. By writing the command \num the loop inserts the number of the current run. This example uses the counter to enumerate the variable labels.

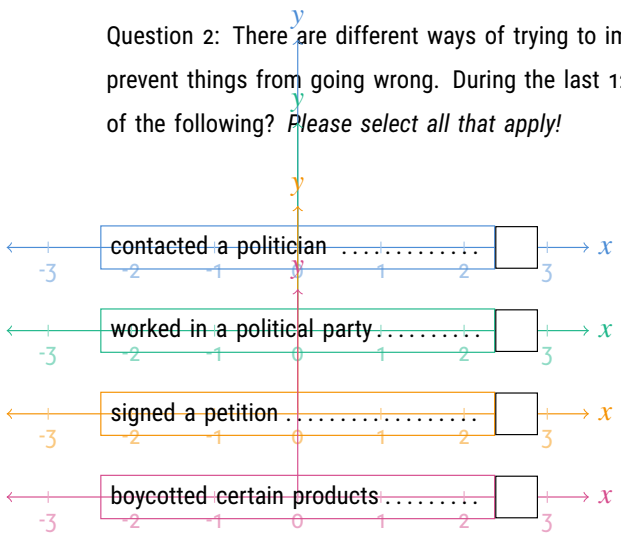
Figure 3.13: Code for a Multiple Choice Question—Loop

```
Question 2: There are different ways of trying to improve things in germany or prevent things from going wrong.
During the last 12 months, have you done any of the following? \textit{Please select all that apply!}

\foreach \label [count=\num from 1] in {contacted a politician, worked in a political party, signed a petition, boycotted
certain products}{

    \begin{variable-multi}{improve\num}{\label}{varlab-mc, text width=4.5cm}{99}
      \begin{values}
        \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
      \end{values}
    \end{variable-multi}

}
```

3.3.3 Positioning the Answer Options

Unlike single-choice questions (Section 3.2), the programming of multiple-choice questions creates a separate tikzpicture for each answer option (Figure 3.14). To change the vertical distance between the answer options, simply insert one of the many \vskip commands from the LaTeX toolbox in the following line after the closing command of the variable-multi environment (e.g., \smallskip, \medskip, \bigskip, \vksip5mm, \vskip1cm).

Figure 3.14: Multiple-Choice Question—Coordinate System



Two-Column Format

Also the answer options of a multiple-choice question can be positioned in a two-column format (Figure 3.15) by integrating the variable-multi environments in a tikzpicture to define the coordinates of each answer option (Figure 3.16).

Figure 3.15: Multiple-Choice Question—Two-Column Layout

Question 2: There are different ways of trying to improve things in germany or prevent things from going wrong. During the last 12 months, have you done any of the following?
Please select all that apply!

contacted a politician . . . . . ☐ signed a petition . . . . . ☐

worked in a political party . . ☐ boycotted certain products . . ☐

The respective code in Figure 3.17 shows that the tikzpicture contains four nodes from which each includes one variable-multi environment. In this example, the node containing the answer option {contacted a politician} is at the coordinates (-4,2); {worked in a political party} at (-4,1), {signed a petition} at (4,2) and {boycotted certain products} at (4,1).

Figure 3.16: Multiple-Choice Question—Two Column Layout, Coordinate System

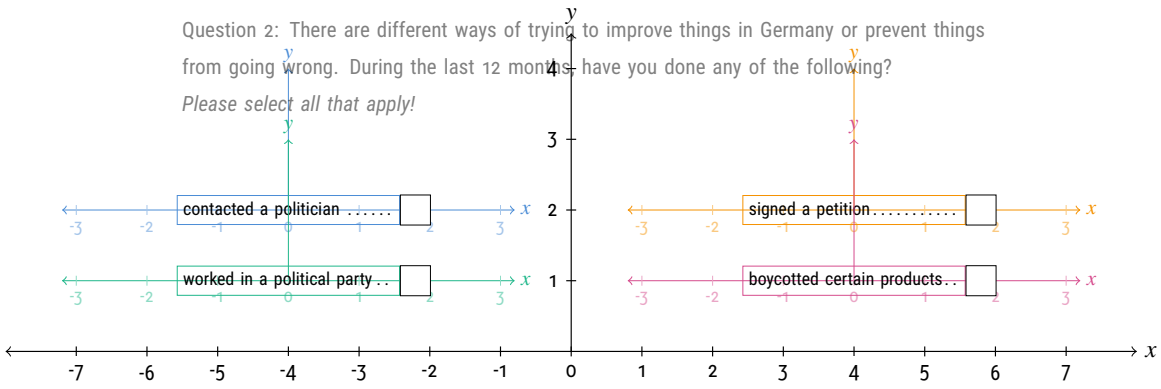


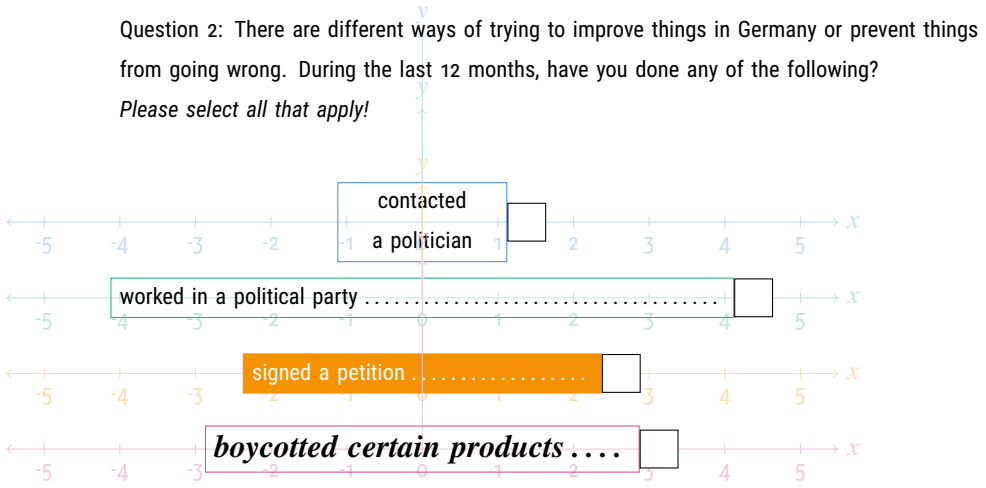
Figure 3.17: Code for a Two-Column Layout for a Multiple-Choice Question

```
Question 2: There are different ways of trying to improve things in germany or prevent things from going wrong.
During the last 12 months, have you done any of the following? \newline
\textit{Please select all that apply!} \newline

\begin{tikzpicture}
  \node at (-4,2) {
    \begin{variable-multi}{improveo1}{contacted a politician}{varlab-mc, text width=4.5cm}{99}
      \begin{values}
        \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
      \end{values}
    \end{variable-multi}
  };

  \node at (-4,1) {
    \begin{variable-multi}{improveo2}{worked in a political party}{varlab-mc, text width=3.5cm}{99}
      \begin{values}
        \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
      \end{values}
    \end{variable-multi}
  };

  \node at (4,2) {
    \begin{variable-multi}{improveo3}{signed a petition}{varlab-mc, text width=3.5cm}{99}
      \begin{values}
        \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
      \end{values}
    \end{variable-multi}
  };

  \node at (4,1) {
    \begin{variable-multi}{improveo4}{boycotted certain products}{varlab-mc, text width=4.5cm}{99}
      \begin{values}
        \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
      \end{values}
    \end{variable-multi}
  };
\end{tikzpicture}
```

3.3.4 Formatting the Answer Text

You can change the variable labels' default layout by adding node options within the third argument of the variable-multi environment.[12] The default settings' name is varlab-mc, and the tikzset contains the node options text width=4cm, align=left. As you can see in Figure 3.18, there are plenty of possibilities to style the answer text, such as choosing a different font style, text alignment, background, or font color.

Figure 3.18: Multiple-Choice Question—Answer Text Layout



The code for these examples (see Figure 3.19) shows how you can overwrite the default layout by adding comma-separated node options within the third argument of the variable-multi environment:

- text width=2cm, align=center reduces the text width of the variable label {contacted a politician} to 2cm and aligns the text to the center of the node
- text width=8cm increases the text width of {worked in a political party} to 8cm
- fill=orange, text=white defines the background color to be orange and the font color to be white for the variable label {signed a petition}
- font=\scriptsize\rmfamily\itshape sets an italic font shape, a bold font series and a large font size for the answer text {boycotted certain products}

[12] Alternatively, you can create a new tikzset and integrate it by replacing the default name with the new name.

Figure 3.19: Code for a Multiple-Choice Question—Answer Text Layout

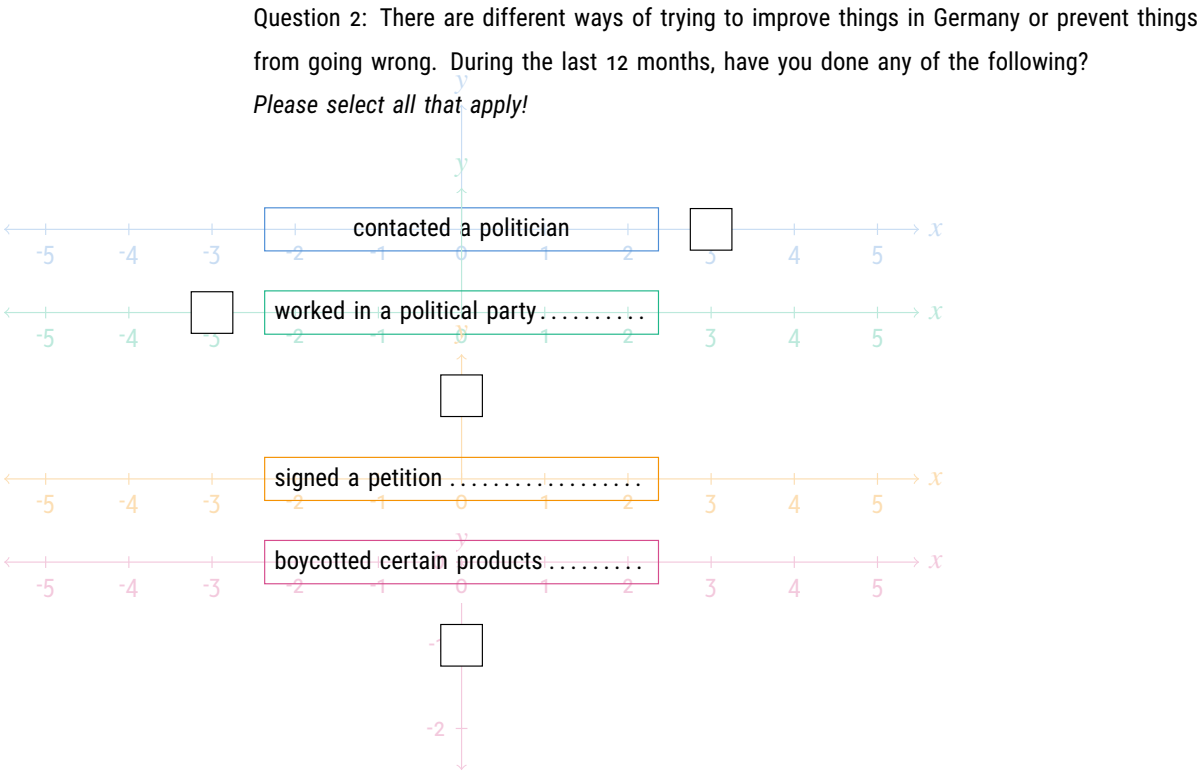
```
Question 2: There are different ways of trying to improve things in germany or prevent things from going wrong.
During the last 12 months, have you done any of the following? \newline
\textit{Please select all that apply!} \newline

\begin{variable-multi}{improve01}{contacted a politician}{varlab-mc, text width=2cm, align=center}{99}
  \begin{values}
    \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
  \end{values}
\end{variable-multi}

\begin{variable-multi}{improve02}{worked in a political party}{varlab-mc, text width=8cm}{99}
  \begin{values}
    \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
  \end{values}
\end{variable-multi}

\begin{variable-multi}{improve03}{signed a petition}{varlab-mc, fill=orange, text=white}{99}
  \begin{values}
    \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
  \end{values}
\end{variable-multi}

\begin{variable-multi}{improve04}{boycotted certain products}{varlab-mc, font=\scriptsize\rmfamily\itshape}{99}
  \begin{values}
    \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
  \end{values}
\end{variable-multi}
```

3.3.5 Aligning the Answer Boxes

Just as with single-choice questions, also with multiple-choice questions, the answer box can be moved around the answer text. Since multiple-choice questions are structured differently, the programming procedure differs. As you have already learned, each answer option of a multiple-choice question corresponds to a different tikzpicture. Within this tikzpicture, you can move the answer box along the imaginary coordinate system's x- and y axes. There are two ways to do this: Either use the second argument of the \answer

command and specify an x and a y value. Or select {0,0} as the coordinates. In this case, the answer box appears in the center of the node containing the text (here, the variable label). Then, within the fourth argument of the \answer command, you can change the default layout using the node options xshift or yshift. Figure 3.20 shows a multiple-choice question where the answer boxes have a different position. How to move the answer boxes is shown in Figure 3.21.

Figure 3.20: Multiple-Choice Question—Answer Box Alignment



By extending the default setting {checkbox-mc} in the fourth argument of the \answer command, for example, with the node option ...

- xshift=3cm, the answer box appears on the right side of the variable label {contacted a politician}.
- xshift=-3cm, the answer box appears on the left side of the variable label {worked in a political party}.
- yshift=1cm, the answer box appears above the variable label {signed a petition}.
- yshift=-1cm, the answer box appears below the variable label {boycotted certain products}.

Figure 3.21: Code for a Multiple-Choice Question—Answer Box Alignment

```
Question 2: There are different ways of trying to improve things in germany or prevent things from going wrong.
During the last 12 months, have you done any of the following? \newline
\textit{Please select all that apply!} \newline

\begin{variable-multi}{improveo1}{contacted a politician}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{}{0,0}{vallab-mc}{checkbox-mc, xshift=3cm}\scoring{b=1}
  \end{values}
\end{variable-multi}

\begin{variable-multi}{improveo2}{worked in a political party}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{}{0,0}{vallab-mc}{checkbox-mc, xshift=-3cm}\scoring{b=1}
  \end{values}
\end{variable-multi}

\begin{variable-multi}{improveo3}{signed a petition}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{}{0,0}{vallab-mc}{checkbox-mc, yshift=1cm}\scoring{b=1}
  \end{values}
\end{variable-multi}

\begin{variable-multi}{improveo4}{boycotted certain products}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{}{0,0}{vallab-mc}{checkbox-mc, yshift=-1cm}\scoring{b=1}
  \end{values}
\end{variable-multi}
```

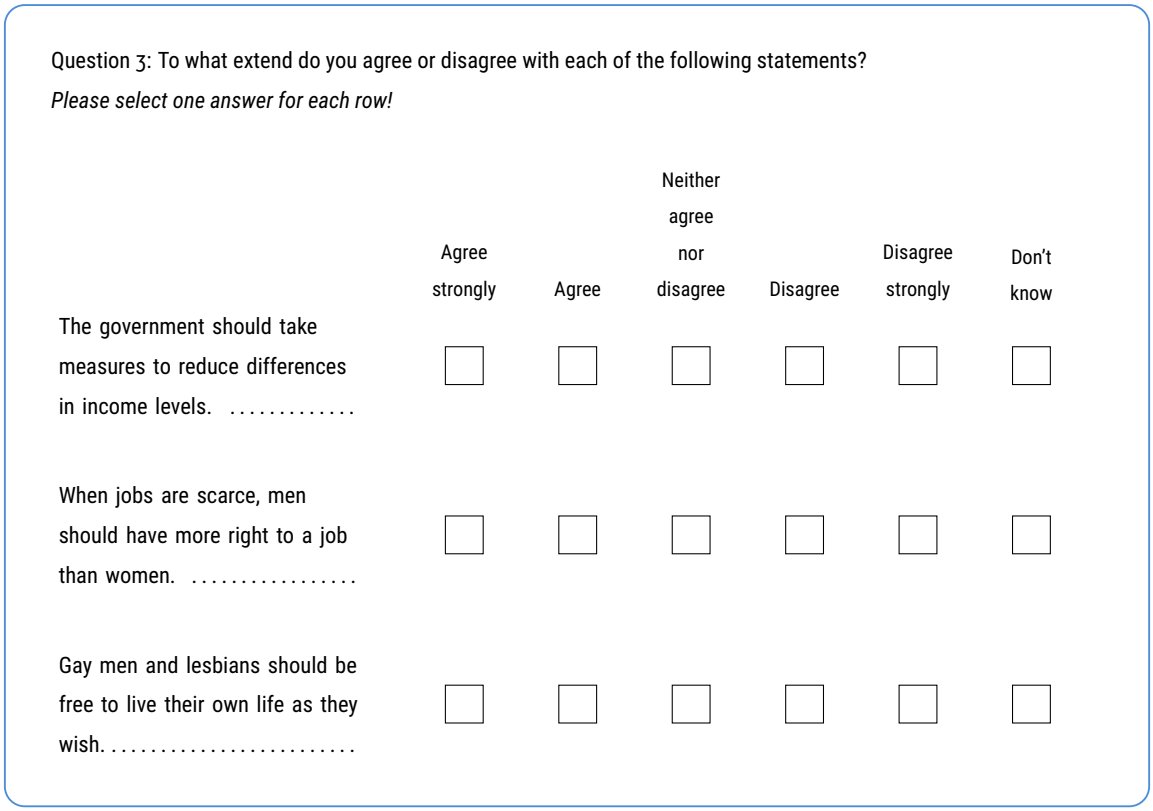
Note, that you could alternatively select an x- and y-coordinate different from zero within the second argument of the `\answer` command. Instead of using `xshift` and `yshift`, you could achieve the same result as in the previous example (Figure 3.20) by selecting the coordinates `{3,0}` for `{contacted a politician}`, `{-3,0}` for `{worked in a political party}`, `{0,1}` for `{signed a petition}`, and `{0,-1}` for `{boycotted certain products}`.

3.4 Matrix Question

3.4.1 Basics

Figure 3.11 presents an example of a matrix question[13]. This matrix question has three dimensions. For each dimension, the respondents can choose an answer from a five-level scale and an additional "don't know" option. Unlike multiple-choice questions, each variable of a matrix question can have more than two values. There are seven possible values in this example, one for each scale level, one for the "don't know" option, and one if no answer is selected (missing value).

Figure 3.22: Matrix Question



The programming of matrix questions works the same as for multiple-choice questions (Section 3.3)—with the only difference that for each row more than one answer box is defined.

[13] Variables B33, B33a, and B34, European Social Survey, 2016

Variable-Multi Environment

The matrix question in this example (Figure 3.22) has three dimensions which each correspondent to one variable in the dataset. For programming the variables you can use the variable-multi environment 3.23. The variable-multi environment has four arguments:

{1} The first argument specifies the variable name, in this example: {equal01}, {equal02}, {equal03}.

{2} The second argument includes the variable label, here: {The government should take measures to reduce differences in income levels.}, {When jobs are scarce, men should have more right to a job than women.}, and {Gay men and lesbians should be free to live their own life as they wish}.

{3} Within the third argument you can specify the variable labels' layout. In this example, the text width is increased by adding the node option text width=4.5cm.

{4} The fourth argument is for defining a missing value; in this example we chose 99 .

Values Environment

As you know from the Multiple-Choice Section (3.3), within the variable-multi environment, you can use the values environment to specify the variables' values and the value labels. Typically, Matrix questions show the answer scale only in the first line. To do this, you can specify the values and value labels for the first variable. For the following variables, you only define the values. For defining the values and value labels, you can use the \answer command within the values environment. In this example, respondents can choose their answer from a five level scale and the additional "'don't know"' option, which requires, in sum, six \answer commands. Each \answer command has four arguments:

{1} The first argument includes the value label. As mentioned before, we specify this argument only for the first variable (row) of the matrix question: {Agree strongly}, {Agree}, {Neither agree nor disagree}, {Disagree}, {Disagree strongly}, {Don't know}.

{2} The second argument holds information about the position of the answer box. Here, you can choose an x- and a y-coordinate to move the answer box within an imaginary coordinate system. Per default, the answer box appears at the center of the node containing the variable label. Because we want to move the answer box a bit to the right, we set the x-coordinate for the first box to 3 and leave the y-coordinate at 0. To set the following answer boxes at equal distance apart, we

choose coordinates with a difference of two units on the x-axis: {5,0}, {7,0}, {9,0}, {11,0}, {13,0}.

{3} Within the third argument, you can style the value labels. Per default, the value labels' position is above the answer box. The default text width is 1.4 cm, and the text is aligned to the center of the node. You can overwrite the default setting of the vallab-mc tikzset by adding node options. In this example, we want a smaller font size for the answer scale, so we add the node option font=small. In Section 3.4.3 you will find some more useful node options to style the answer scale of a matrix question.

{4} The fourth argument of \answer command includes the default tikzset {checkbox-mc} for formatting and positioning the answer box. Using the node options xshift and yshift, you can move the answer box within an imaginary coordinate system, which is actually the same as defining the coordinates within the second argument of the \answer command. Besides this alternative positioning option, you can style the boxes' border, for example, by changing the color. However, be careful with that because this might cause response effects and may lead to measurement error.

Scoring Command

After defining the variable name, variable label, and the value label, you can specify the answer dataset's values if respondents ticked one of the answer boxes. For this, use the \scoring command at the end of each \answer command line. In this example, for each variable, we choose the value 1 for the value label {Agree strongly}, 2 for {Agree}, 3 for {Neither agree nor disagree}, 4 for {Disagree}, 5 for {Disagree strongly}, and 6 for {Don't know}.

Figure 3.23: Code of a Matrix Question

```
Question 3: To what extend do you agree or disagree with each of the following statements?
\textit{Please select one answer for each row!}

\begin{variable-multi}{equal01}{The government should take measures to reduce differences in income levels.}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{Agree strongly}{3,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=1}
    \answer{Agree}{5,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=2}
    \answer{Neither agree nor disagree}{7,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=3}
    \answer{Disagree}{9,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=4}
    \answer{Disagree strongly}{11,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=5}
    \answer{Don't know}{13,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=6}
  \end{values}
\end{variable-multi}

\begin{variable-multi}{equal02}{When jobs are scarce, men should have more right to a job than women.}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
    \answer{}{5,0}{vallab-mc}{checkbox-mc}\scoring{b=2}
    \answer{}{7,0}{vallab-mc}{checkbox-mc}\scoring{b=3}
    \answer{}{9,0}{vallab-mc}{checkbox-mc}\scoring{b=4}
    \answer{}{11,0}{vallab-mc}{checkbox-mc}\scoring{b=5}
    \answer{}{13,0}{vallab-mc}{checkbox-mc}\scoring{b=6}
  \end{values}
\end{variable-multi}

\begin{variable-multi}{equal03}{Gay men and lesbians should be free to live their own life as they wish.}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
    \answer{}{5,0}{vallab-mc}{checkbox-mc}\scoring{b=2}
    \answer{}{7,0}{vallab-mc}{checkbox-mc}\scoring{b=3}
    \answer{}{9,0}{vallab-mc}{checkbox-mc}\scoring{b=4}
    \answer{}{11,0}{vallab-mc}{checkbox-mc}\scoring{b=5}
    \answer{}{13,0}{vallab-mc}{checkbox-mc}\scoring{b=6}
  \end{values}
\end{variable-multi}
```

3.4.2 Using Loops

As you may have noticed, programming a matrix question requires a lot of coding, especially if the question has several rows (variables). Of course, you can speed up programming by copying and pasting the code. The only information that has to be adjusted are the variable name, variable label, value labels, and values. So, since it is not too much information that varies for coding the variables of a matrix question, you could also use a \foreach loop (Figure 3.24).

Figure 3.24: Code for a Matrix Question—Loop

```
Question 3: To what extend do you agree or disagree with each of the following statements?
\textit{Please select one answer for each row!}


\begin{variable-multi}{equal01}{The government should take measures to reduce differences in income levels.}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \foreach {\label} [count=\num from 1] in {Agree strongly, Agree, Neither agree nor disagree, Disagree, Disagree strongly, Don't know}{

        \answer{\label}{1cm+\num*2cm,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=\num}

    }
  \end{values}
\end{variable-multi}


\foreach {\label} [count=\num from 1] in {When jobs are scarce{,} men should have more right to a job than women., Gay men and lesbians should be free to live their own life as they wish.}{

    \begin{variable-multi}{equal\num}{\label}{varlab-mc, text width=4.5cm}{99}
      \begin{values}
        \foreach {\label} [count=\num from 1] in {Agree strongly, Agree, Neither agree nor disagree, Disagree, Disagree strongly, Don't know}{

            \answer{}{1cm+\num*2cm,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=\num}

        }
      \end{values}
    \end{variable-multi}

}
```

Let's start with the matrix question's first row. Typically, the first row differs from the following ones since it shows the value labels as an answer scale. To program the first row, we use the variable-multi environment, as we have learned in the previous section

(Section 3.4.1). With the \foreach command, we have to write the \answer command only once, since it is repeated automatically for each string within a string list. In this example, the string list contains the value labels for the matrix question's first variable.[14] On each run, the loop replaces the \label command[15] with a value label from the list. The loop also contains a counter (\num)[16] that you can use to create different values. You can also use the counter for the coordinates to position each answer boxes. In this example, the boxes are separated two units on the x-axis, so the counter is multiplied with 2cm. Let's see what the loop produces on each run:

First run: \answer{Agree strongly}{1cm+1*2cm,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=1}

Second run: \answer{Agree}{1cm+2*2cm,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=2}

Third run: \answer{Neither agree nor disagree}{1cm+3*2cm,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=3}

Fourth run: \answer{Disagree}{1cm+4*2cm,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=4}

Fifth run: \answer{Disagree strongly}{1cm+5*2cm,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=5}

Sixth run: \answer{Don't know}{1cm+6*2cm,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=6}

Now, we can program the matrix question's remaining rows. Again, we use the \foreach command to repeat the \answer command. Since for these rows the value labels are not displayed, the \label command is not used within the \answer command's first argument. Everything else stays the same. The \foreach command repeats the variable-multi environment for each remaining variable (row). For this, there is a list that contains the variable labels. On each run, the loop replaces the \label command with one of the variable labels of the list. As this \foreach loop also includes a counter, we use the \num command to generate several variable names. Let's see what the loop produces on each run for the first line of the variable-multi environment:

First run: \begin{variable-multi}{equal1}{When jobs are scare, men should have more rights to a job than women.}{varlab-mc, text width=4.5cm}{99}

Second run: \begin{variable-multi}{equal2}{Gay men and lesbian should be free to live their own life as they want.}{varlab-mc, text width=4.5cm}{99}

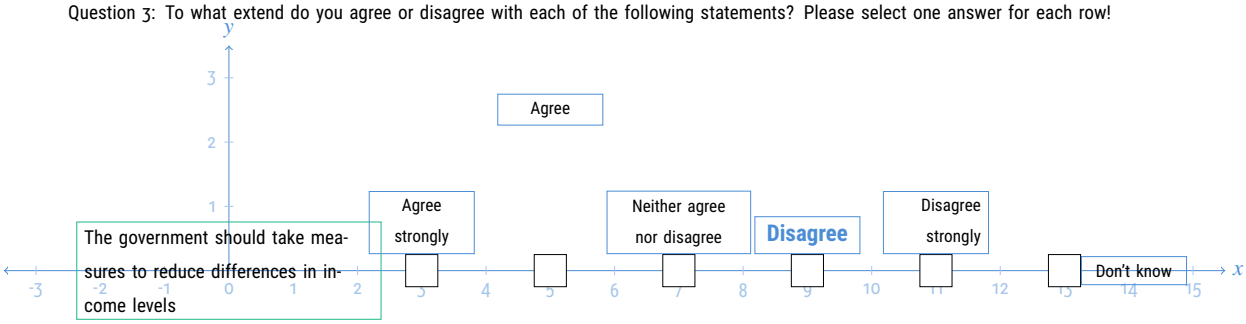
[14] Note that the value labels have to be comma-separates. If a value label contains a comma itself, you can protect it with curly brackets.
[15] Note that you are free to choose any name for this command, e.g. \x or \mycommand
[16] You can also choose this command name.

The matrix question in this example has only three rows, so using a loop does not necessarily reduce code. Typically matrix questions have many more than two rows. In this case, you should sincerely consider proceeding, as shown in this example.

3.4.3 Formatting the Answer Scale

You can style the answer scale of a matrix question individually. For this, you can use the third argument of the `\answer` command. The default layout setting for the answer scale is stored in the `tikzset vallab-mc`. Per default, the text width is set to `1.4cm`, the text is aligned to the center, and the node containing the value label is positioned above the answer box. Figure 3.26 shows some examples that may give you an idea for an individual layout set.

Figure 3.25: Matrix Question—Formatting the Answer Scale



1. For the first value label on the answer scale {Agree strongly}, no adjustments are made. Here, the the default layout setting is shown `vallab-mc`.

2. With the node option `yshift=2cm`, the second value label {Agree} is moved two centimeters on the y-axis.

3. For {Neither agree nor disagree}, the text width is increased by adding `text width=20mm` to the third argument of the `\answer` command.

4. In the fourth example {Disagree},the font size, series, and color are changed by adding the node options `font=\large\bfseries\color{blue}`.

5. Within the fifth node,the value label {Disagree strongly} is aligned to the right using `align=right`.

6. The last examples shows how to rotate the value label {Don't know} around the answer box. You can overwrite the default setting by adding `right=of box\thecsvrow`.

Figure 3.26: Code for a Matrix Question—Formatting the Answer Scale

```
Question 3: There are different ways of trying to improve things in germany or prevent things from going wrong.
During the last 12 months, have you done any of the following? \textit{Please select all that apply!}

\begin{variable-multi}{equal1}{The government should take measures to reduce differences in income lev-
els.}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \answer{Agree strongly}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
    \answer{Agree}{5,0}{vallab-mc, yshift=2cm}{checkbox-mc}\scoring{b=2}
    \answer{Neither agree nor disagree}{7,0}{vallab-mc, text width=20mm}{checkbox-mc}\scoring{b=3}
    \answer{Disagree}{9,0}{vallab-mc, font=\large\bfseries\color{blue}}{checkbox-mc}\scoring{b=4}
    \answer{Disagree strongly}{11,0}{vallab-mc, align=right}{checkbox-mc}\scoring{b=5}
    \answer{Don't know}{13,0}{vallab-mc, right=of box\thecsvrow}{checkbox-mc}\scoring{b=6}
  \end{values}
\end{variable-multi}
```

Working with node options might at first be a little bit confusing, so it could be helpful to display the node frame for a better orientation. For this, add draw to the third argument of the \answer command. When you finished your layout adjustment, remove the draw option. You can find a small overview of some more node options in the previous Section 3.2 in Table 3.2.3. If you make several adjustments, it might be useful to store your node options in a tikzset. By this, you do not have to type all options over and over again, since you just have to replace the default sets' name vallab-mc with the name of the new tikzset you have created.

3.5 Graphical Add-ons

In self-administered questionnaires, easy to understand filter instructions are essential since there is no interviewer to help the respondents navigate through the questionnaire. So, supporting textual instructions through visual elements like arrows or symbols might be a good idea. With the [survey] option of the automultiplechoice package, it is no problem to add visual elements to questions since these are tikzpictures. The tikz package provides an excellent toolbox for positioning and formatting graphical elements within an imaginary coordinate system. You can use the tikz toolbox to add navigational arrows, create open answer fields, or connect the answer boxes. Remember to wisely choose these elements, namely only to improve the respondents' understanding of filling in the questionnaire

and not for subjective esthetic reasons.

3.5.1 Adding Filter Instructions

The process of adding filter instructions differs depending on the question type. For this reason, this section provides you with two examples, one for single-choice questions and one for multiple-choice questions. However, these examples describe only one way of adding filter instructions. If you are firm with the tikz toolbox[17], you might find more solutions.

Single-Choice Questions

Figure 3.27 shows a single-choice question[18] with two filter instructions. In Section 3.2, you have learned that a single-choice questions is built within one tikzpicture. Each answer option corresponds to a node within a tikzpicture environment. For this reason, you can program additional nodes, for example, to add textual filter instructions or to draw navigational arrows within this imaginary coordinate system.

Figure 3.27: Single-Choice Question—Filter Instructions

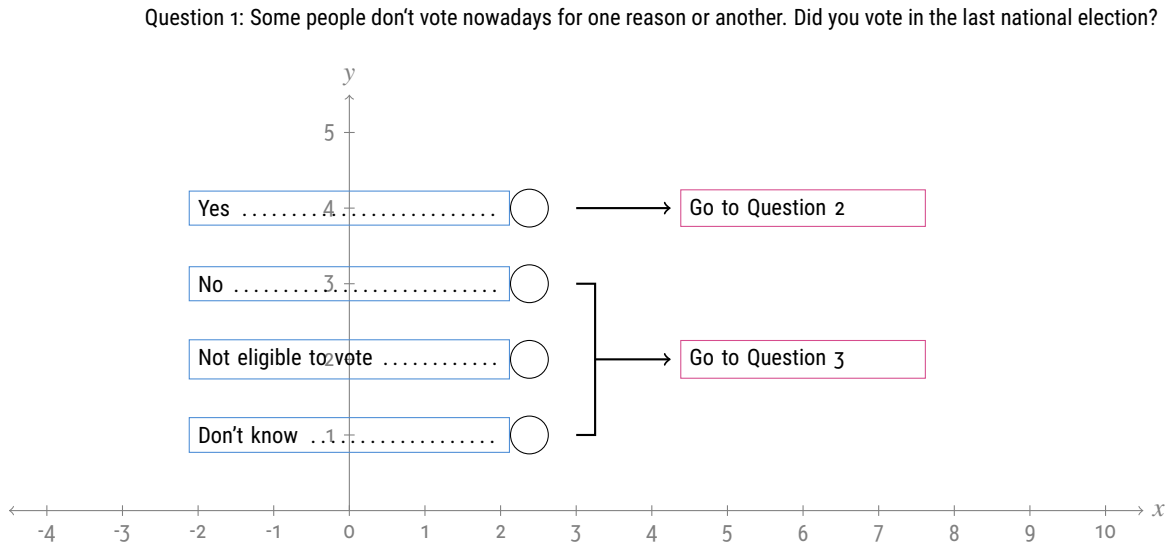


Figure 3.28 shows the code for adding filter instructions to a single-choice question. First, you can create a node that contains the text {Go to Question 2}. To position the node right to the value label {Yes}, you have to choose coordinates for the nodes' center

[17] For more information visit https://www.ctan.org/pkg/pgf
[18] Variable B13, European Social Survey, 2016

(x=6 and y=4). Then, you can use the \draw command to draw an arrow starting at the coordinates (3,4) and ending at (4.25,4).

Programming the second filter instruction is a little bit more advanced. Again, it would be best to start with programming the node containing the textual information {Go to Question 3}. Since you know that the node containing the value label {Not eligible to vote} is positioned at the coordinate 2 on the y-axis, you can also use this value for placing the textual information. For the x-coordinate, you can use the same value (4.25) as for the first instruction {Go to Question 2}.

Figure 3.28: Code for a Single-Choice Question—Filter Instructions

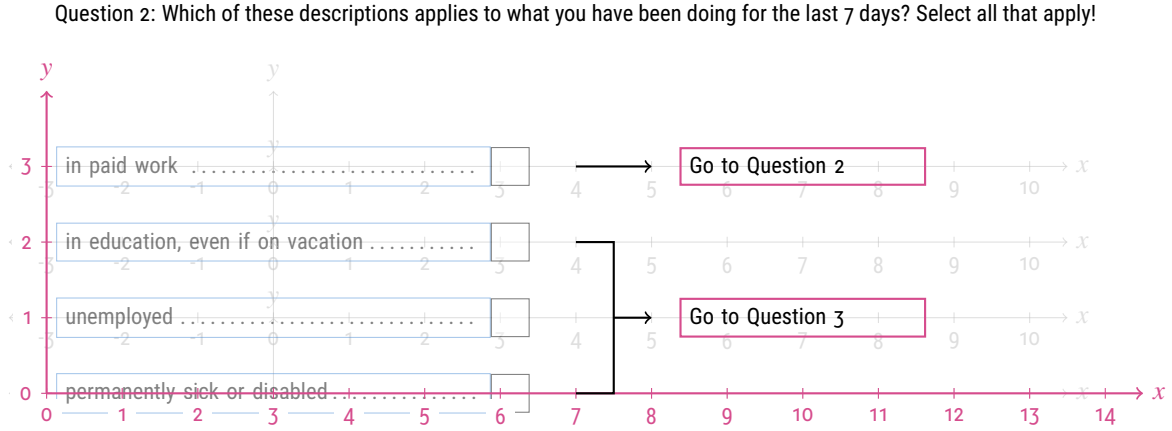
```
Question 1: Some people don't vote nowadays for one reason or another. Did you vote in the last national election?


\AMCboxStyle{shape=oval}

\begin{variable-single}{vote}{99}
  \begin{values}
    \answer{Yes}{0,4}{vallab-sc}{checkbox-sc}\scoring{b=4}
    \answer{No}{0,3}{vallab-sc}{checkbox-sc}\scoring{b=3}
    \answer{Not eligible to vote}{0,2}{vallab-sc}{checkbox-sc}\scoring{b=2}
    \answer{Don't know}{0,1}{vallab-sc}{checkbox-sc}\scoring{b=1}

        \node[text width=3cm] at (6,4) {Go to Question 2};
        \draw[->, thick] (3,4) - - (4.25,4);


        \node[text width=3cm] at (6,2) {Go to Question 3};
        \draw[thick] (3,3) - - (3.25,3) - - (3.25,1) - - (3,1);
        \draw[->, thick] (3.25,2) - - (4.25,2);

  \end{values}
\end{variable-single}
```

For drawing the brace that visually connects the last three answer options, you can use the \draw command. You can control the course of the line by defining coordinates. In this example, the line starts at (3,3), it then goes 0.25 units to the right (3.25,3), then 2 units down the y-axis (3.25,1), and finally 0.25 units to the left (3,1). After you have drawn the brace, you can plot the arrow, also using the \draw command. The procedure is the same as for the first arrow – you only have to change the y-coordinate (Figure 3.28).

Multiple-Choice Questions

The process of adding filter instructions to multiple-choice questions[19] is different from the procedure for single-choice questions because the answer options do not appear within one tikzpicture environment but in separate environments for each answer option (Figure 3.29). As mentioned before, there are different ways of adding visual elements to multiple-choice questions. This example shows you how you can add filter instructions while also using the \foreach command (Section 3.3.2) for multiple-choice questions.

Figure 3.29: Multiple-Choice Question—Filter Instructions



As you can see in Figure 3.29, multiple-choice questions are built by overlapping tikzpictures. If you want to add graphical elements, the most straightforward way is to overlay the question with an additional picture that contains the textual instructions and the arrows.

Figure 3.30 shows the respective code for this example. First, you can decrease the line space between the question and the tikzpicture containing the filter instructions so that both pictures lie on top of each other. Then you can position the additional nodes within the overlaying coordinate system — the pink colored one in Figure 3.29. Within this coordinate system, programming the instructions for multiple-choice questions is similar to single-choice questions.

First, position the node containing the textual instruction {Go to Question 2} at the coordinates x=10 and y=3. Then, use the \draw command to plot the arrow starting right of the answer label {in paid work} and ending left to the instructional text. As coordinates, use (7,3) for the starting point and (8,3) for the ending position.

[19] Variable F17a, European Social Survey, 2016

Second, you can plot the node for the textual instruction {Go to Question 3} at the coordinate (10,1), which corresponds to the right side of the answer label {unemployed}. Then, you can draw the brace, which clasps the three last answer options. The line starts at (7,2), then goes 0.5 units to the right (7.5,2), then 2 units down the y-axis (7.5,0) and then 0.5 units to the left (7,0). Finally, we draw the arrow, starting at (7.5,1) and ending at (8,1).

Figure 3.30: Code for a Multiple-Choice Question—Filter Instructions

```
Question 2: Which of these descriptions applies to what you have been doing for the last 7 days?
\textit{Select all that apply!}

  \foreach \label [count=\num from 1] in {in paid work, in education{,} even if on vacation, unemployed, permanently
  sick or disabled}{

        \begin{variable-multi}{status\num}{\label}{varlab-mc, text width=5.5cm}{99}
          \begin{values}
            \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
          \end{values}
        \end{variable-multi}

  }

\vskip-8.5mm

\begin{tikzpicture}[overlay]
      \node[text width=3cm] at (10,3) {Go to Question 2};
      \draw[->, thick] (7,3) - - (8,3);

      \node[text width=3cm] at (10,1) {Go to Question 3};
      \draw[thick] (7,2) - - (7.5,2) - - (7.5,0) - - (7,0);
      \draw[->, thick] (7.5,1) - - (8,1);
\end{tikzpicture}
```

3.5.2 Adding Open Answer Fields

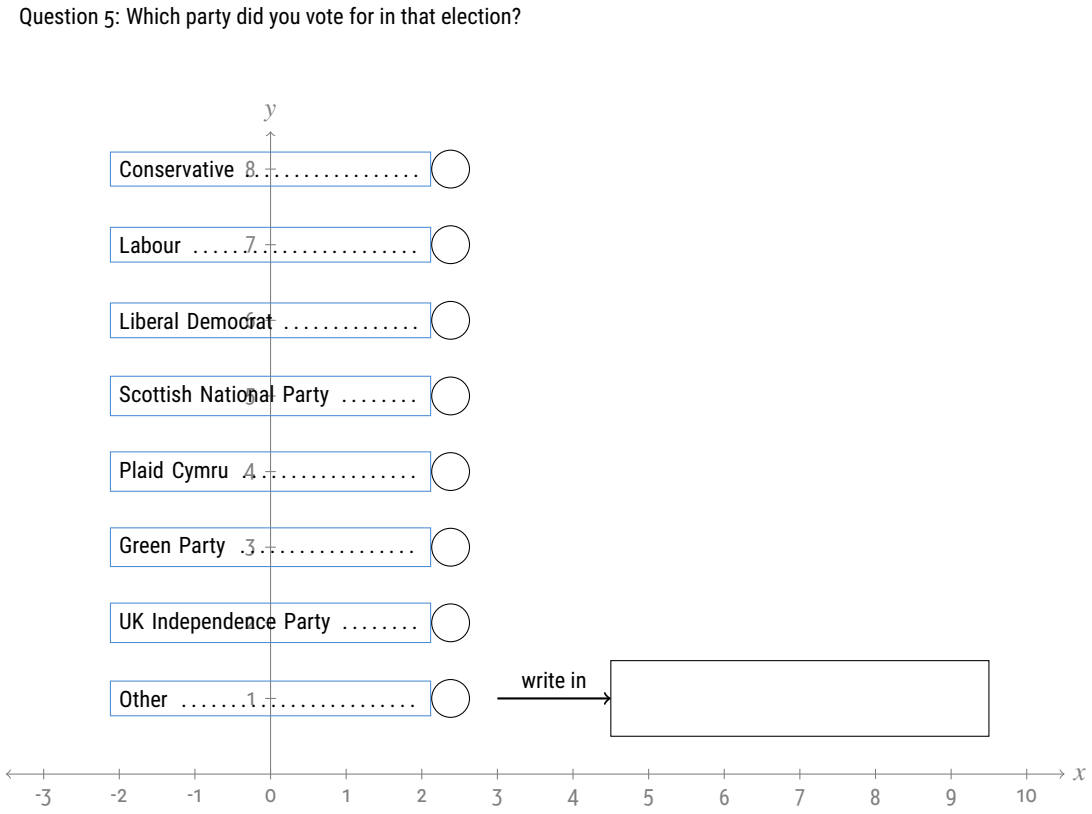
The process of adding open answer fields differs depending on the question type. For this reason, this section provides you with two examples, one for single-choice questions and one for multiple-choice questions. However, these examples describe only one way

of adding open answer fields. If you are firm with the tikz package[20], you might find more solutions.

Single-Choice Questions

Figure 3.31 shows a single-choice question[21] with an open answer field for respondents to write in another party they voted for than those presented by the answer list. The procedure of adding an open answer field is similar to the one of adding filter instructions (Section 3.5.1).

Figure 3.31: Single-Choice Question—Open Answer Field



First, you have to decide about the size of the answer field. It should be big enough to fill in the name of the party but not much bigger since we know that people tend to write as much as there is space (e.g., Don A. Dillman et al., 2014, p.194). In this example, we create an answer field with 5cm width and 1cm height (Figure 3.32). To

[20] For more information visit https://www.ctan.org/pkg/pgf
[21] Variable B14, England, Scotland & Wales codes, European Social Survey, 2016

place the answer field to the right side of the answer label {Other}, you can position the answer field's center at the the x-coordinate 7 and y-coordinate 1.

After plotting the answer field, you can draw the arrow. For this, use the \draw command. The arrow starts at the coordinates (3,1) and ends at (4.5,1) (Figure 3.32). To motivate the respondents to write down their answers, you might want to display write in above the arrow. For this, you can insert a node within the \draw command. Note that in this case, the node command has no backslash (node[above] {write in}).

Figure 3.32: Code for a Single-Choice Question—Open Answer Field

```
Question 5: Which party did you vote for in that election?

\AMCboxStyle{shape=oval}

\begin{variable-single}{party}{99}
  \begin{values}
    \answer{Other}{0,1}{vallab-sc}{checkbox-sc}\scoring{b=1}
    \answer{UK Independence Party}{0,2}{vallab-sc}{checkbox-sc}\scoring{b=2}
    \answer{Green Party}{0,3}{vallab-sc}{checkbox-sc}\scoring{b=3}
    \answer{Plaid Cymru}{0,4}{vallab-sc}{checkbox-sc}\scoring{b=4}
    \answer{Scottish National Party}{0,5}{vallab-sc}{checkbox-sc}\scoring{b=5}
    \answer{Liberal Democrat}{0,6}{vallab-sc}{checkbox-sc}\scoring{b=6}
    \answer{Labour}{0,7}{vallab-sc}{checkbox-sc}\scoring{b=7}
    \answer{Conservative}{0,8}{vallab-sc}{checkbox-sc}\scoring{b=8}

    \node[draw, minimum width=5cm, minimum height=1cm] at (7,1) {};
    \draw[->, thick] (3,1) - - node[above] {write in} (4.5,1);
  \end{values}
\end{variable-single}
```

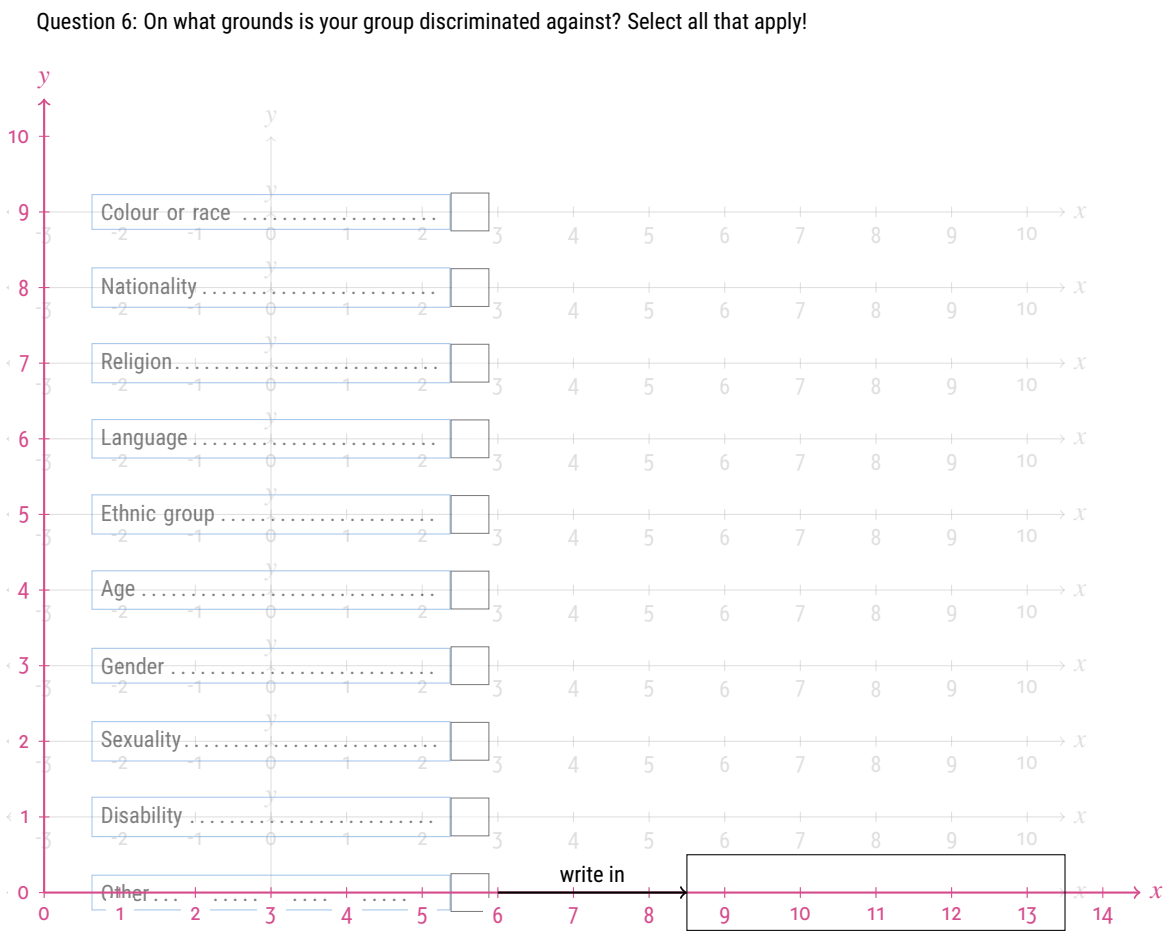
Multiple-Choice Questions

Adding open answer fields to multiple-choice questions works a little bit differently than for single-choice questions. This is because each answer option of a multiple-choice question is generated as one tikzpicture. Therefore, the easiest way is to program another tikzpicture containing the open answer field and then position it as an additional layer over the tikzpictures including the answer options of the multiple-choice question. Figure 3.33 shows a multiple-choice question[22], where respondents can write down other reasons

[22] Variable C19, European Social Survey, 2016

than those that are mentioned by the pre-defined answer list.

Figure 3.33: Multiple-Choice Question—Open Answer Field



The respective code in Figure 3.34 shows how you can add an open answer field while also using the \foreach command[23] for the answer options of the multiple-choice question. As you can see, the code for the open answer field and the arrow is similar to the one in the single-choice example (Section 3.5.2). To overlay the pictures, you have to do two things: First, add the option overlay to the tikzpicture environment that contains the open answer field. Second, reduce the line space by using the \vskip command. Beware that there is a minus sign before the millimeter specification (\vskip-8.5mm). If you wish to place the open answer field beside another answer option, adjust the line space, and find the right position.

[23] Explained in greater detail in Section 3.3.2

Figure 3.34: Code for a Multiple-Choice Question—Open Answer Field

```
Question 6: On what grounds is your group discriminated against?
\textit{Select all that apply!}

  \foreach \label [count=\num from 1] in {Colour or race, Nationality, Religion, Language, Ethnic group, Age, Gender, Sexuality, Disability, Other}{

        \begin{variable-multi}{discrim\num}{\label}{varlab-mc, text width=4.5cm}{99}
          \begin{values}
            \answer{}{3,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
          \end{values}
        \end{variable-multi}

  }

\vskip-8.5mm

\begin{tikzpicture}[overlay]
      \node[draw, minimum width=5cm, minimum height=1cm] at (11,0) {};
      \draw[->, thick] (6,0) - - node[above] {write in} (8.5,0);
\end{tikzpicture}
```

3.5.3 Connecting Answer Boxes

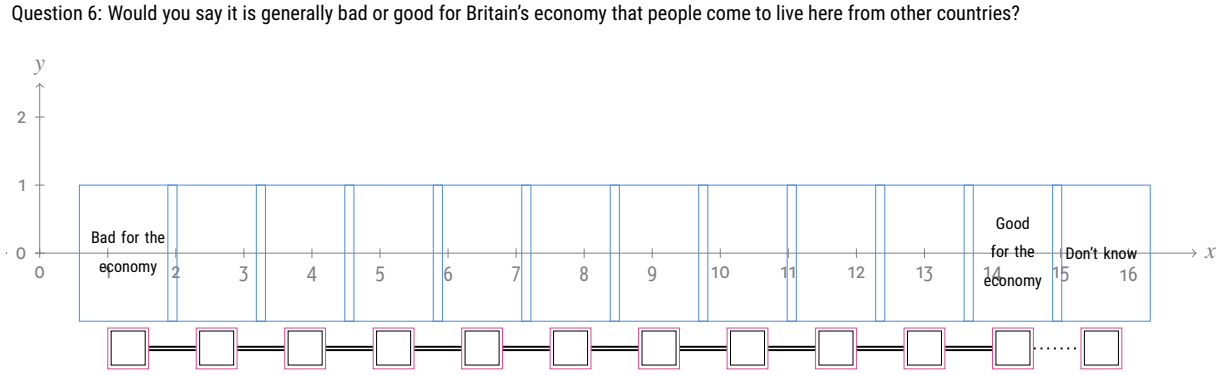
The process of connecting answer boxes differs depending on the question type. For this reason, this Section provides you with two examples, one for single-choice questions and one for matrix questions. However, these examples describe only one way of connecting answer boxes. If you are firm with the tikz toolbox[24], you might find more solutions.

Single-Choice Questions

Figure 3.35 shows a single-choice question[25] with horizontally displayed answer options[26]. Additionally, the answer boxes are connected to support respondents in finding the right answer box. This example uses two different line formats to visually separate the {don't know} option from the main answer scale. Figure 3.35 helps you to understand how the answer scale is created within a tikzpicture.

[24] For more information visit https://www.ctan.org/pkg/pgf
[25] Variable B41, European Social Survey, 2016
[26] Explained in greater detail in Section 3.2.4

Figure 3.35: Single-Choice Question—Connected Answer Boxes



This example not only explains how you can connect answer boxes visually, but it is also shows how you can create macro definitions that will make your code more clear and efficient. For a better understanding of the code example, the macro definitions are explained first.

You can use the preamble of the TEX document to generate macros. With macro definitions, you can control values within the document environment from the preamble. This example shows two types of macros:

new command

This example uses the new command \dist to control the coordinates of the answer labels over the preamble (Figure 3.36). So if you decide to change the position of the answer labels, you do not have to adjust each coordinate manually anymore. Instead, you only change the value in the second argument of the dist commands' definition in the preamble.

tikz style

The example also shows how you can move the style options for formatting the value labels into the preamble (Figure 3.36). For this, create a tikzset in the preamble and choose a name. In this case, the new tikzset is called my-style. By replacing the default name vallab-sc within the third argument of the \answer command with the new name my-style, you can now make changes over the preamble, instead of typing in the style options separately for each answer option.

Figure 3.36: Code for a Single-Choice Question—Connected Answer Boxes

```
\newcommand{\dist}{1.3}

\tikzset{
  my-style/.style={
    text width=1.2cm,
    minimum height=2cm,
    align=center,
    font=\scriptsize,

  },

}

\begin{document}
\begin{Questionnaires}{1}

Question 6: Would you say it is generally bad or good for Britain's economy that people come to live here from other countries?

\begin{variable-single}{discrim}{99}
  \begin{values}
    \answer{Bad for the economy}{1*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=1}
    \answer{}{2*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=2}
    \answer{}{3*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=3}
    \answer{}{4*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=4}
    \answer{}{5*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=5}
    \answer{}{6*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=6}
    \answer{}{7*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=7}
    \answer{}{8*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=8}
    \answer{}{9*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=9}
    \answer{}{10*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=10}
    \answer{Good for the economy}{11*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=11}
    \answer{Don't know}{12*\dist,0}{my-style}{checkbox-sc, below=of lab\thecsvrow}\scoring{b=12}

    \draw[double] (1.52,-1.35) -- (2.37,-1.35);

    \foreach \x in {1,...,9}{
      \draw[double] (1.52+\x*1.3,-1.35) -- (2.37+\x*1.3,-1.35);

    }

    \draw[dotted] (1.52+10*1.3,-1.35) -- (2.37+10*1.3,-1.35);


  \end{values}
\end{variable-single}
```

After understanding the new macro commands, we will look at how to create the lines connecting the answer boxes. You can program the lines within the values environment of the single-choice question by using the \draw command from the tikz package. For the first line that connects the left box with the one beside, try to find the coordinates for the starting (1.52,-1.35) and the ending point (2.37,-1.35). For the next line—the one between the second and the third box—you need to know the distance between the right side of the first box and the right side of the second box. By try and error you will find out that the distance is 1.3 units in this example. With these two pieces of information (coordinates and distance), you can use a systematic approach to plot the remaining lines. The \foreach loop runs through a list of numbers from 1 to 9 and multiplies the x-coordinates of each starting and ending point.

The last line between the {good for the economy} and the {don't know} option you can plot manually by copying the \draw command and replacing \x with 10. You can use a dotted line by exchanging the option dashed with dotted. Alternatively you can use a lot of more line types (solid, densely dotted, loosely dotted, dashed, densely dashed, loosely dashed, dashdotted, ...) and line widths (ultra thin, very thin, thin, semithick, thick, very thick, ultra thick).

Matrix Questions

Connected answer boxes are a practical tool for matrix questions, as they visually separate the sub-questions and help respondents not to slip in the row accidentally.

Figure 3.37: Matrix-Question—Connected Answer Boxes

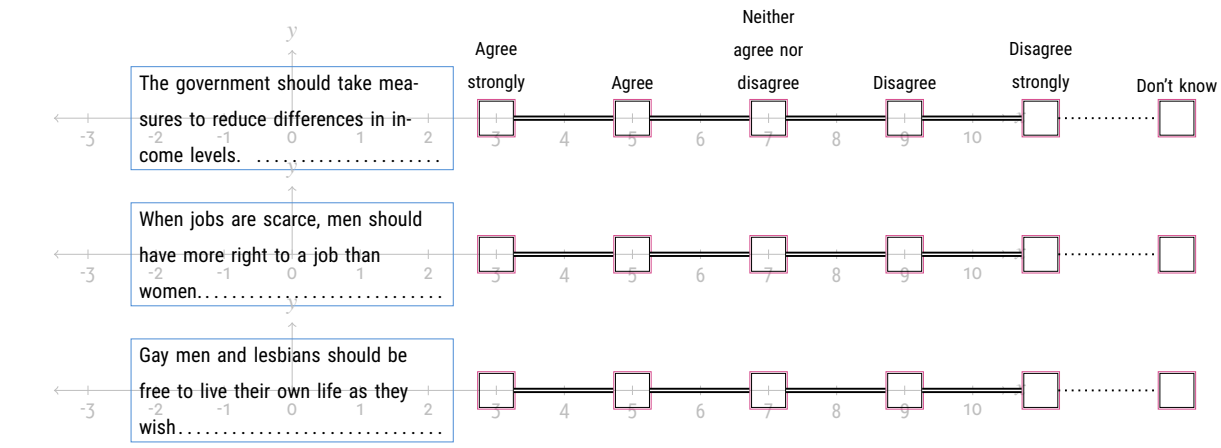


Figure 3.37 shows an example of a matrix question[27] where respondents are asked to tick one box at each row.

As you can see in Figure 3.38, you can program the connection lines for matrix questions in the same way as for single-choice questions. First, find out the coordinates of the starting (3.2,0) and the ending point (4.8,0) of the line that connects the right side of the first answer box with the left side of the second answer box.

Now you can replicate this line while moving it to the right at the same time. For this, you need to know the distance between the answer boxes (here, 2cm). With these two pieces of information (coordinates and distance), you can use the \foreach command. In this example, we want to plot four doubled lines. The loop runs through a numerical list from 0 to 3 and multiplies the x-coordinates (\x) of the lines' starting and ending point at each run.

The dotted line you can draw outside the loop. You only have to exchange the \draw option doubled with dotted and the value \x with 4 to move the dotted line into the space between the last two boxes.

[27] Variable B33, B33a, and B34, European Social Survey, 2016

Figure 3.38: Code for a Matrix-Question—Connected Answer Boxes

```
Question 3: To what extend do you agree or disagree with each of the following statements?
\textit{Please select one answer for each row!}


\begin{variable-multi}{equal01}{The government should take measures to reduce differences in income levels.}{varlab-mc, text width=4.5cm}{99}
  \begin{values}
    \foreach {\label} [count=\num from 1] in {Agree strongly, Agree, Neither agree nor disagree, Disagree, Disagree strongly, Don't know}{

        \answer{\label}{1cm+\num*2cm,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=\num}

    }

    \foreach \x in {0,...,3}{
      \draw[double] (3.2+2*\x,0) -- (4.8+2*\x,0);

    }

    \draw[dotted] (3.2+2*4,0) -- (4.8+2*4,0);

  \end{values}
\end{variable-multi}


\foreach {\label} [count=\num from 1] in {When jobs are scarce{,} men should have more right to a job than women., Gay men and lesbians should be free to live their own life as they wish.}{

    \begin{variable-multi}{equal\num}{\label}{varlab-mc, text width=4.5cm}{99}
      \begin{values}
        \foreach {\label} [count=\num from 1] in {Agree strongly, Agree, Neither agree nor disagree, Disagree, Disagree strongly, Don't know}{

            \answer{}{1cm+\num*2cm,0}{vallab-mc, font=\small}{checkbox-mc}\scoring{b=\num}

        }

        \foreach \x in {0,...,3}{
          \draw[double] (3.2+2*\x,0) -- (4.8+2*\x,0);

        }

        \draw[dotted] (3.2+2*4,0) -- (4.8+2*4,0);

      \end{values}
    \end{variable-multi}

}
```

3.6 Work with Structured Metafiles

The [survey] option of the automultiplechoice package allows to work with structured metafiles. Using structured metafiles, you can import the questionnaire information from an external file. But why should you want to do this? There might be several reasons:

No LaTeX Skills

You may want to work on a questionnaire project with people who have no LaTeX or other programming skills. If so, you can work together on the questionnaire development using a spreadsheet program.

Multilingual Survey

Or, you might want to produce different language versions of the questionnaire. In this case, you can translate the questionnaire within a spreadsheet program and import the language versions into the same TEX file without having to program the layout again and again for each questionnaire.

Online Survey

Maybe, you want to use a mixed-mode design for your survey, eventually by also generating an online survey. For this, you can import the questionnaires metafile into the online survey software Limesurvey without having to type in the questionnaire manually.

Survey Documentation

Also, documenting the questionnaire by data dictionary might be a good idea. Here, you can use your metafile and import the questionnaire's information into a TEX file that generates the data dictionary – without typing in everything manually.

Result Tables & Graphics

You can use the questionnaire metafile for labeling your result tables and graphics.

3.6.1 Metafile

If you want to use a metafile, your metafile requires a specific structure. To be interoperable with online survey software, the metafiles' structure is adopted from Limesurvey[28]. The metafile saves the questionnaire information in text format with a tabular structure. The columns are separated by a tab character, and a linebreak separates the rows. You can view this file by opening it with a simple text editor or a spreadsheet program like Excel, Libre Office Calc, or Open Office Calc. As file ending, you can choose TXT[29] or TSV[30]. We recommend using TSV if you are working on the file with a spreadsheet program. If you already have programmed an online survey with Limesurvey, you can export a TXT file and reuse this metafile with Surveyamc or ClickAMC. ClickAMC is a web application, which automatically produces the questionnaire from the metafile (https://survey.codes/click).

You can save much time and reduce the risk of transfer error if you start your survey project by generating the questionnaires' metafile, especially if you plan to produce several survey products like a paper and an online questionnaire, data dictionary, result tables, and graphics. For this, you need to know the logic of the metafile. This section describes the simplest version of a metafile with the minimum amount of information that is necessary.

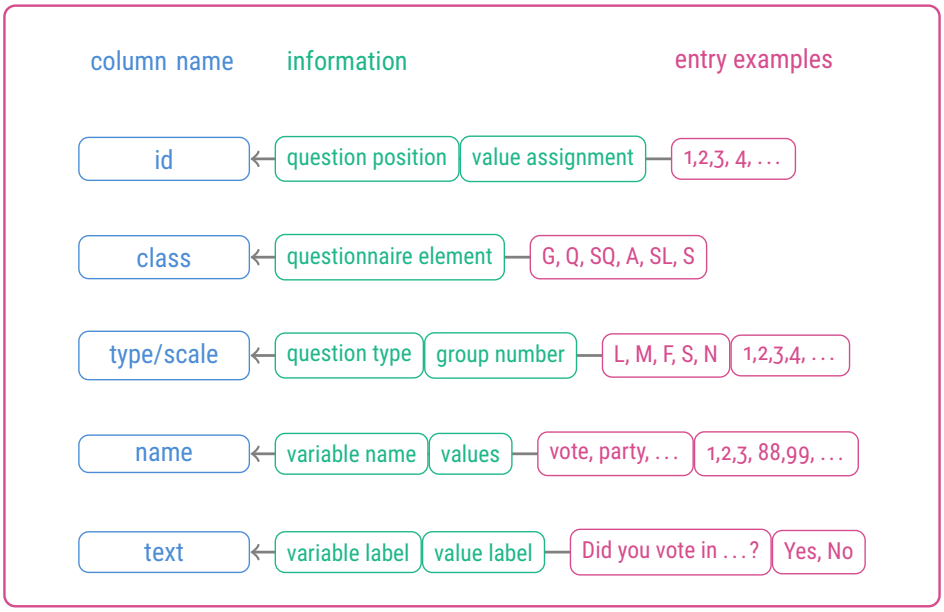
Thinking of variables, you can specify them by defining a variable name, a variable text, assigned values, and value labels. Suppose you are thinking of implementing variables in a machine-readable questionnaire. Therefore, you also need to determine the relationship between the variable information and the questionnaire elements (group, question, answer option), the question type (single choice, multiple-choice, matrix, open-ended), and the questions' position within the questionnaires' order. The metafile provides five columns for storing this information. The following Figure (3.39) shows which variable/question specification belongs to which column.

The first column stores the id, which has to be an integer. Each question group, question, and value gets a unique identifier. Within the second column (class), you define what kind of information the entries in the row are about or to which part of the questionnaire the information refers: is it a question group (G), a question (Q), a sub-question (SQ), or an answer option (A)? The third column (type/scale) you can use to define the question type or to set a number for a question group. Supported question

[28] https://www.limesurvey.org
[29] It needs the file ending TXT to import it into Limesurvey
[30] You can use TXT, TSV or CSV to import the metafile to ClickAMC: https://survey.codes/click

types are: single-choice (L), multiple-choice (M), matrix (F), open-ended numerical (N), and open-ended string (S). You can use these for both modes (Paper and Web), although SurveyAMC and Limesurvey each offer some additional question formats, which are mostly variations of the mentioned ones. Within the column name you can define a variable name or a value. The column text you can specify the variable label or the value label.

Figure 3.39: Information Assignment to Metafile Columns

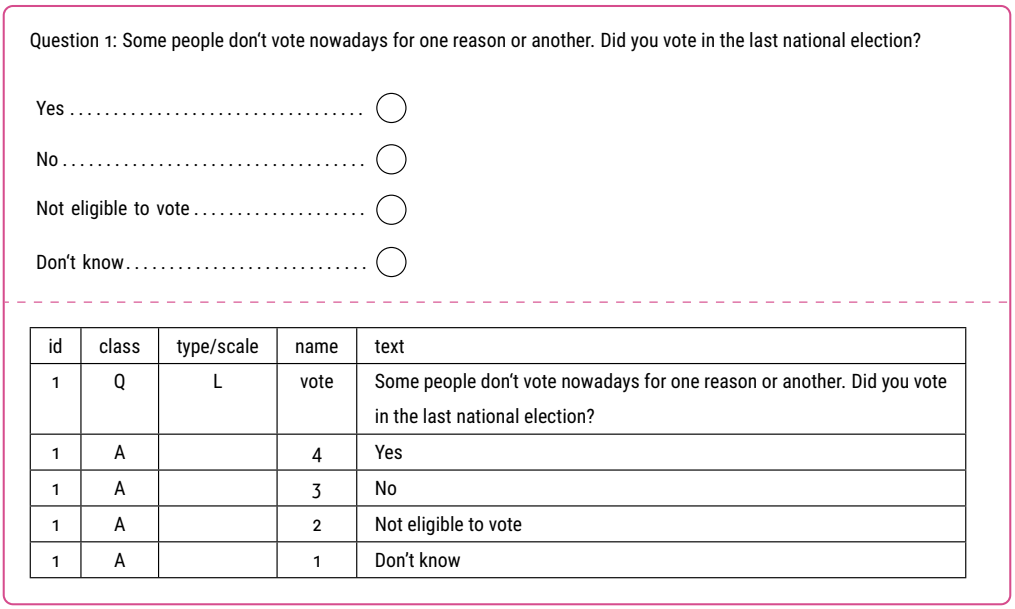


Since you now know about the most important columns' functions, the following pages provide you with some examples for each question type. Try to understand the metafile structure for each question type first (Figure 3.40 – 3.44) before taking a look at the metafile structure for the complete questionnaire example (Figure 3.45). There, you will also find some information about some specific entries that are mainly needed to import the metafile to Limesurvey.

The upper part of Figure 3.40 shows an example of a single-choice question for a paper-based survey programmed with SurveyAMC. The lower part shows this single-choice question's metafile structure as it would look like if you opened the tab-separated values file (TSV or TXT) with a spreadsheet program. Within the first row, you specify that the following information refers to a question (class=Q). You also specify the question's position within the questionnaire order (id=1), the question format to be single-choice (type/scale=L), the question name name=vote and the question text (text=Some people don't vote

for one reason or another. Did you vote in the last national election?).

Figure 3.40: Metafile Example for a Single-Choice Question



id	class	type/scale	name	text
1	Q	L	vote	Some people don't vote nowadays for one reason or another. Did you vote in the last national election?
1	A		4	Yes
1	A		3	No
1	A		2	Not eligible to vote
1	A		1	Don't know

Within the following rows, you define the answer set that belongs to this question. Each row corresponds to one answer option. By choosing class=A, you specify that the row's information belongs to an answer, and the identifier (id=1) indicates that the answer option belongs to question 1. You can define the answer option's value within the column name (e.g., name=4) and the corresponding value label within the column text (e.g., text=Yes).

Figure 3.41 shows an example of a multiple-choice question for a paper-based survey programmed with SurveyAMC and the corresponding metafile structure. For didactic reasons, we pretend that this is also the first question of the questionnaire. Again, you use the first row to specify that the following information corresponds to a question (class=Q). You can determine the question's position within the questionnaire by setting id=1. Since this is a multiple-choice question, define type/scale=M. Use the columns name and text to specify the variable name (name=improve) and variable label (text=There are different ways of trying to improve things in Germany or prevent things from …). As with single-choice questions, you can use the following rows of the metafile to define the answer options of a multiple-choice question. For multiple-choice questions, each answer option corresponds to a variable. Therefore, you need to define so-called sub-questions. For this, you can use class=SQ, which corresponds to the variable name=improve01. improve01 is the second variable of the questionnaire, so you need to set the identifier to two (id=2). The variable label here

corresponds to the answer option (text=contacted a politician). For specifying the remaining answer options proceed the same way. Unlike single-choice questions, the answer options of a multiple-choice question each require a unique identifier and a unique variable name. Finally, you can define the value for each answer option. Therefore, you choose class=A and value=1. To refer this value to the corresponding question set, you select the same identifier you have used for the main question, in this case, id=1.

Figure 3.41: Metafile Example for a Multiple-Choice Question

Question 1: There are different ways of trying to improve things in Germany or prevent things from going wrong. During the last 12 months, have you done any of the following?
Please select all that apply!

contacted a politician ..... ☐

worked in a political party ..... ☐

signed a petition ..... ☐

boycotted certain products ..... ☐

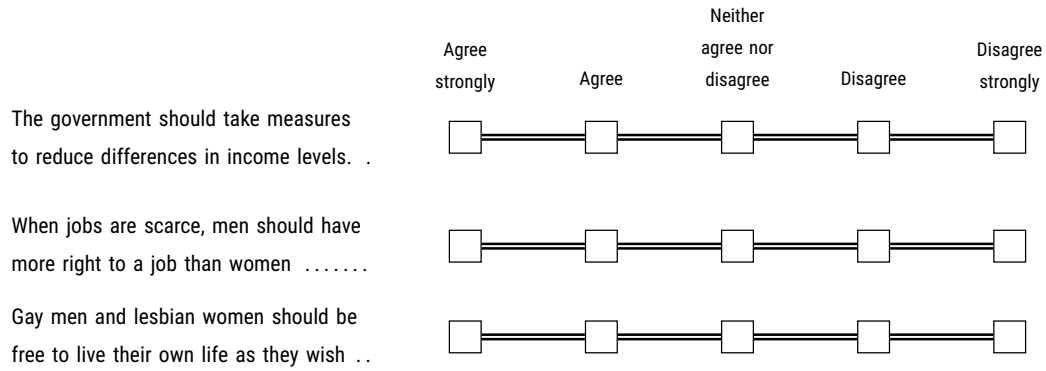
id	class	type/scale	name	text
1	Q	M	improve	There are different ways of trying to improve things in Germany or prevent things from going wrong. During the last 12 months, have you done any of the following?
2	SQ		improve01	contacted a politician
3	SQ		improve02	worked in a political party
4	SQ		improve03	signed in a petition
5	SQ		improve04	boycotted certain products
1	A		1	.

Figure 3.42 shows an example of a matrix question for a paper-based survey programmed with the [survey] option of the automultiplechoice package and the corresponding metafile. The metafile's structure for a matrix question is basically the same as presented before for multiple-choice questions. But with two differences: You need to choose type/scale=F to determine the matrix format as the question type. Furthermore, each variable of the matrix question (corresponding to one answer row) can take more values than a multiple-choice question. In this example, the answer-set (class=A) includes five values and value labels. Each value refers to the question-set (SQs) by the same identifier as the main question (id=1).

Figure 3.42: Metafile Example for a Matrix Question

Question 1: To what extend do you agree or disagree with each of the following statements?

Please select one answer for each row!

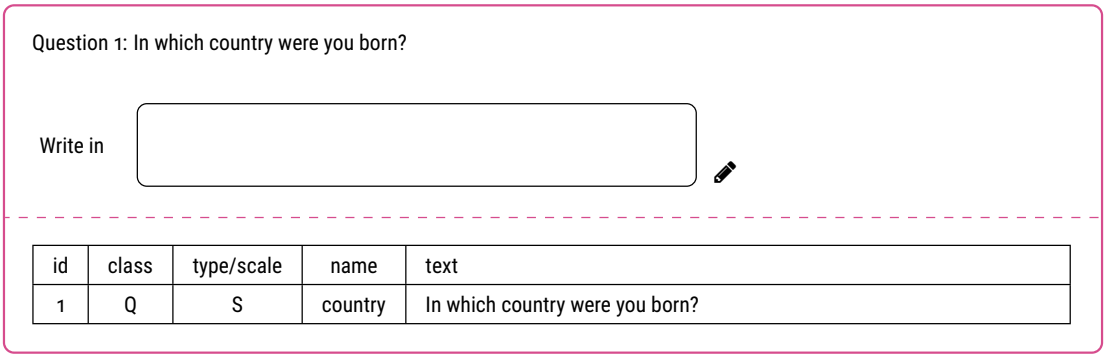


id	class	type/scale	name	text
1	Q	F	agree	To what extend do you agree or disagree with each of the following statements?
2	SQ		agree01	The government should take measures to reduce differences in income levels.
3	SQ		agree02	When jobs are scarce, men should have more right to a job than women
4	SQ		agree03	Gay men and lesbian women should be free to live their own life as they wish
1	A		1	Agree strongly
1	A		2	Agree
1	A		3	Neither agree nor disagree
1	A		4	Disagree
1	A		5	Disagree strongly

Figure 3.43 shows an example for a open-ended question[31] where respondents are asked to write in their answer in their own words (string format). The upper part of Figure 3.43 shows how you can display this question type for a paper-based survey though you can also transfer this question to Limesurvey using the metafile. All you need to do is to define one row. First, specify that the information in this row refers to a question (class=Q). Then, define the question type (type/scale=S). Choose an identifier for this question (id=1) as well as a variable name (name=country) and label (text=In which country were you born?).

[31] Variable C23, European Social Survey, 2016

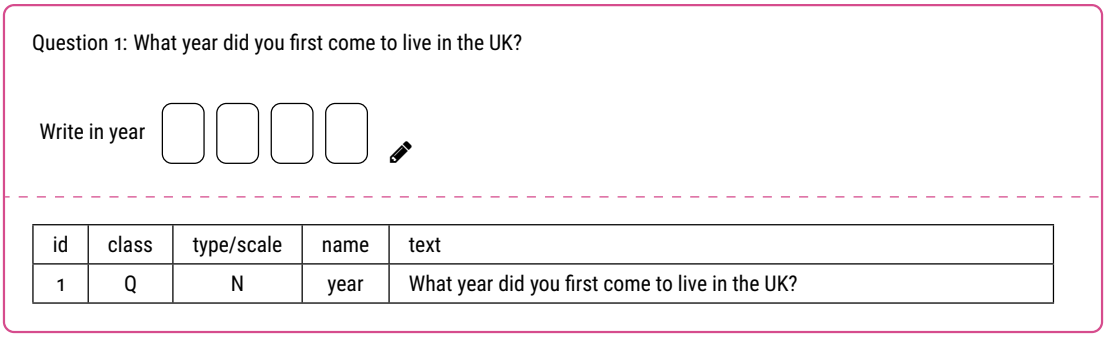
Figure 3.43: Metafile Example for a String Open-Answer Question



id	class	type/scale	name	text
1	Q	S	country	In which country were you born?

Figure 3.44 shows an example of open-ended questions[32] where respondents are asked to fill in numbers. The metafile's structure of this question type is basically the same as for open-ended questions with string characters but with the difference that you have to choose N for type/scale.

Figure 3.44: Metafile Example for a Numeric Open-Answer Question



id	class	type/scale	name	text
1	Q	N	year	What year did you first come to live in the UK?

After understanding the metafile's structure for each question type, you can put all questions together in one metafile. The question sequence within the metafile should be the same as within the questionnaire. The first rows of the metafile are required for the import into Limesurvey. With class=S, name=language and text=en, you set the language of the Limesurvey User Interface to be english. With class=SL, name=surveyls_language and text=en you determine the language of the questionnaire to be english[33]. class=SL, and name=surveyls_title is required to specify a survey title.

[32] Variable C24, European Social Survey, 2016
[33] Use de for german, or fr for french

Figure 3.45: Metafile Example for a Questionnaire

id	class	type/scale	name	text
	S		language	en
	SL		surveyls_language	en
	SL		surveyls_title	Example Questionnaire
1	G	1	.	Question Group 1
1	**Q**	L	vote	Some people don't vote nowadays for one reason or another. Did you vote in the last national election?
1	A		4	Yes
1	A		3	No
1	A		2	Not eligible to vote
1	A		1	Don't know
2	**Q**	M	improve	There are different ways of trying to improve things in Germany or prevent things from going wrong. During the last 12 months, have you done any of the following?
3	SQ		improve01	contacted a politician
4	SQ		improve02	worked in a political party
5	SQ		improve03	signed in a petition
6	SQ		improve04	boycotted certain products
2	A		1	.
7	**Q**	F	agree	To what extend do you agree or disagree with each of the following statements?
8	SQ		agree01	The government should take measures to reduce differences in income levels.
9	SQ		agree02	When jobs are scarce, men should have more right to a job than women
10	SQ		agree03	Gay men and lesbian women should be free to live their own life as they wish
7	A		1	Agree strongly
7	A		2	Agree
7	A		3	Neither agree nor disagree
7	A		4	Disagree
7	A		5	Disagree strongly
2	G	2	.	Question Group 2
11	**Q**	S	country	In which country were you born?
12	**Q**	N	year	What year did you first come to live in the UK?

For interoperability with Limesurvey, it is also important to define at least one question group at the beginning of the questionnaire. You can do this by setting id=1, class=G,

type/scale=1, name=.[34], and title=Question Group 1. It is, of course, possible to define other question groups for the questionnaire. If you do not want to display a group title, replace the text in column text with a dot character. Do not leave the cell empty because otherwise, the import of the TSV file stops at this point.

After defining the metafiles' preamble, you can adjust the identifiers within the first row id. By numbering the questionnaire elements (Group, Question, Sub-Question, and Answer), you can assign questions to a question group and answer-sets to questions.

3.6.2 Metafile Import

After you have created your metafile, you surely want to know how to import the TSV file into LaTeX to generate a questionnaire. In general, you can use the commands from the csvsimple LaTeX package[35]. Since this approach may produce much code, you may be better off using the commands provided by the [survey] option of the automultiplechoice package. On the next pages, you will see several examples of how to import the question text and different question formats. Besides, at the end of the section, you will find an example of using the same metafile to create a data dictionary. You will also find instructions on how to import the file into Limesurvey to produce an online questionnaire.

Import for Paper-Questionnaire

The following examples each show one part of the metafile (Figure 3.45) and the associated code to import the question. Let's start with some general information.

Column Names

The first thing you should know is how to address the columns. For this purpose, you can use the individual or the general column name. The individual names, in this example, are id, class, type/scale, name, text. Equivalently, you can also use the general names: csvcoli, csvcolii, csvcoliii, csvcoliv, csvcolv. For example, to address the first column, you can use either the \id command or the \csvcoli command. The third column has an individual name that is not suitable as a command name because it contains the character / that LaTeX cannot process in the desired way. Good that we can use the general name \csvcoliii instead.

[34] It is important to fill this cell with a dot character.
[35] For more information see https://ctan.org/pkg/csvsimple.

Question Text

Since you now know how to address the columns, we can start importing the question text for the first question. As you can see in Figure 3.46, you can use the question-auto environment. The only thing you have to do next is, declaring the path to the metafile and the position of the information within the file, which you want to import. There is no need to type in the question text anymore. Let's see, our metafile lies in the same folder as the TEX file, so we only have to declare the metafile's name metafile.tsv within the first argument of the question-auto environment. Using the next two arguments, you can specify a condition to filter out the cell that contains the text for the first question. In this case, the condition is if column \name contains the string vote, then write Question 1: and the content of the column \text within the same row for which the condition is true. By compiling the TEX document, LaTeX imports Some people don't vote nowadays for one or another reason. Did you vote in the last national election? exactly at the position where you have written the \text command.

Figure 3.46: Import Single-Choice Question from Metafile

id	class	type/scale	name	text
1	Q	L	vote	Some people don't vote nowadays for one reason or another. Did you vote in the last national election?
1	A		4	Yes
1	A		3	No
1	A		2	Not eligible to vote
1	A		1	Don't know

```
\begin{question-auto}{metafile.tsv}{\name}{vote}
  Question 1: \text
\end{question-auto}

\AMCboxStyle{shape=oval}

\begin{variable-single}{vote}{77}
  \begin{values-auto}{metafile.tsv}{\class}{A}{\id}{1}
    \answer{\text \dotfill}{0,\name}{vallab-sc}{checkbox-sc}\scoring{b=\name}
  \end{values-auto}
\end{variable-single}
```

Single-Choice Question

Figure 3.46 also shows how you can import the answer options for the single-choice question type. This works similarly to importing the question text, except that you need to specify two conditions. For this, you can use the values-auto environment within the variable-single environment you already know (Section 3.2). Within the first argument of the values-auto environment, you can specify which metafile you want to use, here metafile.tsv. With the help of the following arguments, you can declare the filter option. In this case, the conditions are: if column \class contains the string A and if the column \id contains the number 1, then ... Next, you can write the \answer command and import the required information from the metafile. If the conditions are true, you can use the data from the \name column (4,3,2,1) for the y-coordinate as well as for the value-assignment of each answer option (b=4, b=3, b=2, b=1). Further, you can use the information from the \text column as value labels (Yes, No, Not eligible to vote, Don't know). When compiling the TEX file, the \answer command is repeated for each line of the metafile, for which the conditions class=A and id=1 are true.

Multiple-Choice Question

Next, we want to look at how you can import the metadata for a multiple-choice question (Figure 3.47). The import of the question text follows the same procedure as described above for single-choice questions, but the answer options' import works quite differently. Nevertheless, the code structure will look familiar to you if you have read the part "Using Loops" (3.3.2) of the Multiple-Choice Section (3.3). As a reminder, for multiple-choice questions, we generate each answer option using the variable-multi environment. Now we want the variable-multi environment to be repeated for each variable that is part of the question. To do this, we can use the variable-auto environment outside of the variable-multi. Within the first argument of the variable-auto environment, you can specify the metafile you want to use, here metafile.tsv. The second and the third argument, you can use to define a condition. For this example, the filter logic is, if column \name contains the string improve, or improve01, or improve02, or improve03, or improve04, then process the variable-multi environment by replacing \name with the corresponding variable name and \text with the corresponding answer label at each run.

Figure 3.47: Import Multiple-Choice Question from Metafile

id	class	type/scale	name	text
2	Q	M	improve	There are different ways of trying to improve things in Germany or prevent things from going wrong. During the last 12 months, have you done any of the following?
3	SQ		improve01	contacted a politician
4	SQ		improve02	worked in a political party
5	SQ		improve03	signed in a petition
6	SQ		improve04	boycotted certain products
2	A		1	.

```
\begin{question-auto}{metafile.tsv}{\name}{improve}
  Question 2: \text
\end{question-auto}

\AMCboxStyle{shape=square}

\begin{variable-auto}{metafile.tsv}{\name}{improve,
                                          improve01,
                                          improve02,
                                          improve03,
                                          improve04}
    \begin{variable-multi}{\name}{\text \dotfill}{varlab-mc}{77}
      \begin{values}
        \answer{}{2.5,0}{vallab-mc}{checkbox-mc}\scoring{b=1}
      \end{values}
    \end{variable-multi}
\end{variable-auto}
```

Matrix Question

Importing the metafile's data for matrix questions works similar to the import for multiple-choice questions as described before, but with two main differences (Figure 3.48). First, you need to define a labeled answer scale for the first line of the matrix question. Second, other than for multiple-choice questions, matrix questions provide several checkboxes for each answer item. Let's start with the matrix questions' first row, which includes the answer scale. As you already know from the Multiple-Choice Section 3.3, you need the variable-multi environment. To import the variable information from the metafile, you can run the variable-multi environment within the variable-auto en-

vironment. Since you only want to plot the answer scale for the first row, you merely declare one variable, here agree01. So if column \name contains the string agree01 then run the variable-multi environment and fill in the data from the column \name as a variable name and from the column \text for the answer-option (To what extend do you agree or disagree with each of the following statements?).

Next, we want to repeat the \answer command for each value the variable agree can take on. For this, you can use the values-auto environment, which you already got know for repeating answer-options for single-choice questions. For this question here, you can set the condition, if class=A and if id=7, then use the content of the column \text for the value labels (Agree strongly, Agree, Neither agree nor disagree, Disagree, Disagree strongly) and the content of column \name for the y-coordinate (1,2,3,4,5) and the value-assignment (b=1, b=2, b=3, b=4, b=5). This information is inserted one after the other, each in its own \answer command, as long as the conditions are true.

Now you have programmed the first row of the matrix question. To generate the remaining rows, just copy the code from above and change two things: First, supplement the variable list. In this example, agree01 is exchanged with agree02, agree03. Second, delete the \text command within the \answer command's first argument since you don't want to repeat the labeled answer scale.

Open-Answer Questions

Figure 3.49 and 3.50 show examples for importing open-ended questions with string or numerical answer formats. Since the Auto-Multiple-Choice software can not automatically transfer open answers to the answer dataset, surveyamc provides no specific commands for them. Instead, you can generate the answer space by using commands from the TikZ package[36]. Figure 3.49 shows how you can use TikZ to create an open response field for a slightly longer response text. In Figure 3.50, you can see how to generate answer fields where respondents should enter numbers. Besides using TikZ for generating the answer space, you can use the question-auto environment for importing the question text as already described at the beginning of this section. For the open-answer with strings in this questionnaire example, the filter logic is, if \name=country, then write Question 11: \text (Figure 3.49). The question text of the numeric open-answer is imported by setting the condition, if \name=year, then write Question 12: \text (Figure 3.50).

[36]For more information see https://www.ctan.org/pkg/pgf

Figure 3.48: Import Matrix Question from Metafile

id	class	type/scale	name	text
7	Q	F	agree	To what extend do you agree or disagree with each of the following statements?
8	SQ		agree01	The government should take measures to reduce differences in income levels.
9	SQ		agree02	When jobs are scarce, men should have more right to a job than women
10	SQ		agree03	Gay men and lesbian women should be free to live their own life as they wish
7	A		1	Agree strongly
7	A		2	Agree
7	A		3	Neither agree nor disagree
7	A		4	Disagree
7	A		5	Disagree strongly

```
\begin{question-auto}{metafile.tsv}{\name}{agree}
  Question 3: \text
\end{question-auto}

\begin{variable-auto}{metafile.tsv}{\name}{agree01}
    \begin{variable-multi}{\name}{\text \dotfill}{varlab-mc}{77}
      \begin{values-auto}{metafile.tsv}{\class}{A}{\id}{7}
        \answer{\scriptsize\text}{2+\name*1.8,0}{vallab-mc}{checkbox-mc}\scoring{b=\name}
      \end{values-auto}
    \end{variable-multi}
\end{variable-auto}

\begin{variable-auto}{metafile.tsv}{\name}{agree02, agree03}
    \begin{variable-multi}{\name}{\text \dotfill}{varlab-mc}{77}
      \begin{values-auto}{metafile.tsv}{\class}{A}{\id}{7}
        \answer{}{2+\name*1.8,0}{vallab-mc}{checkbox-mc}\scoring{b=\name}
      \end{values-auto}
    \end{variable-multi}
\vskip1mm
\end{variable-auto}
```

Figure 3.49: Import String Open-Answer Question from Metafile

id	class	type/scale	name	text
11	Q	S	country	In which country were you born?

```
\begin{question-auto}{metafile.tsv}{\name}{country}
  Question 11: \text
\end{question-auto}

\begin{tikzpicture}
  \node[draw, rounded corners=2pt, minimum width=12cm, minimum height=2cm] at (0,0)
  {};
  \node at (5.7,-.75) {\faPencil};
\end{tikzpicture}
```

Figure 3.50: Import Numeric Open-Answer Question from Metafile

id	class	type/scale	name	text
12	Q	N	year	What year did you first come to live in the UK?

```
\begin{question-auto}{metafile.tsv}{\name}{year}
  Question 12: \text
\end{question-auto}

\begin{tikzpicture}
  \node[draw, minimum width=6mm, minimum height=8mm] at (0,0) {};
  \node[draw, minimum width=6mm, minimum height=8mm] at (.8,0) {};
  \node[draw, minimum width=6mm, minimum height=8mm] at (.8*2,0) {};
  \node[draw, minimum width=6mm, minimum height=8mm] at (.8*3,0) {};
  \node at (0.8*4,-.2) {\faPencil};
\end{tikzpicture}
```

Question Group

You may want to import information from the metafile that can not be accessed by setting only one condition—for example, the title of a question group. Since there might be several question groups, it is not sufficient to only declare `class=G`. There also needs to

be a second condition that identifies the required group. For this, you can use the group identifier that is stored in column type/scale. As described before, this column name is problematic for the use as a LaTeX command since it contains the special character /. Instead, you can use the general column name; in this case, it is \csvcoliii.

Both conditions you can declare by using the auto environment and combining it with two nested \idthenselse commands. First, specify the metafiles path (metafile.tsv) within the first argument of the auto environment. Second, start with the first condition: if class=G, then test if also csvcoliii=1, then import the content of the column \text, in this case, Question Group 1.

Figure 3.51: Import Question Group Title from Metafile

id	class	type/scale	name	text
1	G	1	.	Question Group 1

```
\begin{auto}{metafile.tsv}
  \ifthenelse{\equal{\class}{G}}{
      \ifthenelse\equal{\csvcoliii}{1}{
          \text
      }{}
  }{}
\end{auto}
```

Import for Data Dictionary

Figure 3.52 shows an example of importing the questionnaire metadata for generating a data dictionary. The commands used in this example are from the csvsimple package, and they are documented in detail in the csvsimple manual[37]. The general logic is the same as for importing the data for a questionnaire, whereas that the metafile is processed completely, like when you are generating form letters from a CSV file. For formatting the data dictionary, in this example, the additional packages ifthen and enumitem are loaded in the preamble of the TEX document. Additionally, a new counter (\questnum) is programmed that is used to enumerate the questions. Since the \csvreader command reads the TSV file sequentially, the survey title is imported first. Next, the question

[37] See https://ctan.org/pkg/csvsimple

groups are inserted, then the questions, followed by the subquestions, and finally, the answers are included. This script's code works for every metafile that is structured like it is explained in this manual. You can adjust this procedure to generate any other survey documents.

Import for Online Questionnaire

You can also import your metafile in Limesurvey software to generate a web questionnaire. The only thing you have to do is changing the metafiles' ending from TSV to TXT. The content and the structure of the file stays the same. In your Limesurvey profile, go to "Create survey". Then select the menu item "Import". A page will open, where you can upload your metafile. After importing the questionnaire, you can add filter instructions, and change the layout options by using the Limesurvey User Interface.

Automated Import for Paper Questionnaire and other Survey Products

You can also create a machine-readable questionnaire with LaTeX without having to know LaTeX yourself. How to do that? With the web application ClickAMC. All you need is the questionnaire metafile in TSV format. On the web page https://survey.codes/click, you can upload the metafile. Additionally, you can choose a layout for your questionnaire from different variants. When you press "Send", you will automatically receive the PDF document of the questionnaire and the questionnaires' TEX file so that you can still make adjustments (e.g., adding filter instructions or add open-answer fields). You will also get a Stata Do-File with which you can label your answer dataset as well as the PDF document and the corresponding TEX file of the data dictionary.

The source code of the web application is freely available on GitLab (https://gitlab.com/CSaalbach/clickamc-project). There you will find, among other things, the scripts that create the individual survey products. You can download them and use them locally on your computer. You can customize the scripts to suit your own needs. Maybe you want your data dictionary to have a different font, or you want a different color set for your questionnaire? Furthermore, you can, of course, contribute to the app via GitLab by submitting a merge request of your suggested changes.

Figure 3.52: Import for Data Dictionary

```
\usepackage{csvsimple}
\usepackage{ifthen}
\usepackage{enumitem}

\newcounter{qn}
\def\qestnum{\refstepcounter{qn}\theqn}

\begin{document}

\csvreader[head to column names, separator=tab]{metafile.tsv}{}{
  \ifthenelse{\equal{\name}{surveyIs_title}}{% SURVEY TITLE
    \LARGE\textbf{Data Dictionary: \text}
    \bigskip
  }{}
}

\csvreader[head to column names, separator=tab]{metafile.tsv}{}{
  \ifthenelse{\equal{\class}{G}}{% QUESTION GROUPS
    \Large\textbf{Gruppe \id: \name}\normalsize
  }{}

  \ifthenelse{\equal{\class}{Q}}{% QUESTIONS
    \begin{itemize}
      \item \textbf{Question: \qestnum}
        \begin{itemize}
          \item[] ID: \id | Variable: \textbf{\name}
          \item[] Text: \textbf{\text}
        \end{itemize}

    \end{itemize}
  }{}

  \ifthenelse{\equal{\class}{SQ}}{% SUBQUESTIONS
    \begin{itemize}
      \item Subquestion
      \begin{itemize}
        \item[] ID: \id | Variable: \textbf{\name}
        \item[] Variable Lable: \textbf{\text}
      \end{itemize}
    \end{itemize}
  }{}

  \ifthenelse{\equal{\class}{A}}{% ANSWER-SETS
    \begin{itemize}[leftmargin=40mm]
      \item[\textbf{[\name]}] \text
    \end{itemize}
  }{}

}
```

3.7 Generate the Answer Dataset

The SurveyAMC project is integrated in the Auto-Multiple-Choice software[38] as a option [survey] of the LaTeX package automultiplechoice. Besides the LaTeX package, the Auto-Multiple-Choice software includes a program which is able to process the scanned answer sheets into a structured datafile.

Auto-Multiple-Choice has a graphical user interface and is therefore also easy to operate without programming skills. To convey a better understanding of the features of the Auto-Multiple-Choice software, its workflow is described briefly.

First, within the software's user interface the LaTeX questionnaire is compiled. In this step it can be chosen whether individualized questionnaires should be created or if respondents should be provided with copied versions. Individualized questionnaires differ in their bar-code, which is necessary for the scanning process of the answered sheets. The option for generating individualized questionnaires enables response control, nonresponse analysis, and experimental study designs.

After printing, responding, and scanning, the scanned files are imported into the software. The answers are processed using optical mark recognition by interpreting the gray value inside the answer boxes. A high gray value is read as a ticked answer box, a small value as a box not ticked. The benchmark of the gray value can be adjusted. Furthermore, the software's interpretation can be checked manually using a graphical user interface. For each page, the answer boxes are presented in two categories ticked and not ticked. To change the automatic assignment to one of the categories, a double click on the respective box is sufficient.

After the validation, the software automatically generates a structured answer dataset, which can be exported as a comma-separated file and can therefore be analyzed with any conventional statistical software package.

For more information, you can visit the Auto-Multiple-Choice Website at https://www.auto-multiple-choice.net/. There you will find help for installation and a documentation.

[38] For more information see https://gitlab.com/jojo_boulix/auto-multiple-choice

Bibliography

Bethlehem, J. (2015). Essay: Sunday shopping-the case of three surveys. In Survey research methods (Vol. 9, 3, pp. 221–230).

Biemer, P. P. & Lyberg, L. E. (2003). Introduction to survey quality (R. M. Groves, G. Kalton, J. N. K. Rao, N. Schwarz, & C. Skinner, Eds.). Hoboken, New Jersey: Wiley Series in Survey Methodology.

Christian, L. M. & Dillman, D. A. [Don A.]. (2004). The influence of graphical and symbolic language manipulations on responses to self-administered questions. Public Opinion Quarterly, 68(1), 57–80.

Couper, M. P. (2011). The future of modes of data collection. Public Opinion Quarterly, 75(5), 889–908.

de Leeuw, E. D. (2018). Mixed-mode: Past, present, and future. Survey Research Methods, 12(2), 75–89.

Dillman, D. A. [Don A.], Gertseva, A., & Mahon-Haft, T. (2005). Achieving usability in establishment surveys through the application of visual design principles. Journal of Official Statistics, 21(2), 183.

Dillman, D. A. [Don A.], Smyth, J. D., & Christian, L. M. (2009). Internet, phone, mail, and mixed-mode surveys: The tailored design method. John Wiley & Sons. Retrieved from http://bcs.wiley.com/he-bcs/Books?action=chapter&bcsId=9087&itemId=1118456149&chapterId=103125

Dillman, D. A. [Don A.], Smyth, J. D., & Christian, L. M. (2014). Internet, phone, mail, and mixed-mode surveys: The tailored design method. John Wiley & Sons.

Dillman, D. A. [Don A], Reips, U.-D., & Matzat, U. (2010). Advice in surveying the general public over the internet. International Journal of Internet Science, 5(1), 1–4.

European Social Survey. (2016). Source questionnaire, round 8, 2016/2017. Retrieved from https://www.europeansocialsurvey.org/docs/round8/fieldwork/united_kingdom/ESS8_questionnaires_GB.pdf

Geisen, E. & Romano Bergstrom, J. (2017). Usability testing for survey research. Morgan Kaufmann.

Israel, G. D. (2010). Effects of answer space size on responses to open-ended questions in mail surveys. Journal of Official Statistics, 26(2), 271.

Jenkins, C. R. & Dillman, D. A. [Don A.]. (1995). Towards a theory of self-administered questionnaire design. Bureau of the Census.

Weiss, H. (1936). Die "Enquete Ouvrière" von Karl Marx. Zeitschrift für Sozialforschung, 5(1), 76–98.

Zhang, C. & Conrad, F. (2014). Speeding in web surveys: The tendency to answer very fast and its association with straightlining. In Survey research methods (Vol. 8, 2, pp. 127–135).

Index