Tidy questionnaire data

Author One[1*] **and Author Two**[1]

[1]Department of xxx
[*]correspondingauthor@email.com

ABSTRACT

Questionnaires are foundational tools in social, behavioral and cognitive sciences. While the past century witnessed significant advances in the design and deployment of questionnaires, comparatively little attention was paid on how to structure the resulting data. The ubiquitous current practice consists in adopting the "wide data format", where each row in a table represents a respondent and each column a specific questions, responses or other attributes. The wide format has clear advantages when applied to small datasets. However–as we show in here–when dealing with more complex datasets, this format quickly becomes impractical, error prone and difficult to reuse. Furthermore, this format prevents questionnaire research from taking advantage of modern digital technologies, multi-modal assessment and advanced data analytic methods. As a potential solution we recommend instead the long data format, and show it can handle most of the limitations posed to the wide data format. This change of format reflects a deeper change in perspectives: focusing on the person-by-question interaction as being the key observational unit rather than the person-by- questionnaire(s). It also makes it more apparent that the same data model could in principle be used to structure questionnaire data and data from computerized cognitive tests where the long format is widely used (i.e., each row in such tables typically representing a "trial" which typically refers to a person x stimulus interaction). A shared data model for behavioral and questionnaire data–as proposed by the PROJECT data model–simplifies data documentation and reuse, the sharing of data analysis methods across both types of instruments (e.g., analysis of response times in questionnaires), but also affords the development of a new generation of software tools, data analytic methods, and research questions, which will ultimately improve scientific research quality and yield novel insights about the human mind.

Please note: Abbreviations should be introduced at the first mention in the main text – no abbreviations lists or tables should be included. Structure of the main text is provided below.

Introduction

Questionnaires[1] have been a main research instrument in social sciences since the mid 19th century, with the first use of questionnaires in psychology being attributed to Gustav Theodor Fechner in 1860 (Gault, 1907). Today, the term "questionnaire" yields more than 1'7 million hits on PubMed (https://pubmed.ncbi.nlm.nih.gov/?term=questionnaire) across virtually all fields of science, testifying to its ubiquity as a data collection method.

The long history of questionnaire based research is marked by improvements in the design, distribution and analysis of questionnaires (Krosnick, 1999). Yet, questionnaires remain underrated and their full potential unexploited. Indeed, most questionnaires today appear still grounded in the pen and paper medium of the past century. Digital technologies now permit a rapid and wide distribution of questionnaires via the Internet, the collection of datasets that are not only more voluminous but also much richer than in the past. Modern technologies allow for adaptive questionnaires well beyond the simple branching structures seen today (e.g., hiding or showing questions depending on previous responses). Questionnaires could be considerably more versatile by adjusting the difficulty of the language to the reading level of the respondent or the granularity of the response options to their cognitive abilities, for example. Digital technologies facilitate the collection of rich behavioral data beyond simple answers to questions. Variables like date and time of day (i.e., timestamp), response times (i.e., how long it took a person to answer a question), changes of mind (i.e., changing responses before submitting) or geo-location are already straightforward to collect. Additional measurements, such as facial emotion recognition, gaze tracking, heart rate and blinking rate (e.g., using webcam video recordings while people fill out questionnaires)–collected separately for each question–may offer new insights and could significantly augment our understanding of respondents. Clearly, current practices in

[1]In the context of this work, we define a questionnaire as being a type of research instrument that mostly consists of questions presented to participants in the form of text and which participants are asked to respond to. The term survey, which is sometimes used interchangeably, refers to sampling data from a population, typically, but not always, using questionnaires.

questionnaire based research have not yet taken full advantage of the possibilities modern technologies offer. One of the impediments to such advancements, we believe, is the ineffective management and processing of questionnaire data which typically do not scale up and prevent automation.

The goal of this work is to contribute to a more consistent and effective use of questionnaires by offering recommendations for better structuring their data. This work is part of a larger initiative aiming to improve data practices in behavioral sciences and raise research quality (for more details, see the PROJECT website https://PROJECT/standards/).

How to structure questionnaire data?

While there are many resources on how to design better questionnaires (e.g., Vannette, 2014; Krosnick & Presser, 2009), or analyze questionnaire data (e.g., Falissard, 2012), there are surprisingly few guidelines on how to best structure and save questionnaire data (e.g., Netscher & Eder, 2018) despite the fact that there are many benefits to standardized data models (e.g., Poldrack et al., 2024; Defossez et al., 2020), including facilitating the use and reuse of data, its documentation and enabling the development of automated data processing and visualization tools. In general, standardized data structures are an integral part of scientific quality assurance.

Standardizing questionnaire data however is challenging because data from questionnaires can be quite complex. A "question" can have many different formats (e.g., text, images, sounds) and offer participants a wide range of input options (e.g., radio buttons, numeric text fields) that may lead to different kinds of data types (e.g., numeric, text, ordered, nominal). Because questionnaires typically use a mixture of question types, it can be challenging to represent all that information consistently (in a data modeling sense). Furthermore, how best to organize and structure questionnaire data depends in part on the specific use for that data and the current data processing practices. For example, online questionnaire software may use a relational database to record questionnaire data (e.g., LimeSurvey). However, such relational databases are not typically shared "as is" and are much harder to manipulate and process than the simple individual tabular data files that seem to be the default adopted by most data analysts today (Wickham, 2014)–those software solutions therefore generally offer a means to export the data as a single table or spreadsheet.

In this paper, we focus on how to structure questionnaire data to facilitate their analysis given current practices and in accord with open science principles (e.g., no proprietary data formats). We will first describe how questionnaire data is typically shared today. We will then introduce the concept of tidy data and describe its benefits. Finally, we will introduce our recommendations for structuring questionnaire data and use concrete examples to illustrate our points.

How is publicly shared questionnaire data typically structured?

The wide data format

In our experience, researchers rarely refer to standards or guidelines when formatting or sharing questionnaire data. Yet, there seems to be an implicit consensus in that questionnaire data often appear to be formatted in similar ways. More specifically, the vast majority of questionnaire data is organized in a **wide format**, where each row contains all the data from one respondent and where the columns refer to the questions, the answers to questions or to other types of information (for a toy example of such data, see Table 1.)

Table 1. Example of wide data format for questionnaire data.

subject_id	question_1	answer_1	question_2	answer_2
s001	"How old are you?"	23	"Do you like cinnamon?"	"yes"
s002	"How old are you?"	37	"Do you like cinnamon?"	"no"

Key limitations of the wide data format

The wide format for questionnaire data often has the advantage of being human-friendly when datasets are small. More specifically, because in the wide format each row represents all the responses collected from one participant, it is easy to inspect all the responses made by a given participant. Furthermore, because each question is represented as a column, it is also convenient to explore how overall participants responded to a particular question. The wide format also has the advantage of having variables that are semantically consistent and well-defined (e.g., all values within a column of the wide table are of the same type). While these are important benefits of the wide data format,

it is important to note its severe limitations. Below we present the main limitations of the wide data format before introducing the tidy data format as a better alternative.

Uncoupling of data that belong to the same questions.

It is typically the case that more than one variable is needed to describe the response to a question. For example, in addition to the actual question and a person's response to it (e.g., the choice of a particular option on a Likert scale), one might also want to record how long it took the person to respond (i.e., the response time; e.g., to evaluate if people actually read the question or clicked randomly), how often people changed their mind before validating their response, a timestamp, a geolocation tag or a code to ascertain that the collected data is genuine and has not been post-processed. When using the wide format for questionnaire data, the multiple attributes describing a given response are stored across separate columns (e.g., `q1_text`, `q1_response`, `q1_response_time`). Furthermore, there are situations where each participant may be asked the same question multiple times. In those cases, under the wide data format, one would typically append a repetition index to the variable names (e.g., `q1_text_r1`, `q1_text_r2`, `q1_text_r3`). It is easy to see then that when a questionnaire has hundreds of questions, with each question being described by a handful of attributes and the questionnaire being repeated multiple times, that the result is a very large number of columns with somewhat long and complex names.

This state of affairs is problematic for several reasons (Wickham, 2014). First, because the intrinsic coupling of the attributes about a given response is not preserved in the data structure, there is a risk to incorrectly map attributes to questions and thus to report invalid results (e.g., incorrectly grouping `q1_text_r2` with `q2_response_r2`). Second, because data is encoded in the variable names rather than in the values of a dedicated column (i.e., the `_r1` suffix in the column name as opposed to separate column named e.g., `repetition_index`), accessing and using that data is comparatively harder and requires different data manipulation processes (i.e., parsing columns names). Thus, the wide format is inconvenient, error-prone, and does not scale up.

Sparsity and missing values

When people complete multiple questionnaires, with different people possibly completing different subsets of questionnaires, the wide data format quickly becomes impractical and inefficient. Indeed, if a question was asked even to a single person in the sample, that question will generate multiple columns that need to be filled with NA values for all the participants in the table that did not respond to that question. The sparser the data, the more inefficient the wide data format will be. This issue may not be apparent for small questionnaires that are almost completely filled out by all participants, but it is obvious when combining multiple questionnaires or when questionnaires are longer and contain branching structures(Koczyska, 2022).

Furthermore, when NA values are "artificially" introduced to conform with the wide data format it can become impossible to determine for example if a given missing value results from a person not having been exposed to a question or instead having been exposed but decided not to answer. One may use different "codes" to represent different types of missing values, but such codes are not widely supported by programming languages. Thus, the wide format is inefficient and possibly misleading.

Inefficient handling of metadata

Nobody will know what the column `q2_response_r1` in the data table means without additional explanations–data without metadata is often useless. One way to provide such explanations is via a codebook, a document that lists and explains all the columns in a data table as well as all the possible values that cells within those columns can take.

Consider the case where the same question is asked on three different occasions (r1, r2, r3) and on each occasion, we record the person's response and response time. As stated earlier, under the typical wide format, we would need to encode this data using 9 columns (e.g., `q1_text_r1`, `q1_response_r1`, `q1_response_time_r1`). In addition to the mental gymnastics required to keep track of how to group the various columns, there is also quite an overload to document what these variables represent. Indeed, for each of these variables, the codebook needs to explain what they refer to (e.g., "`q2_response_r2` is the answer made to the question:"Do you like cinnamon¿', the second time it was asked. This question offers respondents two response options:"yes" and "no".) It is easy to see then that the same text needs to be repeated, almost exactly, over and over again to describe each of the columns that refer to the same question (e.g. "... the third time it was asked. ..."). In this toy example, we present only one question; in real-life examples, where questionnaires can have hundreds of questions, managing this type of metadata manually is error-prone and can quickly grow out of control.

These are, we believe, the main issues of the wide data format. Still other issues concern the difficulty to create good column names that encode information consistently when the data collection protocol is complex or the risk to have information in the column names that could identify participants and thus infringe data privacy (Netscher & Eder, 2018).

To sum up, the wide data format for questionnaire data may be convenient in a limited–although rather common–use case where participants all complete the same small set of questions once and where only their choices are recorded. However, if we are to tackle more advanced research designs and to exploit the potential of questionnaires as a research instrument, it is clear that we will quickly hit the limitations of the wide data format and that an alternative is needed. Below we present the key principles behind *tidy data* before showing how they overcome the limitations of the wide format listed above.

Tidy data

There are general recommendations for how to structure tabular data to facilitate their analysis. The concept of *tidy data* (Wickham, 2014; see also Broman & Woo, 2018) has caught a lot of traction within the R community and led to the development of a suite of elegantly designed software packages that greatly facilitate data manipulation, modeling and visualization (Wickham et al., 2019). Two points need to be stressed before presenting the key principles behind tidy data. First, there are many ways to organize a given dataset as a table. While some of these ways are clearly inadequate, others may be useful for particular operations (e.g., creating a new column by summing the values of two other columns). The tidy way is the best *default* state for that data table (for a toy example of such data, see Table 2.) Second, the tidy table is not adequate for all possible operations–it is therefore *expected* and even unavoidable that data analysts will have to reshape their data tables in various ways to serve specific purposes (for an example of sample code in R using the tidyverse package to convert the long format into a wide one, see code chunk 1).

```r
# load package
library(tidyverse)

# define toy dataset in the long data format
df <- tribble(
    ~subject_id, ~question_id, ~question_text, ~answer,
    "s001",   1, "How old are you?", "23",
    "s001",   2, "Do you like cinnamon?", "yes",
    "s002",   1, "How old are you?", "37",
    "s002",   2, "Do you like cinnamon?", "no",
)

# reshape the long data into the wide data format
df_wide <- df %>% pivot_wider(id_cols = subject_id,
                    names_from = question_id,
                    values_from = c(question_text, answer))
```

Code Chunk 1. Reshaping data from long to wide is a breeze with the `pivot_wider()` function from the tidyverse package. Note that there is also a `pivot_longer()` function for the reverse reshaping operation.

Table 2. Example of long data format for questionnaire data. Contrast this table with Table 1.

subject_id	questions_id	question_repetition	question_test	response	response_time
s001	2	1	“Do you like cinnamon?”	“yes”	1.32
s001	2	2	“Do you like cinnamon?”	“yes”	0.98
s001	2	3	“Do you like cinnamon?”	“no”	3.78

Rule 1: Each row represents an observational unit.

A core concept to organize data tables is the idea of *observational unit* which defines what the data table is about. For example, a dataset could be about students’ test scores, about air travel, or car engine performance. In each of these cases, there is an entity (e.g., student) for which we have various attributes (e.g., age, gender, grade). It is

important for structuring a table to be clear on what this primary entity is in order to ascertain what constitutes an observational unit. This observational unit determines what should be included in the table (or should instead be placed in a different table) and, most importantly, what constitutes a row in that table.

Considering questionnaire data, if one organizes the data table so that each row represents a person (as in Table 1), one implicitly makes the statement that the observational unit is at the person level (and thus that the columns in the table describe a person). Alternatively–and this is the view that we hold–one may decide that the observational unit is the "response", which is formed by the interaction between a person and a question (as in Table 2). In this case, each row in the data table would be indexed by an instance of a person x question encounter. This implies that the data from a given person will be spread over multiple rows in the data table. This way of structuring data is less common for questionnaire data but is more typical of data in cognitive computerized testing, where each row represents a "trial" (see https://PROJECT/data-model/spec/trials/).

Rule 2: Each column describes an aspect of that observational unit.

As each row represents an observation, each column represents an attribute or a variable that describes an aspect of that observation. Going back to our survey example, under the wide format, a value of "yes" on the `question_2` variable implicitly describes person "s001". Under the long format, the value "yes" on the "response" variable describes instead the interaction between "s001" and the question with `question_id==2`.

There are several points to note here. First, if a table contains columns that are *not* describing the observational unit formed by a row, those columns should be moved to a different table (see also Rule 3 below). For example, in Table 2, because the observational unit is the response, it wouldn't make sense to have a column describing the age of the participant (such a variable could however make sense in Table 1, where the observational unit is the participant.) Second, under this format, all variables describing a given observation are tied by virtue of being in the same row. This prevents the risk of incorrectly mixing descriptions from different observations.

A key property of tidy data is that each column in a data table describes a specific or "atomic" aspect of the observation. For example, `question_id` is a valid column name. However, `question_1` and `answer_1` are inadequate since these variable names contain data in their name (i.e., the `1` in `question_1` refers to the case where `question_id==1`; this `question_id` variable should have its own column rather than being "hidden" in the column names). For additional concrete examples of messy datasets and how to tidy them in R, see Wickham (2014).

Rule 3: Each type of observational unit is stored in a different table.

While it is practical to have all the data included in a single table, it is typically the case that multiple tables are needed to describe a dataset. As stated earlier, a table should contain observations that are of the same kind (i.e., describing the same observational unit); different kinds of observations should be stored in separate tables.

For example, when collecting data from participants using questionnaires, in addition to collecting their responses to each question, it is common to also collect background information such as age and gender. In this example, we have two types of observations–one describing participants, the other describing how participants responded to individual questions. Consequently, we should store this data in two separate tables–the `subject_id` variable (or "key") in both tables allows linking data across them. Note also that it is typically discouraged to have data redundancies in a relational database, because redundant data–multiple copies of the same data in different locations–has a greater risk of yielding inconsistencies across copies.

There are of course cases where one needs to combine data from separate tables (e.g., did people from different ages respond differently to a given question?) With the appropriate tooling, it is straightforward to join two tables, provided they share columns (i.e., keys; for an example of code sample joining two tables in R using the tidyverse package, see code chunk 2 and its output Table 3).

```r
# ... continued
demographics <- tribble(
  ~subject_id, ~age, ~gender, ~ses_level,
  "s001",   23, "male", 4,
  "s002",   37, "female", 8,
)

# join data tables
df_joined <- left_join(df_wide, demographics, by = "subject_id")
```

Code Chunk 2. Joining data tables with shared keys. In this example we use the `left_join()` function from the tidyverse package in R to join a demographics table to a questionnaire data table (wide format) using `subject_id` as the key to map entries across the two tables.

subject_id	question_text_1	question_text_2	answer_1	answer_2	age	gender	ses_level
s001	"How old are you?"	"Do you like cinnamon?"	23	"yes"	23	"male"	4
s002	"How old are you?"	"Do you like cinnamon?"	37	"no"	37	"female"	8

Table 3. Output of joining two tables (i.e., results from running chunk 2).

Tidy questionnaire data: advantages of the long data format

Tidy data typically has a long data format (i.e., tables have few columns and many rows). Although the long format is less common, it has important advantages that we describe next.

Effective representation of sparse data and missing values

When the data is sparse—participants are exposed only to a small set of possible questions. Representing the data in the wide format can be highly ineffective because the data table has to be filled with missing values to complete all participant-by-column cells (see Figure 1). Instead, the long data format can represent the same data in a much more compact way, recording only the participants-questions interactions that actually occurred. Furthermore, the wide data format creates a potential issue with missing values. As shown in Figure 1, the cases where missing values are injected in the data for it to conform with the wide data format needs to be distinguished from genuine missing values, for example, when participants are exposed to a question but decided not to answer it.

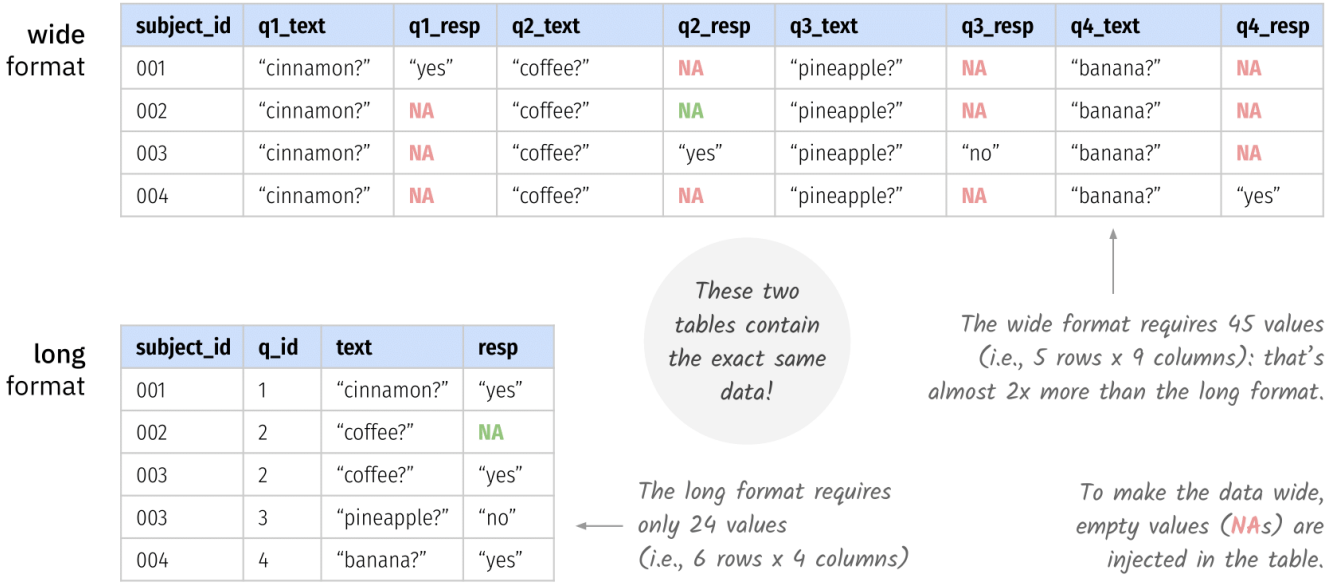

Figure 1. The long data format is more effective to store sparse data. This figure provides a concrete example: the exact same data is stored using 6 rows x 4 columns = 24 values when using the long data format, while it requires almost twice as much under the wide data format (5 rows x 9 columns = 45 values).

Selecting and filtering data

Figure 2 illustrates the fact that selecting variables related to a particular question is much harder under the wide than under the long, tidy format. Indeed, under the wide format, assuming the naming of the columns follows a clear pattern (e.g., "q1_response_r2" to refer to the response on question q1 on its second occurrence), accessing for example all the data corresponding to one particular questions (i.e., all "q1_" prefixed variables) will typically require some form of regular expression matching. This regular expression will be both specific to the column naming convention used (without which each column will need to be named in extenso, typically requiring to keep a codebook at hand at all times) and the specific subset of requested data (e.g., grabbing all data related to first

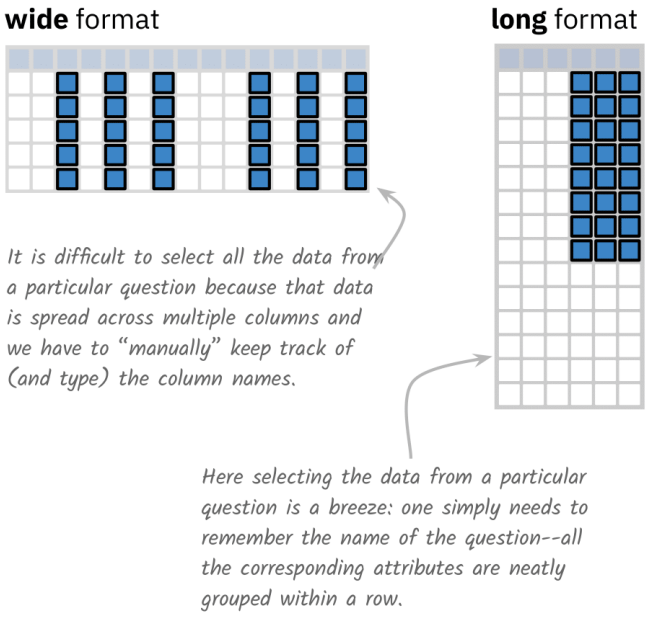

Figure 2. Selecting the data referring to a question of interest is harder under the wide than under the long data format.

time questions). In contrast, filtering data in the long data format is easier and more consistent (for an example of code in R using the tidyverse library, see code chunk 3).

```
# create toy dataset
df_long <- tribble(
    ~subject_id, ~question_id, ~question, ~answer, ~repetition,
    "s001",   1, "How old are you?", "23", 1,
    "s001",   2, "Do you like cinnamon?", "yes", 1,
    "s002",   1, "How old are you?", "37", 1,
    "s002",   2, "Do you like cinnamon?", "no", 1,
    "s001",   1, "How old are you?", "23", 2,
    "s001",   2, "Do you like cinnamon?", "yes", 2,
    "s002",   1, "How old are you?", "37", 2,
    "s002",   2, "Do you like cinnamon?", "yes", 2,
    "s001",   1, "How old are you?", "23", 3,
    "s001",   2, "Do you like cinnamon?", "yes", 3,
)

# create a "wide" version of the dataset
df_wide <- df_long |>
  mutate(question_id = paste0('q', question_id),
         repetition = paste0('r', repetition)) |>
  pivot_wider(names_from = c(question_id, repetition),
              values_from = c(question, answer),
              names_glue = "{question_id}_{.value}_{repetition}")


## Goal: get all data corresponding to question 1 or to second repetitions

# -- under wide format:
df_wide |> select(matches("^q1_"))
df_wide |> select(matches("r2$"))
```

```
# -- under the long format
df_long |> filter(question_id == 1)
df_long |> filter(repetition == 2)
```

Code Chunk 3. Selecting and filtering data under the wide vs long data format. Note that in this example, the operations remain rather simple because we used a consistent column naming scheme and tidyverse offers effective column selection tools. It is likely that when using other tools for data analysis (e.g., Excel) these difficulties will be much more salient.

Adding response attributes

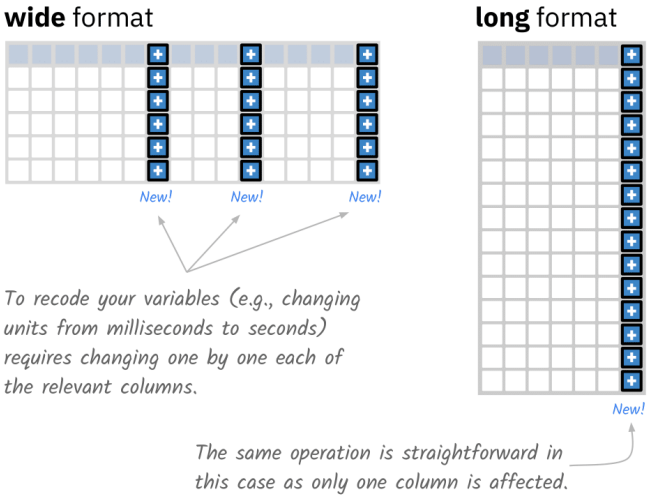

Figure 3. Changing or augmenting response data with additional attributes (e.g., response times) requires adding many columns in the wide data frame (one per response), with an increased risk of mistakes. In contrast, under the long data format, only a single additional column is needed per new attribute.

Adding more attributes to describe a response is straightforward when using the tidy format, but rather cumbersome when using the wide one (see Figure 3). For instance, assuming a 100 question long questionnaire, adding a `response_time` attribute to each response only adds a single column to the tidy data table while it adds 100 additional columns (with names like `q1_response_time`, `q2_response_time`, etc.) under the wide data format. This state of affairs is of course magnified the more attributes are included to describe a given response (e.g., `timestamp`, `is_optional`, `changed_response`). These additional attributes are critical to make the most out of questionnaire data (e.g., timestamps could be used for telemetry, identify bottlenecks and in general improve the questionnaire design; `repsonse_time` could be used to determine if a question is hard or if people are just randomly pressing keys). Because in the wide data format more columns are added or created it is also much harder to check if a column is missing (e.g., is there a timestamp for the third repetition of question 42?).

Finally, writing code to transform columns (i.e., use data to compute new variables) is much easier under the long format than under the wide format. Imagine for example that you want to change the units of the response-time variable from milliseconds to seconds. Under the long data format this is a trivial task that can be completed with one line of code. Under the wide data format however, this operation would be quite challenging, time consuming and error prone.

Metadata

Describing data (e.g., with a codebook) is much easier under the long, tidy data format than it is under the typical wide data format (see Figure 4). Assuming a questionnaire with 100 questions, 3 attributes per question and 2 repetitions, the data table would have 601 columns (100x3x2 + 1 subject_index) under wide data format, each of which requiring its own entry in a codebook (i.e., 601 rows in a codebook table) generating large files with considerable redundancies and thus a higher risk for errors. Alternatively, under the long, tidy data format, the same data would have only 6 columns (subject_index, question_id, repetition_index and the three attributes). The 100 questions could have their own codebook (i.e., 100 entries, one per question) as well as the different types

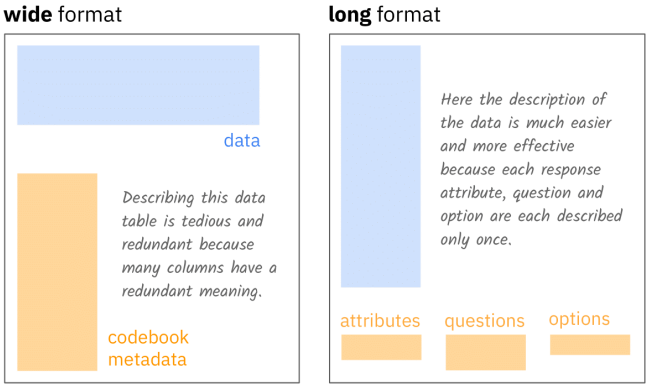

Figure 4. Metadata are more concise and easier to manage under the long, tidy format than under the wide data format.

of response options offered for each question (e.g., with only 1 entry if all the questions used the same Likert scale). Thus, in total, under the tidy data model, in our example, the codebook would have only 107, non-redundant entries, which is much smaller and more convenient to use than the 601 codebook entries under the typical wide data format.

Another important concern when using questionnaires is how to deal with multiple languages. In international projects the same questionnaires are typically distributed in multiple languages. Moreover, the language used by respondents in a study may not be the language required by the journal where the study will be published nor the language used by the scientific community that may reuse that dataset. Managing multiple languages under the wide data format can be quite cumbersome, leading to multiplication of columns and data documentation. Under the long data format, the English text could be used in the main data file together with a column indicating in which language a given question was asked. The translations of the question may then be compactly stored in dedicated Question and Option tables that can be joined using keys (e.g., question_id).

Combining multiple questionnaires

Combining data from multiple questionnaires completed by a common set of participants is comparatively much easier under the long than under the wide data format. We already mentioned the issues regarding the efficiency of data representation (see Figure 1) and data documentation (see Figure 4). The additional problem we describe here is that data from different questionnaires may use different data models, and it may therefore not be possible to join them directly: the columns and values of the data may first need to be recoded for consistency. For example, one data table may encode the question id using an index (e.g., `q1_response`) while another may use some sort of code (e.g., `sleep_response`) or the same question_id may exist in two different tables but refer to different questions. Harmonizing questionnaire data (as well the corresponding codebooks) is much easier under the long format than under the wide format because there are fewer columns to change. In fact, it is generally recommended to transform the wide data into the long format when attempting to combine data across multiple questionnaires (Koczyska, 2022).

Linking datasets

An important advantage of the tidy/long data format over the wide one relates to the idea of linking the questionnaire data to other kinds of data in a way that is both convenient and allows for that additional data to be properly formatted and easy to use. One example of such a linked data table that we already mentioned is the metadata describing the questions that were asked to participants (which could for instance include the domain, the required reading level, the emotional valence and other factors that could be used to further process the response data). If this type of data were properly formatted, it could serve as data for other types of analyses (e.g., topic analysis of questionnaire databases)–indeed, one person's metadata is another person's data (Pomerantz, 2015).

It would be desirable for questions in a questionnaire to refer to a question database so that datasets using the same questions could be identified, integrated, compared and augmented, and insights and data about a question could be effectively reused in future questionnaires and drive better questionnaire designs. Achieving this is rather straightforward with the long data format where a `question_id` could have values linking unambiguously to such

databases. Under the wide data format however, things are more challenging since column names are modified to incorporate dataset specific information, effectively breaking the link between questions and their source database. For example, a column `V1_1_2` could refer to a question in a database called `V1_1_2` or to the second repetition of question `V1_1` or the question `V1` presented as the first question to respondents the second time the questionnaire was completed.

A final example of data one might want to systematically link to questionnaire data, in particular if the questions require respondents to input textual responses, are mouse clicks and keystroke data (which keyboard key was pressed when) as this type of data may be used to authenticate participants and possibly make inferences about their cognitive processes (Conijn et al., 2019). Under the long data format, it is straightforward to connect the response data table with the keystroke data table: both tables need to have columns to refer to `subject_id`, `question_id` and possibly other variables (e.g., if the same question was asked multiple times). Things are messier under the wide data format as in that case the keystroke data need to refer to column names, where it may not be clear which column to link the keystroke data with, and where a change in columns names (e.g., in order to join multiple tables) may break[2] the links between the response data and their related data tables.

Thus, it appears that even though an analyst's main focus might be on a single tabular data containing participants' responses, the long format allows that data to be meaningfully and effectively connected to other data that may significantly augment our understanding of the studied phenomena.

On the difficulty to make questionnaire data tidy

In the previous section we demonstrated the clear advantages of the long over the wide data format. There are however some challenges related to the long format. Some readers may for instance object that the common operation of computing participant-scores (e.g., a "extraversion" score) from questionnaire data–which typically involves creating a new column by summing other columns–is much easier under the wide data format than the long one. While it is true that more operations are needed to compute such scores under the long data format, the steps are rather straightforward to implement (see code chunk 4 for an example in R).

```
# create toy dataset
quiz_data <- tribble(
  ~subject_id, ~question_id, ~question_text, ~answer, ~score,
  "s001",   "q10", "What is the capital of France?", "Paris", 1,
  "s001",   "q11", "What is the capital of Luxembourg?", "Luxembourg", 1,
  "s001",   "q12", "When was Leanoardo da Vinci born?", "1452", 1,
  "s002",   "q10", "What is the capital of France?", "Paris", 1,
  "s002",   "q11", "What is the capital of Luxembourg?", NA, 0,
  "s002",   "q12", "When was Leanoardo da Vinci born?", "1900", 0,
)

# code to score quiz responses: this code could be separate from the data analysis script and reused on ot
my_scoring_function <- function(q10, q11){
  0.8 * q10 + 1.2 * q11
}


# computing scores typically follows this blueprint;
scores <- quiz_data |>
  # reshape data to wide format
  pivot_wider(id_cols = subject_id,
              names_from = question_id,
              values_from = c(score)) |>

  # apply scoring function
  mutate(score = my_scoring_function(q10, q11)) |>
```

[2]Note that dataset specific column names require dataset specific data analysis scripts (preventing automation) and that changing column names during the analysis can break previous code (for an example, see Arslan, 2019.)

```r
  # retain only relevant columns
  select(subject_id, score)

print(scores)
```

Code Chunk 4. Example code demonstrating how to compute a score from a set of questions under the long data format in R using the tidyverse package.

A more serious concern is that it is typically not possible with questionnaire data to simultaneously satisfy the following two conditions: a) all the data is encoded in a single data table and b) the data table is strictly tidy (Wickham, 2014).

The reason why single tables for questionnaire data are not strictly tidy is mostly because responses to questions can be of different types. For example, one question may offer the response options "yes" and "no", another may propose "yes", "no" and "I don't know" and a third one may ask for a text input (where respondents might for instance type in the word "yes"). What this example shows is that the same value of "yes" has a somewhat different meaning across these three cases. Under the wide data format, this is not a problem since the type of variable is consistent within columns (e.g., the same "yes"/"no" response options have been used systematically each time question 1 was asked, hence all values of `q1_response` are of the same type). Under the long data format things are somewhat more complicated and a data model design choice needs to be made. One option is to not include the response in the main data table and instead have separate linked tables for the responses (one table per type of response; e.g., one table for "yes/no", one for "yes/no/I don't know" and one for text inputs). This relational database way of organizing the data ensures that each table is tidy. However, we end up having to use multiple tables and manage links between tables to process the data–this seems rather at odds with current data analysis practices and most statistical data analysis software today have been designed for single rectangular data tables (Wickham, 2014). The alternative option is to squeeze multiple data types into a single table. Here we see two different sub-options. The first sub-option creates a column for each type of response filling with NA all unused cases (e.g., `q1_response_yesno` would have value of "yes", while `q1_response_yesnoIdontknow` and `q1_response_textinput` would have values of NA). This solution does not seem satisfactory for multiple reasons, including in particular the need to search for the right column when analyzing a particular response. The second sub-option is to define a data model that is more abstract (columns are only loosely typed) and where additional information needs to be used to fully interpret the responses. Continuing with the previous example, a `response_description` column (with values that are all of type "text") may have the value "yes" in all the three mentioned cases, but there would in addition be a column called `option_id` which could have the values "yes/no", "yes/no/I don't know" and "text_input". This solution is more elegant and compact, but places the burden of not mixing response types on the data analyst.

The Behavese data model for questionnaires

PROJECT is a collection of integrated projects and software systems designed to accelerate, integrate and scale-up cognitive science research to yield scientific discoveries and innovation while promoting best practices and open science.

The PROJECT data model (PROJECT_ACRONYM)–a central component of PROJECT–is born out of two key realizations. Firstly, while many datasets are now publicly available (e.g., on https://osf.io/ or https://zenodo.org/), it is very hard to find data that are readily usable. In fact, many datasets may not be usable at all, even after putting in considerable effort. Indeed, some data files are incomplete, lacking documentation, weirdly formatted and recorded using proprietary data formats. Secondly, as we were running a large scale study using many different cognitive tasks and questionnaires, we realized that if we were to analyze each task as a one-shot case, it would take us a lot of time and we would end up writing code that would be hard to reuse and to maintain. Furthermore, it would be difficult to enforce common principles across tasks and datasets (e.g., how to treat outliers, whether or not to log transform response times, whether or not to filter out incorrect responses) and hence it would be hard to ascertain if specific results are due to differences in data analytic approaches or to actual differences in the data. In short, data that follow specific conventions and standards make it possible a) to reuse other people's data with minimal efforts and b) to develop standardized and reusable code.

It is important to clarify what we mean by data model to distinguish PROJECT_ACRONYM from other related projects, standards and conventions. Data has many facets and for each of them, conventions and standards can

be defined. One such facet is descriptive metadata, that is, data that describes the data of interest in a way that enables search engines to find relevant dataset based on user queries. Standards like https://schema.org/ have been developed for this purpose and can describe a large variety of things, including datasets. Nevertheless, extensions to such standards may be needed for specific use cases. This may be the case for example when users search datasets based on features that are not readily available or unintuitive in the existing standards (e.g., such as searching for datasets that involve more than a thousand participants).

The PROJECT data model does not focus on such metadata; instead it focuses on how to describe and encode the main concepts of behavioral data (such as stimulus and responses) in the data set itself, how to consistently name the features of observations (i.e., known as a controlled vocabulary), what units of measurements to use and how to organize datasets into files and folders.

A detailed description of the current version of PROJECT_ACRONYM can be found elsewhere (https://PROJECT/data-model/). Here we describe only a handful of PROJECT_ACRONYM key design choices as they equally apply to questionnaire data.

Key PROJECT_ACRONYM design choices

focus on the interaction between one person and one stimulus; The main observational unit in PROJECT_ACRONYM is the interaction between a person and an instance of a task which typically leads to a response that can be meaningfully interpreted by researchers. In a cognitive test, the task might be to decide if an arrow is pointing to the left or right by clicking one of two possible keys (these inputs define the response options). A "trial" in this context is a data construct that glues together event data that pertain to one instance of an interaction between an agent (e.g., a human participant) and a stimulus (e.g., a specific image of a left arrow, shown in a specific way). A response is one or multiple inputs or actions emitted by the agent and which form a meaningful piece of information within the context of the task (e.g., a "choice" that evaluates to correct).

In the case of questionnaires, we follow the same logic. A trial now refers to the interaction between an agent and a specific question item. A question has specific properties (e.g., its text content) and is displayed in a specific way (e.g., shown until a response is entered). Agents are offered input options to enter their responses (e.g., clicking on a checkbox, entering a text or number), and those responses can sometimes be scored or evaluated (e.g., a correct answer in a quiz).

Thus in both cognitive tests and questionnaires cases, the observational unit can be the agent stimulus interaction, resulting in the long data format. Two points are worth noting here. First, the terminology used in these two use cases tend to be quite different. For instance, the term "stimulus" is common in cognitive tests but rarely used in questionnaires, where terms like "question" and "item" are de rigueur. We prefer the consistent use of one set of terms, rather than mixing related terms; we also prefer to adopt the terms from the cognitive testing domain (e.g, stimulus, response) over the terms from the questionnaire domain (e.g., question, item) as the latter seems less general and more ambiguous (e.g., items may sometimes refer to a question AND its associated response options). Using the term "stimulus" to describe a question may feel odd at first; however, a question is in fact a stimulus and the capability to have a common data structure for cognitive tests and questionnaires outweighs in our view preferences for specific terms expressing the same idea.

The second point to note is that while in computerized cognitive tests, trials typically occur in succession (i.e., a stimulus is shown, the agent provides inputs that form a response; then the next trials starts and a new stimulus is shown), in questionnaires, it is quite common that multiple questions are presented at once to respondents (e.g., a sheet of paper students have to fill out, a "question matrix" presented on the screen where each row is a different question). This visual design choice seems necessary when questionnaires are distributed on sheets of paper; however, when using digital media it is possible and preferable to display only one question at a time as research clearly shows that this yields better data (e.g., fewer responses are skipped) and an improved user experience (Liu & Cernat, 2018; Toepoel et al., 2009). When multiple questions or trials are presented at once, the order of the questions of the sheet are used to order the trials in the data tables (and there is a construct to indicate they belong to the same page) but there is no guarantee or evidence that this was indeed the case.

keep the main information in one table; delegate further details to secondary, linked tables

A second key design choice of PROJECT_ACRONYM is to keep most of the relevant information within a central "Trial" table and delegate more specific information to linked, secondary tables. A single "Trial" table containing the most relevant data about the participant, the stimulus and the response is in line with current practices (but at

odds with relational database design principles). Specific information that might be important only in some cases or have a different data format (e.g., the trajectories of the computer mouse while answering a particular question) are stored in separate tables (e.g., a mouse trajectory table) where the rows of that table link back the Trial table via the "trial" key variable.

This solution makes it easy to analyze questionnaire data (as the main information is the Trial table) but offers a consistent way to store more detailed information for more specialized purposes (i.e., linked tables).

separate data into instrument specific files nested within subject specific folders A third PROJECT_ACRONYM design choice concerns how to organize a dataset into folders and files. It is quite common to see datasets where all the data is contained in a single file. When scaling up to thousands of participants and many questionnaires and other data generating activities, grouping all the data in a single file rarely makes sense. Instead, PROJECT_ACRONYM recommends separating the data into person or agent specific folders. Within that agent folder, data is further distributed across activity specific folders (e.g., one folder for questionnaire A, one for questionnaire B, one for the cognitive test C). This convention has several benefits. Indeed, if data from a participant needs to be deleted for some reason (e.g., data privacy request), it is much easier and safer in this case to delete all the data from one person (i.e., simply delete one folder) as in the case where data is organized into one file or in activity specific files (where it would be necessary to open all the data, search for specific entries, delete those entries and resave the data). It is also easier to write data analysis code and share subsets of data with others as selecting a subset of participant folders would constitute a meaningful subset of the data (contrary to selecting rows in a table where one is not guaranteed to have all the data from a given participant).

In addition, we recommend storing information that is about the study or common to all study participants in files and folders that are at the root level of the data set folder (e.g., the codebooks, the description of the instruments) and to use open data file formats to store data (e.g., tables are csv files).

Questionnaire data in PROJECT_ACRONYM

Now that we covered some general ideas behind PROJECT_ACRONYM it might be useful to see more concretely how questionnaire data can be described in PROJECT_ACRONYM. Again, we focus here only on the main ideas of the PROJECT_ACRONYM Trial table for questionnaires and refer the reader to the PROJECT_ACRONYM documentation for further details.

Stimulus

Stimulus refers to the actual question shown to agents (typically human respondents). A stimulus instance or presentation has many properties: some generic attributes of stimuli are stored in the Trial table (see below); other more specific attributes may be stored in a linked Stimulus table (e.g., if the stimulus is an image or a video). The main stimulus properties in the Trial table are the following:

stimulus_id is an alphanumeric string that uniquely identifies a question. This unique id can be used for instance to determine if two questions are the same (within or across datasets or studies) and to possibly access more data about that question (e.g., via a question database that provides for example a scientific reference for that question or its reading level).

stimulus_index: is an integer that tracks the order of the stimulus presentation. In the context of a questionnaire, stimulus_index = 1 would typically refer to the first question in a questionnaire and stimulus_index=23 to the twenty-third question in a questionnaire.

stimulus_description: is a textual description of the stimulus. In the case of questionnaires, stimulus_description contains the english version of the question text shown to participants (if the question was presented in different languages–which would be stated in the language column of the Trial table–the different translations would be available in related tables (e.g., the Stimulus table).

stimulus_onset: indicates how many seconds after the trial onset this stimulus was presented to agents. In questionnaires, this value is typically 0.

stimulus_duration: indicates for how many seconds the question was displayed on the screen.

Options

Options define the set of possible responses that an agent can make. For example, if a question offers the options to respond "yes", "no" or to not respond at all, then it is not possible for the agent to enter a date or a text for

example. Conversely if the option is a short text field, the agent may in principle input any sequence of say 255 characters. It is thus important to note that options are not equivalent to inputs–as to use a specific option may require multiple user inputs–and options are not equivalent to responses–they define and restrict what potential responses could be.

The main properties of options stored in the Trial table are:

option_id is a unique identifier of a specific response option. For example, a specific Likert scale offering agents the possibility to choose among 7 levels of agreement ranging from "strongly disagree" to "strongly agree" may have an option_id value of "agreement_7". Note that a given question can oftentimes be used with different options (e.g., "agreement_7" vs "agreement_5" vs "agreement_2") and that the combination of a specific question text (i.e., question_id) and a specific option (i.e., option_id) is sometimes referred as an "item". We have no explicit concept for the combination of these two ids in PROJECT_ACRONYM.

Response

Response refers to the meaning that is given to the agent's inputs and may be computed using an aggregation of event data. The main features of the responses that are encoded in the Trial table are the following:

response_description is a short text describing the response given by the agent. In the case of questionnaires, this may be the label of a chosen option (e.g., "strongly agree") or the text entered in a text field (i.e., for an open question) or even a numeric input (e.g., "42").

response_numeric is a numeric value associated with the response. In some cases, response_value is empty (e.g., in the case of text inputs); in some cases, response_value is the same as response_description (e.g., when agents entered a number); in other case, response_value refers to a numeric value that is associated to an option chosen by the agent (e.g., the "never" option may be associated with the value of 0 and "always" with a value of 1). This is a numeric interpretation of a textual or categorical variable that is independent from the question asked; it is not a "code" that is used for scoring (see "score" section below).

response_option_index refers to the index of the option chosen with the set of offered options (when these are a set like "agreement_7"). This is necessary for example when options are presented in random orders (e.g., in a quiz).

response_time indicates how many seconds it took the agent to enter their response relative to the moment where entering a response for that trial became possible (e.g., onset of the options on the screen).

Evaluation

It is often necessary to evaluate the response provided by the agent. Importantly, how to evaluate a response depends on the specific question asked and often even depends on the questionnaire as a whole. The main features to evaluate a response within PROJECT_ACRONYM are the following:

accuracy indicates with a number ranging from 0 to 1 if the response was correct.

correct indicates with a boolean value if the response is correct (true) or incorrect (false).

score is a number that indicates the value the experimenter associates to this response in this particular context. For example, in the BIS/BAS questionnaire, agents have to answer a set of questions by choosing for each one of four possible options (i.e., "very true for me", "somewhat true for me", "somewhat false for me", "very false for me"). For some questions, responding "very true of me" yields a score of 1 while for others it may yield a score of 4 (i.e., the reverse coded items) or even a score of 0 (i.e., the filler items). These item scores are then aggregated in some specific way to compute questionnaire-level scores (e.g., a BAS fun seeking score).

It is important to note that several of the PROJECT_ACRONYM concepts presented above are confounded in many current questionnaire datasets and that this can be quite problematic. For example, when responses are encoded using only a single numeric value (e.g., 2, 3 4) it is unclear if these values represent the index of the selected options, a numeric value associated to the label of the option or a score associated to a particular way of responding to that particular question. When *response_numeric* and *score* and not distinguished as concepts in the data, it may not be possible to know if the responses in a dataset were "reverse-coded" or if they represent the "raw responses"–which arguably makes the dataset unusable.

Discussion

Questionnaires are a central research tool in social sciences and their potential has yet to be fully exploited. A key impediment towards realizing this potential is a lack of standards and principles to organize questionnaire data so they can be efficiently processed, integrated, reused and shared.

Most questionnaire data today are organized following the wide data format, where each row refers to a respondent and each column to an attribute of each of the questions or responses. Although very common, and compliant with popular statistical software (e.g., SPSS), we have shown that this data format has numerous important shortcomings. To this wide data format, we prefer the alternative, long data format as it explicitly maintains attributes attached to a response, stores data more effectively, facilitates data processing (e.g., selecting and filtering) and the addition of further response describing attributes (e.g., response times) as well as simplifies data documentation (e.g., shorter, less redundant codebooks). It also makes it easier to combine data from multiple questionnaires and link relevant datasets (e.g., response data with keystroke data).

It seems clear to us that the long data format is superior to the wide one. However, because the responses participants give to each question can be of different data types (e.g., one question may ask participants to select among "yes" or "no", another to rate a frequency on a 1-7 Likert scale or to enter some text or a number), it is typically not possible for this long data format to strictly follow the tidy data principles (Wickham, 2014). Relational databases are the natural way of dealing with such data. However, current data analysis software and practices are not well equipped to analyze such data, expecting instead a small set of rectangular tables (Wickham, 2014).

The solution we propose here is a good compromise. We adopt the long data format for its numerous advantages. We also adopt the common practice of grouping the most relevant data within a single table (hence rejecting the relational database approach). These two decisions lead us to follow the tidy principles only in a loose way (i.e., columns are not strongly typed, e.g., a variable might be encoded as a float rather than a 7 level Likert scale). More detailed information about data types are however maintained in the data and can be used to strongly type, tidy subsets of the data as well as for data documentation (e.g., a codebook).

How to best organize data depends primarily on how that data is expected to be used, and by whom. The current dominant approach of organizing questionnaire data seems to be driven by two separate usages, each demanding separate design principles.

The first usage of the tabular data is the analysis, including computing averages across rows or computing sums across a subset of columns (e.g., to compute a score). Common commercial data analysis software even expects data to be in this format. There is however also a second, perhaps more implicit usage of the wide format for tabular questionnaire data: it provides an interface to the data, a means for the analyst to visually compare responses across questions or inspect the distribution of responses for a given question across respondents. This second usage is highlighted for instance in recommendations to order the columns of a questionnaire data table so they reflect the order in which questions were experienced by users (e.g., Netscher & Eder, 2018).

We believe that structuring data to support both types of usages leads to a result that satisfies neither. Instead, it makes more sense to us, on the one hand, to adopt the long data format to facilitate data storage, processing and documentation and, on the other hand, to design better data analysis software tools that take advantage of the long data format, data standards and additional files (e.g., metadata, paradata) to offer analysts better tools for exploring and visualizing questionnaire data (e.g., navigate the questions as seen by a participant together with their response and response distribution of the sample). Mixing these two usages when structuring questionnaire data keeps us stuck in a suboptimal situation and prevents progress in data processing and software development.

In this article, we covered important ideas, ideas that might be general enough to prevent this document from becoming outdated too quickly. But we did not cover everything that is required to fully structure, document and share questionnaire data (Horstmann et al., 2020; see for example, Netscher & Eder, 2018; Towse et al., 2021). Our main goal with this article was to raise awareness about the importance of the structure of the data itself and explain why certain design choices seem better than others. For the actual specification and implementation details, which are likely to evolve over time, we refer the reader to the dedicated website (https://PROJECT_WEBSITE/data-model/).

Standardized data calls for standardized processing, including automated data checks, codebooks and data visualization. Automating the boring stuff frees up more time for deep thinking and actual scientific work. We hope this work will contribute to improve research practices, collaborative research and facilitate the development of

specialized software tools (as has been the case in other fields; Poldrack et al., 2024) to the benefit of the scientific community.

References

Arslan, R. C. (2019). How to Automatically Document Data With the *Codebook* Package to Facilitate Data Reuse. *Advances in Methods and Practices in Psychological Science*, *2*(2), 169–187. https://doi.org/10.1177/2515245919838783

Broman, K. W., & Woo, K. H. (2018). Data Organization in Spreadsheets. *The American Statistician*, *72*(1), 2–10. https://doi.org/10.1080/00031305.2017.1375989

Conijn, R., Roeser, J., & Van Zaanen, M. (2019). Understanding the keystroke log: The effect of writing task on keystroke features. *Reading and Writing*, *32*(9), 2353–2374. https://doi.org/10.1007/s11145-019-09953-8

Defossez, A., Ansarinia, M., Clocher, B., Schmück, E., Schrater, P., & Cardoso-Leite, P. (2020, December 23). *The structure of behavioral data*. https://doi.org/10.48550/arXiv.2012.12583

Falissard, B. (2012). *Analysis of questionnaire data with R*. CRC Press. https://doi.org/10.1201/b11190

Gault, R. H. (1907). A History of the Questionnaire Method of Research in Psychology. *The Pedagogical Seminary*, *14*(3), 366–383. https://doi.org/10.1080/08919402.1907.10532551

Horstmann, K. T., Arslan, R. C., & Greiff, S. (2020). Generating Codebooks to Ensure the Independent Use of Research Data: Some Guidelines. *European Journal of Psychological Assessment*, *36*(5), 721–729. https://doi.org/10.1027/1015-5759/a000620

Koczyska, M. (2022). Combining multiple survey sources: A reproducible workflow and toolbox for survey data harmonization. *Methodological Innovations*, *15*(1), 62–72. https://doi.org/10.1177/20597991221077923

Krosnick, J. A. (1999). Survey research. *Annual Review of Psychology*, *50*(1), 537–567. https://doi.org/10.1146/annurev.psych.50.1.537

Krosnick, J. A., & Presser, S. (2009). Question and Questionnaire Design. In *Handbook of Survey Research* (Second Edition). Elsevier.

Liu, M., & Cernat, A. (2018). Item-by-item Versus Matrix Questions: A Web Survey Experiment. *Social Science Computer Review*, *36*(6), 690–706. https://doi.org/10.1177/0894439316674459

Netscher, S., & Eder, C. (2018). Data Processing and Documentation: Generating High Quality Research Data in Quantitative Social Science Research. *GESIS Papers*. https://doi.org/10.21241/SSOAR.59492

Poldrack, R. A., Markiewicz, C. J., Appelhoff, S., Ashar, Y. K., Auer, T., Baillet, S., Bansal, S., Beltrachini, L., Benar, C. G., Bertazzoli, G., Bhogawar, S., Blair, R. W., Bortoletto, M., Boudreau, M., Brooks, T. L., Calhoun, V. D., Castelli, F. M., Clement, P., Cohen, A. L., … Gorgolewski, K. J. (2024). The past, present, and future of the brain imaging data structure (BIDS). *Imaging Neuroscience*, *2*, 1–19. https://doi.org/10.1162/imag_a_00103

Pomerantz, J. (2015). *Metadata*. The MIT Press.

Toepoel, V., Das, M., & Van Soest, A. (2009). Design of Web Questionnaires: The Effects of the Number of Items per Screen. *Field Methods*, *21*(2), 200–213. https://doi.org/10.1177/1525822X08330261

Towse, A. S., Ellis, D. A., & Towse, J. N. (2021). Making data meaningful: Guidelines for good quality open data. *The Journal of Social Psychology*, *161*(4), 395–402. https://doi.org/10.1080/00224545.2021.1938811

Vannette, D. L. (2014, September 15). *Questionnaire design: Theory and best practices* [Workshop]. Computational Social Science, Stanford University-Institute for Research in the Social Sciences.

Wickham, H. (2014). Tidy Data. *Journal of Statistical Software*, *59*(10). https://doi.org/10.18637/jss.v059.i10

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T., Miller, E., Bache, S., Müller, K., Ooms, J., Robinson, D., Seidel, D., Spinu, V., … Yutani, H. (2019). Welcome to the Tidyverse. *Journal of Open Source Software*, *4*(43), 1686. https://doi.org/10.21105/joss.01686

Contributions

Contributed to Conceptualization: Author1, Author2

Contributed to Funding acquisition: Author1

Contributed to Methodology: Author1, Author2

Project administration: Author1, Author2

Contributed to Software: Author1, Author2

Contributed to Validation: Author1, Author2

Contributed to Writing – original draft: Author1

Contributed to Writing – review & editing: Author1, Author2

Acknowledgements

Anonymized for review.

Funding information

Anonymized for review.

Competing interests

The authors declare no competing interests.

Data accessibility statement

This paper does not include any external data.