Received: 1 November 2018 | Revised: 29 July 2019 | Accepted: 18 December 2019

DOI: 10.1111/pere.12305

ARTICLE

Personal RELATIONSHIPS WILEY

Online implementation of an event history calendar with formr: A tutorial

Larissa L. Wieczorek[1,2] | **Cyril S. Tata**[1] | **Lars Penke**[1,3] | **Tanja M. Gerlach**[1,3]

[1]University of Goettingen, Germany

[2]University of Hamburg, Germany

[3]Leibniz Science Campus Primate Cognition, Goettingen, Germany

Correspondence
Larissa L. Wieczorek, Institute of Psychology, University of Hamburg, Von-Melle-Park 5, Hamburg 20146.
Email: larissa.wieczorek@uni-hamburg.de

Abstract

Event history calendars (EHCs) are popular tools for retrospective data collection. Originally conceptualized as face-to-face interviews, EHCs contain various questions about the respondents' autobiography in order to use their experiences as cues to facilitate remembering. For relationship researchers, EHCs are particularly valuable when trying to reconstruct the relational past of individuals. However, although many studies are conducted online nowadays, no freely available online adaptation of the EHC is available yet. In this tutorial, detailed instructions are provided on how to implement an online EHC for the reconstruction of romantic relationship histories within the open-source framework *formr*. Ways to customize the online EHC and provide a template for researchers to adapt the tool for their own purposes are showcased.

KEYWORDS

event history calendar, online assessment, open source, relationship research, retrospection

As part of IARR's encouragement of open research practices, the authors have provided the following information: This article is a methods tutorial and thus was not preregistered. No data were collected. The materials used in this tutorial are available. The materials can be obtained at: https://osf.io/xcywv/.

1 | INTRODUCTION

A common goal in relationship science is to investigate events and patterns of relationship-related behavior over a long period of time. Examples of such research topics are interaction or attachment patterns across and within different types of relationships (e.g., Neyer & Asendorpf, 2001; Robins, Caspi, & Moffitt, 2002; Roisman, Madsen, Hennighausen, Sroufe, & Collins, 2001; Sprecher & Felmlee, 1997), dating and sexual behavior (e.g., Asendorpf, Penke, & Back, 2011; Brown & Sinclair, 1999; Kinsey, Pomeroy, & Martin, 2003; Penke & Asendorpf, 2008; Turner et al., 1998), or the choice of long-term partners (e.g., Campbell, Chin, & Stanton, 2016; Eastwick, Harden, Shukusky, Morgan, & Joel, 2017; Gerlach, Arslan, Schultze, Reinhard, & Penke, 2019; Park & MacDonald, 2019).

Arguably, the best way to assess consecutive data points that could speak to the topics above—and to many other topics in various research areas as well—would be longitudinal study designs (Bordens & Abbott, 2010; Schaie, 2000). However, the time, effort, and costs to conduct such follow-up studies can be enormous (Bordens & Abbott, 2010; Schaie, 1983). Moreover, there are situations where it is impossible to perform a longitudinal study and follow the individuals over time because the (start) time of interest has already passed. In such cases, researchers have to rely on retrospective data collection that is based on the participants' autobiographical memories. Unfortunately, human memory performance is known to decline with time (Ebbinghaus, 2013; Rubin & Wenzel, 1996) and to be prone to many biases or even the possibility of false memories (e.g., Belli, 1998; Hyman Jr & Pentland, 1996; Loftus, 1979; Ross, 1989; Schwarz, 2007). For example, a common finding in sexuality research is that men tend to systematically overestimate the number of their sexual partners when answering regular self-reports (Brown & Sinclair, 1999; Mitchell et al., 2018). In contrast, when it comes to negative life events such as depressive episodes or adverse childhood experiences, people (both men and women) tend to underestimate the frequency of events in retrospect (Hardt & Rutter, 2004; Patten, 2009).

Fortunately, knowledge about the hierarchical structure of autobiographical memory has increased, and techniques for the facilitation of memory retrieval have been developed (Belli, 1998; Berntsen & Rubin, 2004; Freedman, Thornton, Camburn, Alwin, & Young-DeMarco, 1988). One very useful tool for retrospective data collection is the *event history calendar* (EHC), sometimes also called life history calendar (see Freedman et al., 1988; Martyn & Belli, 2002). Today, there is a large number of studies using EHCs in a face-to-face interview (for a review, see Glasner & van der Vaart, 2009). For large-scale studies and current common use of online studies (see Bohannon, 2016; Mathur & Evans, 2005; Selm & Jankowski, 2006), however, transferring the method to online environments would be highly desirable. In this tutorial, after pointing out the advantages of EHC approaches for retrospective data collection in general and relationship science in particular, we aim to provide detailed instructions for setting up an online EHC within the open-source study framework *formr* (Arslan & Tata, 2017).

1.1 | EHCs and exemplary research questions

The EHC is a tool that was originally conceptualized as a live face-to-face interview and helps respondents to recall their past by making use of the respondents' own past experiences as cues for remembering (Belli, Smith, Andreski, & Agrawal, 2007; Martyn & Belli, 2002). During an EHC interview, respondents are encouraged to consider various personally relevant events, which are documented in a calendar grid. The events collected via the EHC serve as cues that

facilitate autobiographical memory retrieval (Belli, 1998; Belli et al., 2007; Freedman et al., 1988; Martyn & Belli, 2002). In accordance with the structure of autobiographical memory, the EHC grid and interview are organized thematically by domains and temporally by chronological ordering (Martyn & Belli, 2002). These domains are defined by certain types of events (e.g., clinical treatments) or people (e.g., dating partners). At the beginning of an EHC, it makes sense to start with the domain that will be the easiest to recall and least threatening for the participant. Common first domains include extended periodical events such as residences, educational stages, or occupations. In addition, significant political or private events, such as the election of a new government or a job promotion, can be helpful in reconstructing the past (Belli, 1998; Martyn & Belli, 2002). Figure 1 shows the grid of an example of an EHC constructed for a study investigating dating behavior. Past studies provided evidence of the capability of EHCs to improve scope and quality of data, especially regarding data as delicate as sexual behavior or substance use (e.g., Luke, Clark, & Zulu, 2011; Martyn, Reifsnider, & Murray, 2006; Morselli, Berchtold, Suris Granell, & Berchtold, 2016).

Now let us think about some research questions relevant to relationship science that could be answered with a customized EHC. When using an EHC to collect personally relevant information about different kinds of relationships, including friends, family, coworkers, or romantic partners, we could ask: Does relationship quality and attachment differ across these relationships? Where and how did participants meet their later friends or romantic partners? How long did these relationships last and who ended them? An EHC also collecting detailed information about living arrangements could track these arrangements over the course of the participants' lives and document if there are other life circumstances that coincide with specific living arrangements: How do stages of education relate to specific living arrangements? And how about professional transitions and income situations? Similar approaches are conceivable for research on sexual behavior: Does moving to bigger cities go along with having more casual relationships or one-night stands? What circumstances accompany the experience of a fulfilled sexual life? Due to its flexibility, a customized EHC may be instrumental in collecting the data needed to retrospectively answer these kinds of questions.

Half-year	99-I	99-II	00-I	00-II	01-I	01-II	02-I	02-II	03-I	03-II	04-I	04-II
Residence												
Living arrangement												
School (type and grade)												
Employment												
Important life events												
Long-term partners												
Relationship type (e.g. monogamous)												
Short-term partners												

FIGURE 1 Example of an event history calendar grid that is used during EHC interviews. Participants are asked to recall events for each year in half-year intervals, with years being represented by two-digit numbers and corresponding half-years by Roman numerals in the EHC columns

1.2 | The advantage of an open-source online implementation

As outlined above, EHCs were originally conceptualized as face-to-face interviews. However, a considerable amount of current (relationship) research is based on the use of online surveys (Bohannon, 2016; Mathur & Evans, 2005; Selm & Jankowski, 2006). Online surveys are quite popular because they possess a number of favorable qualities (Kraut et al., 2004; Riva, Teruzzi, & Anolli, 2003; Selm & Jankowski, 2006; Sue & Ritter, 2011; Wright, 2005): They allow access to large and potentially more diverse populations; sometimes, even rare populations can be reached. Moreover, online surveys are usually less expensive than laboratory or field studies, enable fast data collection, potentially save time and effort of the researcher, and may facilitate sharing of sensitive information by providing a higher degree of anonymity for the participants. To summarize, there are several good reasons why researchers decide to run studies online. We hasten to say that, even though this tutorial focuses on an EHC tailored for the assessment of relationships, any field of study with an interest in reconstructing past events or transitions may benefit from the current online implementation of the EHC. For example, criminological research could adapt an online EHC assessing the frequency and timing of violence or arrest (Morris & Slocum, 2010; Yoshihama, Gillespie, Hammock, Belli, & Tolman, 2005). In clinical and health research, the tool can be used in order to record adherence to treatments, symptoms, patterns of substance use, and other behavior that is either promoting or detrimental to health (Fikowski, Marchand, Palis, & Oviedo-Joekes, 2014; Martyn & Belli, 2002). Another example is the field of organizational psychology, where online EHCs could help in capturing employment histories, including different jobs and periods of unemployment (Belli et al., 2007; Belli, Shay, & Stafford, 2001). As summarized by Axinn, Pearce, and Ghimire (1999), the EHC method for data collection can be applied to an extremely diverse set of subjects.

It is worth mentioning that computer-assisted EHCs already exist (Belli et al., 2007; Brüderl, Castiglioni, Ludwig, Pforr, & Schmiedeberg, 2017). However, even though they are computer-assisted, these EHC adaptations still require an interviewer instead of enabling participants to complete the calendar on their own. Overcoming this limitation, Morselli et al. (2016) developed an online EHC that they used to investigate young adults' life histories, sexual behaviors, and substance use. Comparing their online EHC with a conventional online questionnaire, they found that responses obtained using the online EHC also showed higher consistency across two administrations within a 2-week time interval. Therefore, Morselli et al. provide initial evidence that EHCs can improve data quality in an online setting, too. However, the authors did not share their materials and—similar to the computer-assisted EHC versions by Belli et al. and Brüderl et al. discussed above—based their online implementation on programming languages likely unfamiliar to the average researcher.[1]

In order to combine the methodological benefits of calendar-based questionnaires and online data collection, we aimed to transfer EHCs from the laboratory to the Internet. Crucially, however, to facilitate adoption by other researchers, we also strived to avoid the use of software with rights reserved to the copyright holder and sought to base our implementation on programming languages and a modular structure that are relatively easy to share and comprehend. Hence, we implemented a solution that uses the already-existing framework of the open-source software *formr* (www.formr.org; Arslan & Tata, 2017). By using an open-source software tool, we aimed to make the tool accessible to a broad community of researchers and invite them to join its collaborative development. Moreover, as formr makes it very easy to share study materials, it may help support more transparent science in that it enables other researchers to exactly reproduce entire study designs (or portions thereof; see Kraker, Leony, Reinhardt, & Beham, 2011; Open Science Collaboration, 2015). Therefore, direct replication using the exact same methodology is facilitated.

2 | TUTORIAL: IMPLEMENTING AN EHC IN FORMR

2.1 | Study example and information about the required software

2.1.1 | Study example

For this tutorial, we will make use of an exemplary research goal, which will guide us through the steps of creating an online EHC. Imagine you would like to collect data of all partners and ex-partners a person had. Furthermore, you aim to include only participants who had at least two different romantic relationships as this enables you to carry out some within-subject comparisons. In order to support the participants during their recall of former partners and the corresponding relationship lifespans—this could mean going back in time for several years—you plan to make use of an EHC during the online assessment. This example EHC is supposed to collect information about residences, occupations, and important life events of the participants, starting from the date of their first relationship until today. The three listed calendar domains will be implemented in the study in the given order, followed by a fourth domain that collects the dates and names of the participants' partners and ex-partners. You can visit https://my-ehc-run.formr.org in order to test the study we are going to implement as a participant ahead of setting up your own EHC study.

Several research topics could be investigated following this specific data collection. Examples of such research topics will be specified below. For now, it is sufficient to keep in mind that the goal of our example EHC is to collect residences, occupations, important life events, and finally the names and relationship start and end dates of all current and former partners of the participants. Please note that every entry in the EHC will be called an *event* in order to increase readability—regardless of whether this entry describes a specific and discrete event, such as the birth of a child, or an extended episode of the person's life, such as a residence, a job, or an illness.

2.1.2 | Primer: Software and technical terms used in this tutorial

In general, this tutorial and the EHC build on different types of software, including the survey framework formr and the programming languages *R* and *JavaScript*. Of course, a good grasp of this software will be helpful in making changes to the calendar. However, no prior knowledge is required to follow this tutorial as all the steps of setting up the exemplary EHC are explained in detail, and online materials are provided as fully functional templates. In addition, Table 1 provides short definitions of all formr-specific and other technical terms used in the tutorial.

Before giving a short introduction to formr, we will take a closer look at a specific technical aspect: the visual representation of life events collected during our study. For this graphical display, we draw on something called a *Gantt chart*, which can be created using a programming language called *Mermaid.js*.

About Gantt charts and mermaid.js

Mermaid.js[2] is an easy-to-write scripting language that can be used to generate charts from text via JavaScript (Sveidqvist, 2018). The so-called Gantt diagram is a type of bar chart that was originally developed to illustrate project schedules. However, it can be used to illustrate start and end dates of events, too. Events visualized in Gantt charts created with mermaid.js can be further categorized into different domains. Depending on the input, one receives a graphical display illustrating the temporal extension of various events, which are separated by their

TABLE 1 Overview and explanations of technical terms used in this tutorial

Term	Definition
Formr-specific terms	
Item table	Spreadsheet that defines a survey's item types (see table section *formr item types)* and content. The item table is organized in columns (see table section *formr columns in the item table*).
Paging	Function in the settings of the *survey framework* that, if activated, enables the user to navigate back and forth within a *survey*. Once activated, this cannot be undone.
Run	Control structure of your study that can be edited via the *study control framework*. When creating a new run, formr automatically creates a study link based on the name of the run. A run comprises of *run units* and represents the actual study that participants participate in.
Run unit	Subcomponent of a *run* that carries out a certain task. Multiple run units can be added to a *run* to create a longer or shorter and a more or less complex study. The minimum setup of a run comprises a survey unit and a stop unit (see the formr documentation[a] for more).
Study control framework	Interface that enables you to create and edit a *run*. Beside adding and removing *run units*, the study control framework allows adding advanced styles and functions to the *run* via *CSS* and *JavaScript* in the run settings. Moreover, it lets you test a *run* before making the study link public.
Survey	Questionnaire within a study. Its items are defined in an *item table* and uploaded or updated via the *survey framework*. One or more surveys can be added to a run as survey run units. The survey name can be used to refer to that survey or answers given to items within it.
Survey framework	Interface that enables you to upload an *item table* that becomes a *survey* and to control some survey settings, such as paging. For each *survey*, formr creates a result table that can be exported in different file formats.
Utility R package	*R* package that is helpful when working with data collected via formr. The package is independent of the *survey framework* and the *study control framework* and is not required to run a study but is a useful tool for data organization and processing in *R*.
formr columns in the item table (see the formr documentation[a] for more)	
$Choice_1$-$Choice_n$	Defines the text of answers that a user can choose in multiple choice items. In your *item table*, you need as many choice columns as answers of the item with the most answer options. In the results of a *survey*, the user's choices are saved as numbers that correspond to the number of its choice column.
Class	Allows adding *CSS* classes in order to visually style the corresponding item in a certain way. Some classes are predefined in formr, and others can be created in the *CSS* settings of the *study control framework*.
Label	Defines the item text that is displayed. Usually, it is shown on the left-hand side of the answer choices or text field.
Name	Assigns a name to each item. User answers will be stored under that variable name, and it allows you to refer to an item. The item name needs to be unique, that is, a name cannot be used twice within the same *item table*.
Optional	Determines whether an item is optional. Most items are mandatory by default. By using "`*`" in this column, you can turn items optional instead. Using "`!`" requires a response to items that are optional by default.

(Continues)

TABLE 1 (Continued)

Term	Definition
Showif	When empty, the corresponding item is shown to all users. Alternatively, you can use this column to define a condition with logical operators via *R* code. As a result, the item is only shown to users for whom the condition is TRUE.
Type	Defines the identity/function of an item (see *item types* for the types used in this tutorial).
Value	Allows you to preset a value. This can help to make a survey more user-friendly and set those answers as a default that are most likely. Furthermore, the value column can be used to define functions of a *calculate* item.
formr item types (see the formr documentation[a] for more)	
Calculate	Item that stores results of calculations that are specified via *R* code in the *value column*. As it is a hidden item, values are stored when the user does not see the item nor needs to react to it. Calculate items are very useful to access external data or to calculate user-specified values that you want to refer to later. Please note that you cannot refer to a calculate item on the same page where it was created. When based on values from the same *item table*, the calculate item needs to be followed by a *submit item* before you can refer to it for the first time.
Counter	See *number*. Please note that a counter is no stand-alone item type.
MC_multiple	Item with a check box for every answer that is defined in one of the *choice columns*.
Month	Item that is displayed as a calendar grid and allows the user to pick a certain year and month. Adding parameters (e.g., "month `2 years,now`") constrains the date range from which dates can be picked.
Note	Item that displays text. It can also be used to include pictures or graphics produced via *R* code in your survey.
Number	Item that allows the user to enter a number. Adding parameters (e.g., "number `1,100,1`") constrains the range of available numbers with the first two values and defines the steps with which the numbers increase with the third value. *A number* item can be turned into a button that counts how often the user clicked on it by adding the *CSS* class *counter*. To do so, the word "counter" needs to be inserted into the *class column* in the row of a *number* item within the *item table*.
Submit	Item that is displayed as a button at the end of a survey page. It saves all answers given on that page and leads to the next page or to the survey's end.
Text	Item that allows the user to enter a text in a single-line input field. Adding a parameter (e.g., "text `100`") defines the maximum number of characters that may be entered.
Other software and technical terms	
CSS (cascading style sheets)	Language that describes the style of an *HTML* document. Its code defines classes that change the look of all elements they are assigned to.
Gantt chart	Bar chart that was originally developed for project management. It displays the temporal order of tasks, events, or activities.
HTML (hypertext markup language)	A standardized system for developing worldwide web pages.
JavaScript (JS)	An object-oriented computer programming language commonly used to create interactive applications within web browsers.

(Continues)

TABLE 1 (Continued)

Term	Definition
Mermaid.js	A JavaScript library that can be used to generate and display *Gantt charts* on web pages.
OpenCPU	Framework for embedded scientific computing and reproducible research. It provides an interface that allows users to execute *R* code in an online environment and to combine it with other programming languages such as *JavaScript*.
PHP (hypertext preprocessor)	A scripting language that can be used in developing dynamic web applications.
R	Open-source software for statistical computing and graphics.

Note: Within each section, terms are sorted alphabetically. Words in italic font refer to a term that is defined in this table as well.
[a]www.formr.org/documentation

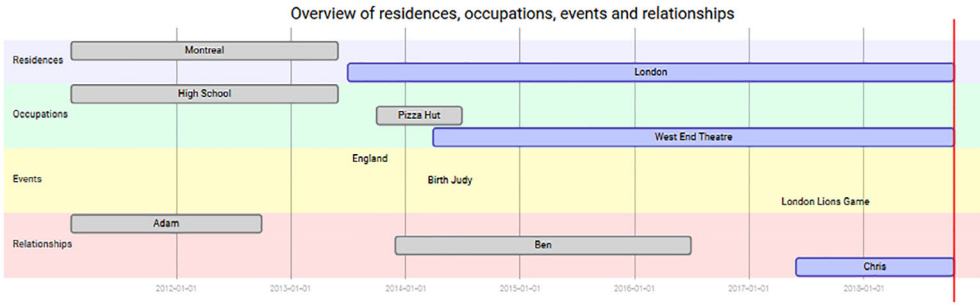

FIGURE 2 Example of a Gantt diagram, generated from fictitious information that could have been collected within the event history calendar section of a survey. Current events have blue bars, and past events have gray ones. The fictive participant first lived in Montreal, where she visited high school. Now, she lives in London and works at the West End Theater after a short episode of working for Pizza Hut. Her current partner is Chris; he is the third partner after Adam and Ben

domain but may refer to the same time frame. In this way, the temporal overlapping and order of events becomes visible, whereas clarity is ensured due to the separate sections for every domain. Figure 2 shows a Gantt chart created from fictitious data as it could result from a participant of our study example.

About formr

formr is an open-source software developed by Arslan and Tata (2017).[3] It provides an online framework to set up online studies of varying complexity. In this tutorial, we only give a quick overview of the software's general structure and use. More information about formr can be obtained from the formr homepage, www.formr.org/about. For detailed documentation on how to use formr above and beyond the setup of an EHC as outlined in this tutorial, please visit www.formr.org/documentation and read Arslan, Walther, and Tata's (2019) article explaining the software.[4] formr features three main components, namely, a *survey framework*, a *study control framework* (Arslan et al., 2019), and a *utility R package* (Arslan, 2018). The survey framework provides a regular questionnaire interface and allows researchers to pose questions to users and collect the corresponding responses. The items of a formr questionnaire or survey are specified in an *item table*, which is simply a spreadsheet that can be uploaded to the survey framework. The spreadsheet can

be created locally as a Microsoft Excel file, OpenDocument spreadsheet, comma-separated value (CSV), or tab-separated value (TSV) file, or it can be created online using Google Spreadsheets.[5] The use of Google Spreadsheets is recommended, mainly because it enables researchers to collaboratively create studies online and facilitates rapid reloading of newer versions of the item table when debugging. Nevertheless, we will work with local Excel files in this tutorial because there is no surety that Google Spreadsheets will remain the same over the next years. The study control framework of formr allows researchers to administer a study when performing complex tasks. Such tasks include managing access to a study, sending out reminders via email or text messages, giving feedback to users, writing R scripts for data analysis and overviews, or customizing who answers which questions and when. Although the first component, the survey framework, is provided by most other software out there, the control framework is unique to formr. Finally, formr is accompanied by a utility R package[6] (Arslan, 2018). It is independent of the survey and control framework but helps to organize the collected data or to conduct further on-site analysis by providing R helper functions.

The study and control frameworks of formr are written in the programming language *PHP*. Due to the additional integration of a framework called *OpenCPU* (Arslan et al., 2019; Ooms, 2014), researchers are able to control various aspects of their study using R code. As noted before, no programming skills are required to use the basics of formr or to create the EHC introduced in this tutorial, but different kinds of coding can be applied in order to customize a study implemented in formr. Researchers familiar with R can use their favorite R packages to create dynamic survey texts, generate user feedback, or control further study aspects. In addition to the flexibility provided by R, researchers can further change the look and feel of their questionnaire by specifying custom cascading style sheets (CSS) and increase usability by writing custom JavaScript for their study. As will be shown further below, the latter is necessary to integrate mermaid.js with formr when creating an online EHC. In addition, studies created with formr can easily be combined with other survey software: As it is possible to use the study control framework without the survey framework (but not the other way around!), formr studies can also be integrated with other software. Thus, if you would like to conduct an EHC study but do not want to go without your favorite and familiar survey software, you could just set up a formr-based EHC as explained in this tutorial and add in questionnaires that are based on another online survey software before and after it.[7] Due to its ability to combine different programming languages and ease of use, formr is a very powerful tool to create a variety of innovative and complex online studies.

2.2 | Steps of implementation

In the next section, you will learn how to create an EHC that builds itself up based on the information a participant provides. In order to get an idea on how the EHC appears within an online survey, Figure 3 illustrates how the calendar displays the data from the first domain at the beginning of the survey and becomes bigger with every domain and event the participant lists. We will start with an explanation on how to create a formr account and then look at some considerations that are useful when planning an online EHC. Finally and with the support of the survey materials that are available online, we will go through the steps of implementing the calendar by combining formr with mermaid.js.

As technology changes from time to time, we cannot be sure that this tutorial will stay current forever. Therefore, we added an online wiki at https://osf.io/xcywv/wiki/home/ that contains a brief up-to-date version of all steps explained next. In addition, the provided materials will be replaced by new versions if necessary. For that reason, discrepancies between the code explained within this article and the code actually used might occur at some point after publication of this tutorial.

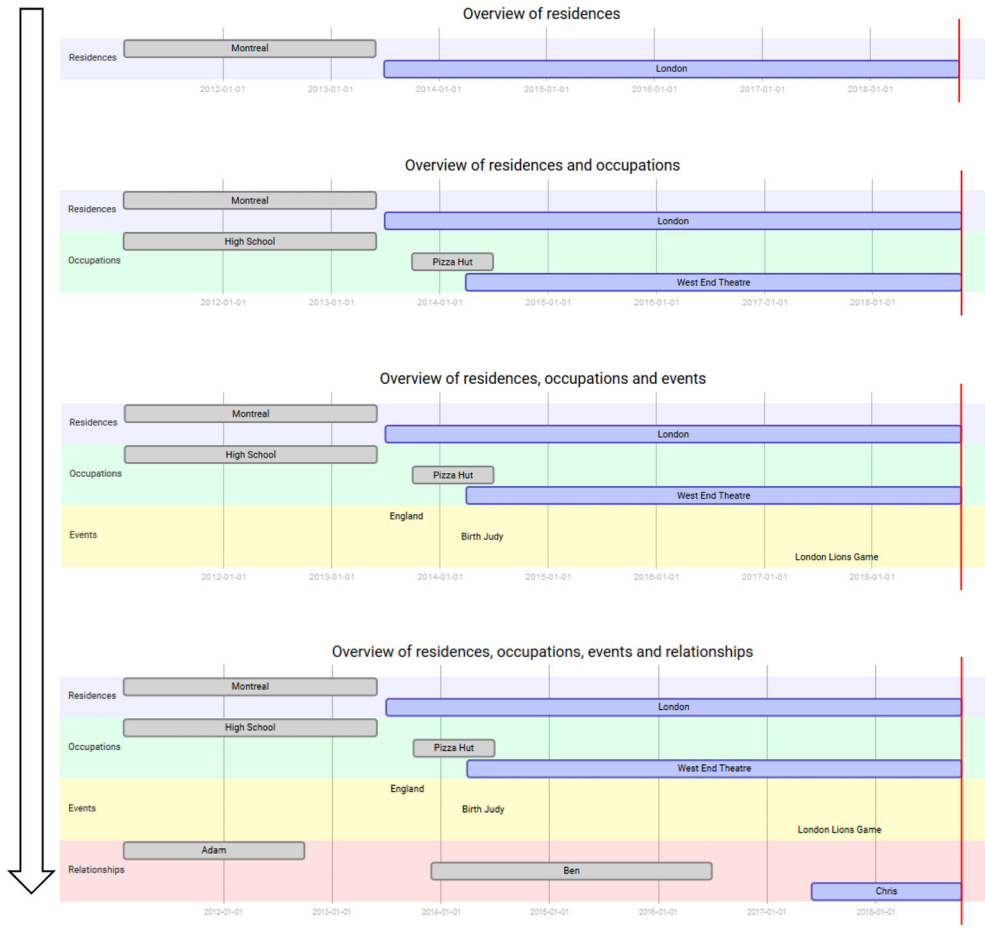

FIGURE 3 Flowchart illustrating the continuous setup of the event history calendar: With every survey page a participant completed, a new domain is added to the Gantt chart

2.2.1 | *Step 1*: Creating a formr account

In order to use formr and its extended features, you will first need to obtain an administrator account. To do so, visit https://formr.org and click on the *Sign up* link. Enter your email, choose a password, and click on the *Sign up* button. If your registration was successful, you should receive an email confirming your registration. To complete the registration process, you will need to send a request to formr administrators to grant you administrator rights: Simply send a short message to one of the email addresses that are listed after going to *Documentation* → *Getting Started* in formr. Once the rights are granted, your account is ready to set up your study.

2.2.2 | *Step 2*: Considerations in advance

Before implementing the EHC in your online study, you need to think about the following:

- Which are the event "domains" you want to ask the participants about? Depending on the variables of interest, this could be anything. As Martyn and Belli (2002) note, only the most important domains should be included on the EHC as large amounts of data are being collected. You should also think about the order: The EHC should start with a very general domain that is easy to remember and end with the most specific one, which often contains your variable of interest. In our example, the domains in their order are *Residences*, *Occupations*, *Events* (the participant can list any event that was personally important for her or him), and finally *Relationships*.
- How many events can a participant enter per domain? The number of possible events needs to be predefined. Think about a number of events that makes sense to you: On the one hand, participants should have the possibility to provide much information. On the other hand, too many episodes in the diagram will look crowded.[8] For the *Residences* and *Occupations* domains in our example study, participants will be required to name at least one event each. In contrast, listing events in the *Events* domain will be optional. Finally, participants will be asked to list at least two partners in the *Relationships* domain.
- What temporal categorization is sensible for your data collection? Should the participants date their events exactly to the day (which is usually utopian when collecting data retrospectively), or is it rather practical to ask participants to indicate the week, month, quarter, or year of an event? In our example, participants will be asked to indicate the month and year in which events started and ended.

2.2.3 | *Step 3*: The survey spreadsheet

As outlined above, surveys in formr are designed using spreadsheets. In such an item table, one row specifies an item via multiple *columns* with certain functionalities (see Table 1 for an overview of columns that organize the item table and *item types* that are used within this tutorial). For this tutorial, we will use an Excel spreadsheet that already contains all item definitions that are required to replicate our exemplary study. All files used in this manuscript are hosted using the open science framework (OSF). Download the spreadsheet from https://osf.io/afzwt/ and save it locally on your computer. The file's name might not be changed as formr creates the *survey name* from it, and the code in the JavaScript settings (which will be explained later) refers to this name. Changing the spreadsheet's name would result in an inoperable study unless the R and JavaScript code snippets are modified accordingly. If you prefer to work with Google Spreadsheets, you can clone the spreadsheet in your Google account. The provided spreadsheet contains a column with some explanations to provide some orientation within the sheet. We will now take a look at the various items in the spreadsheet to understand how they contribute to the buildup of the EHC.

Remember that an EHC is organized thematically by domain and temporally by chronologically ordered events (Martyn & Belli, 2002). In our online implementation, every event collected in the calendar requires (1) a title (name of the event), (2) a start date, and (3) an end date or (4) an indicator specifying if the event has not yet ended. There should be the possibility of adding more than one event per domain, but a cap on the number of events needs to be specified.

Define the earliest date

Before we define the different domains, we need to define at what date participants can start to add events. Please open the downloaded spreadsheet: After a general instruction on the EHC, the item of the type *month* in row 3 requests the date of the participants' first romantic relationship. The lower and upper limit of the *month* item definition validates that users can choose dates not older than 20 years ago and not younger than the current day. In row 5, a *calculate*

item with the name "first_date" stores the year and month that were indicated before via a function in the *value* column. Later, this information will be used dynamically to define at what date participants can start to add events. Thus, based on the respective start of the first romantic relationship, each participant's EHC will request events starting from that date.

Creating the residences domain

As you can see in the spreadsheet, the instruction in row 8 refers to the "first_date" variable that we created previously via R code. When a participant opens the corresponding survey page, she or he will see the indicated date of the first relationship instead of the R code. Continuing with the spreadsheet, let us take a look at rows 9-50 concerning the *Residences* domain and compare them with Figure 4 for an explanation.

Figure 4 illustrates how the items of the *Residence*s domain are defined within the spreadsheet. As can be seen, four items are defined to collect the information (1)–(4) that is required to describe one event (in this case, one residence) as delineated above. In formr parlance, this translates to (1) an item of the type *text* to enter a residence; (2) an item of the type *month* to enter a start date; (3) a second *month* item to enter an end date; and (4) an item of the type *mc_multiple* to indicate an event as ongoing, that is, not ended. The *month* item for the end date and the *mc_multiple* item for current events are optional (as marked by "*" in the *optional* column) because these two are mutually exclusive in terms of content: An event has either ended and thus has an end date or is ongoing, but never both.

As noted before, the number of events a participant can enter needs to be predefined. To predefine the number of events that a participant can list within the domain, the four mentioned item rows need to be repeated for every potential event entry. In our case, we decided to limit the number of events to 10 (see spreadsheet). This means that the four items in the spreadsheet with the names "address_R1," "datefrom_R1," "dateto_R1," and "current_address_R1" defining one event need to be repeated nine times. Note that the item names in the *name* column of the spreadsheet need to be adapted when repeating the rows (e.g., "address_R1," "address_R2," "address_R3," etc.) because formr does not accept two items with the same name.

Even though we have predefined 10 events, a participant should only see and need to fill out as many residences as she or he has had. Thus, we need a mechanism that controls how many item groups are displayed to the participants at a time. In the spreadsheet, we use the *showif* column to hide the items (rows) not needed to be shown and add an item of type *number* with the CSS class (as defined in the *class* column) called *counter* at the end of the item definitions in row 49. formr translates this *number* item with the special class into buttons that can be clicked by the participant to add or remove item rows. This *counter* needs a default value in the *value* column to work. This value dictates "how many event rows" are shown to the participant by default. In our example, its default value is "1" because every participant needs to indicate at least one residence. As you can see in the *showif* column, all events after the first one are assigned a *showif condition*, defining that the set of items for a further event is only displayed if the *counter* variable exceeded a certain value. In practice, this means that the items collecting the title and dates of a further residence are only shown if the participant requested to add a new one. Moreover, note that, if you would like to predefine more or fewer events, you would need to adapt the number of repetitions of the four spreadsheet rows defining one event, as well as the upper limit of the *counter* variable (lower and upper limit are defined by values next to the *number* item in the *type* column). Finally, every survey page needs to end with a *submit* item, a button that permits the participant to save his or her responses and move on to the next page of the survey. Figure 5 shows how the items defined in the spreadsheet are displayed to the participant.

Rows 51 and 52 of the spreadsheet define a validation mechanism for the *Residences* domain. It checks if all residences have labels and if the move-out date is always greater than the move-in

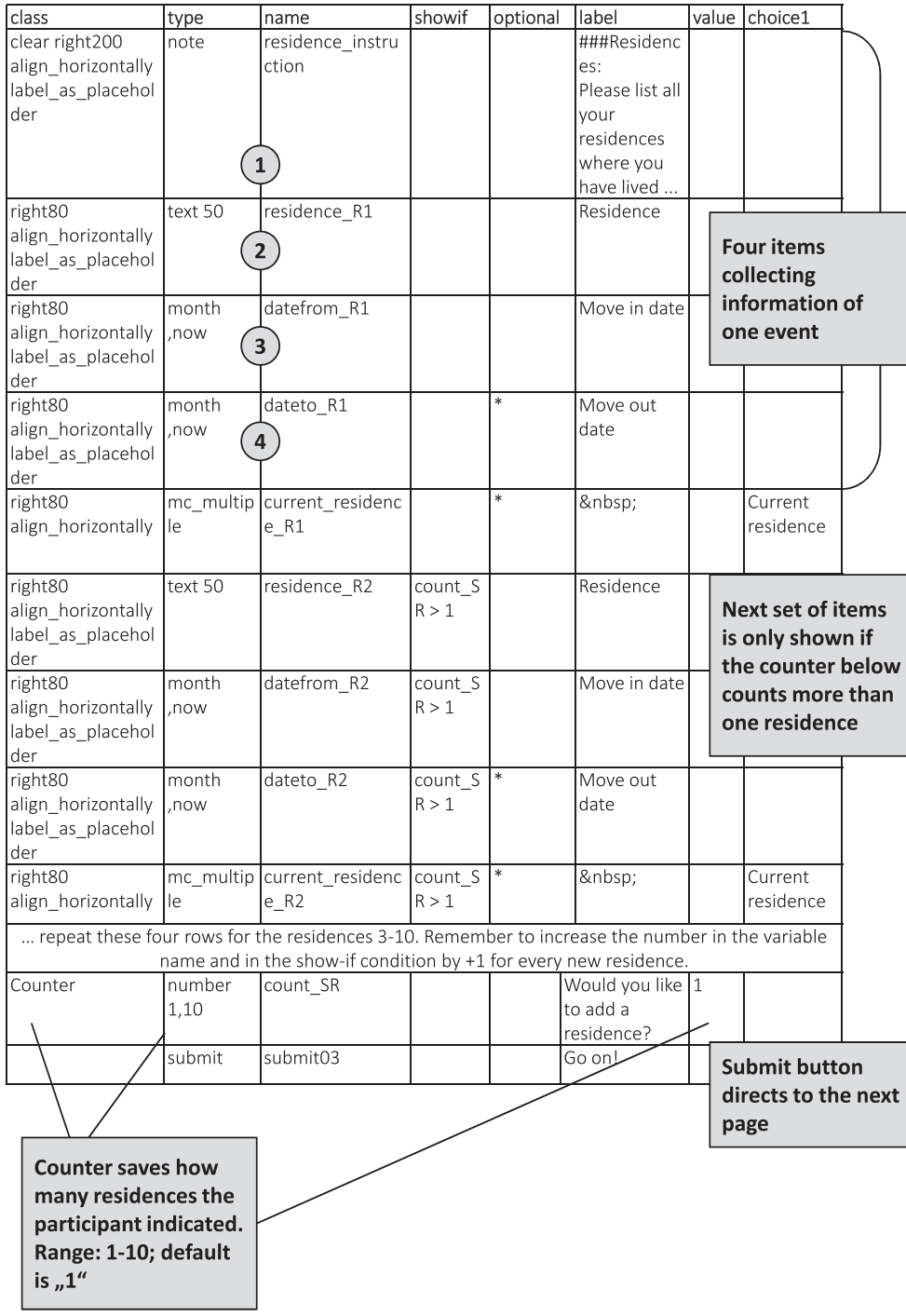

FIGURE 4 Spreadsheet defining items for the domain *Residences*. Participants will be allowed to enter a maximum of 10 residences

date. Row 53 defines a formr *note* item that presents the calendar to the user for this domain. In order to display the Gantt chart that we use as our EHC, we need to insert R code into the *label*

column of this row. A so-called *chunk* of R code starts after the characters “”{*r*}” and ends with “””. Thus, text written within such a chunk is read as R code by formr. In R, it is possible to define custom functions. In the *label* column of row 53, an R function that aggregates data entered by a participant (on her or his residences) is defined. The following R code creates the relevant text structure required by mermaid.js to draw up the EHC. This structure is read internally by JavaScript as specified in the study settings (more on this later). The resulting calendar graphic, generated from the input of the *Residences* domain and exemplarily displayed in Figure 6, will be presented to the participants when processing the next domain on a new survey page. The full R code, together with short explanations in comments highlighted in black, is given below.

```
```{r}
# Function that collects an event's title, start date, end date, and
the information if it is current; the collected information is pasted
in the format that can be read by mermaid.js
gannt_event_row <- function (i, data, col_title, col_datefrom,
col_dateto, col_current) {
  empty <- ''

  if(!(paste0(col_title, i) %in% colnames(data))) {
    return (empty)
  }

  title <- last(data[paste0(col_title, i)])
  if (is.na(title) || title == '') {
    return (empty)
  }

  is_current <- last(data[paste0(col_current, i)]) == 1
  datefrom <- last(data[paste0(col_datefrom, i)])
  dateto <- ifelse(is_current, format(Sys.Date(), "%Y-%m-%d"),
last(data[paste0(col_dateto, i)]))
  tag <- ifelse(is_current, "active", "done");
label <- paste0("a", i)

  paste(title, ":", tag, ",", label, ",", datefrom, ",", dateto, " ",
sep = " ")
}

# Vector that stores all information of the Residences domain: The
function specified above runs in a loop for every entry of a
participant
residences <- NULL;
for (x in 1:last(My_EHC_survey$count_SR)) {
  residences <- c(residences, gannt_event_row(x, My_EHC_survey,
'residence_R', 'datefrom_R', 'dateto_R', 'current_residence_R'))
}

# Below, the text input for mermaid.js is given; it includes the vector
with the information of the Residences domain
```
<div class="formr-mermaid">

Gantt
  title Overview of residences
  dateFormat YYYY-MM-DD
  section Residences
  `r paste(residences, collapse="\n")`
</div>
```

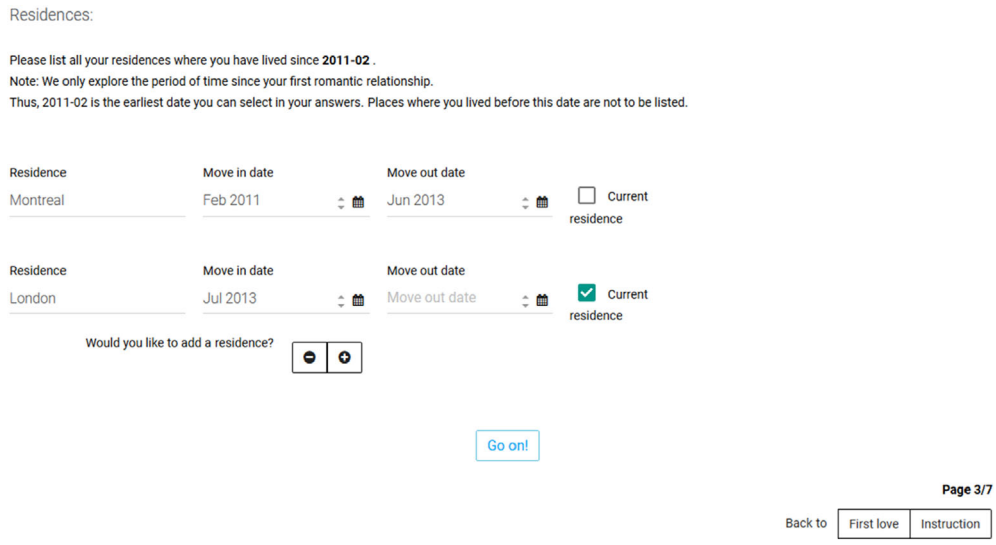

FIGURE 5 Example of the user interface in formr with items collecting the information required for the event history calendar. The example shows items collecting residences and their temporal categorization as defined in Figure 4. Events are collected starting from February 2011 because the fictive participant indicated this as the date of the first romantic relationship

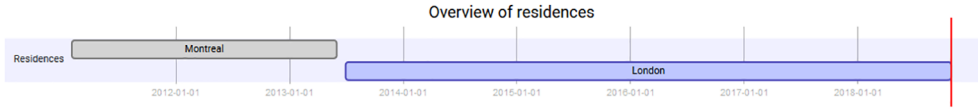

FIGURE 6 Output of an event history calendar with fictive data collected during the *Residences* domain, corresponding to the data input that is shown in Figure 5

Creating the occupations domain

For the next domain(s), the steps that were required for the *Residences* domain need to be repeated: Again, all items/events need to be predefined. Of course, all labels and variable names need to be adapted to the new domain. In our example, the second domain is the domain *Occupations*, and its items are defined in rows 54–96 of the spreadsheet. The four items required for one event will now be named differently. For example, we use the names "occupationtitle_O1," "datefrom_O1," "dateto_O1," and "current_occupation_O1" for the first occupation, followed by names with rising numbers for the items predefining the occupations from 1 to 10. After all items, including a further *counter* item for occupations in row 95, are defined in a manner that is parallel to the first domain, the rows defining a validation mechanism are inserted. Again, all variable names need to be changed in order to (a) differ from the *Residence* section and (b) specifically refer to the *Occupations* section. After defining the items necessary for the *Occupations* domain, we use R to aggregate the collected data and display an updated EHC showing information from the *Residences* and *Occupations* domains. Again, short explanations are provided within comments that are highlighted in black. R code that is new, compared to the R code printing only the first domain in a calendar, is written in bold font.

```
```{r}
gannt_event_row <- function (i, data, col_title, col_datefrom,
col_dateto, col_current) {
  empty <- ''

  if(!(paste0(col_title, i) %in% colnames(data))) {
    return (empty)
  }

  title <- last(data[paste0(col_title, i)])
  if (is.na(title) || title == '') {
    return (empty)
  }

  is_current <- last(data[paste0(col_current, i)]) == 1
  datefrom <- last(data[paste0(col_datefrom, i)])
  dateto <- ifelse(is_current, format(Sys.Date(), "%Y-%m-%d"),
last(data[paste0(col_dateto, i)]))
  tag <- ifelse(is_current, "active", "done");
label <- paste0("a", i)

  paste(title, ":", tag, ",", label, ",", datefrom, ",", dateto, " ",
sep = " ")
}

residences <- NULL;
for (x in 1:last(My_EHC_survey$count_SR)) {
  residences <- c(residences, gannt_event_row(x, My_EHC_survey,
'residence_M', 'datefrom_M', 'dateto_M', 'current_residence_M'))
}

# A second loop and vector are added to store the information of the
Occupations domain; again, the function specified above runs in a loop
for every entry of a participant
occupations <- NULL;
for (x in 1:last(My_EHC_survey$count_SO)) {
  occupations <- c(occupations, gannt_event_row(x, My_EHC_survey,
'occupationtitle_O', 'datefrom_O', 'dateto_O', 'current_occupation_O'))
}

# Below, a new text input for mermaid.js is given: The title is
extended and the vector with the information of the Occupations domain
is added
```
<div class="formr-mermaid">

Gantt
  title Overview of residences and occupations
  dateFormat YYYY-MM-DD
  section Residences
  `r paste(residences, collapse="\n")`
  section Occupations
  `r paste(occupations, collapse="\n")`
</div>
```

Special case: Creating the events domain

You might want to give participants the opportunity to name single-date events as opposed to extended episodes. These kinds of events only require the first two pieces of information, (1) an event title and (2) a date. Therefore, only two items, (1) an item of the type *text* and (2) an item of the type *month*, need to be defined in the spreadsheet, with names following the scheme "eventtitle_E1," "datefrom_E1," and so on (see rows 100–120 of the spreadsheet). The second *month* item (the end date) and the indicator of a current event, as they were used for extended events, can be left out. In addition, we decided to make the *Events* domain in our example study

optional—participants can list important life events but skip this section if nothing special happened in the relevant time interval. This is why all items of this domain, except the *counter* (which is renamed and redefined again in order to refer to the *Events* domain), are marked as optional with "*" within the *optional* column of the spreadsheet. Both special characteristics—single-date events and optional nature of the domain—require a new function and some adaptations of the R code printing the Gantt diagram. Again, short explanations are provided within comments that are highlighted in black, and the new code is presented in bold font. To save space, the R code collecting information from the first two domains (with extended events) is replaced by "....":

```
... [Function collecting information of the extended events]

# Additional function that collects an event's title and start date;
the collected information is pasted in the format that can be read by
mermaid.js, provided that any events were listed
gannt_events <- function (i, data, col_title, col_datefrom) {
  empty <- NULL

  if(!(paste0(col_title, i) %in% colnames(data)) ||
is.na(data[paste0(col_title, i)])) {
    return (empty)
  }

  title <- last(data[paste0(col_title, i)])
  if (is.na(title) || title == '') {
    return (empty)
  }

  datefrom <- last(data[paste0(col_datefrom, i)])
  tag <- "done"
  label <- paste0("a", i)

  paste(title, ":", tag, ",", label, ",", datefrom, ",", datefrom, " ",
sep = " ")
}


... [Loops and vectors storing information of the extended events]

# A third loop and vector are added to store information of the Events
domain; the new function specified above runs in a loop for every entry
of a participant
events <- NULL;
for (x in 1:last(My_EHC_survey$count_SE)) {
  events <- c(events, gannt_events(x, My_EHC_survey, "eventtitle_E",
"datefrom_E"))
}

# Below, a new text input for mermaid.js is given: The title is
extended and, provided that the participant added any events, the
vector with the information of the Events domain is added
```

Gantt
  title Overview of residences, occupations, and events
  dateFormat YYYY-MM-DD
  section Residences
  `r paste(residences, collapse="\n")
  section Occupations
  `r paste(occupations, collapse="\n")`
  `r ifelse(!is.null(events), paste(c("section Events", events),
collapse="\n"), "")`

```
```

A new function is used here to parse the data entered into the *Events* domain and generate the text structure required for the Gantt chart. Note here that the "start date" of the event is used as the end date to follow the pattern of the two previous domains. (Actually, mermaid.js is able to print Gantt graphs with a start date only, but it is necessary to insert an end date to keep the loop in the code above running.) Moreover, the function and the code of the *Events* section within the Gantt chart are specified in a way that the domain is only printed if the participant added any life events. Otherwise, this part is just skipped.

Creating the relationships domain

Until now, we have learned how to define the items and the Gantt charts of the three domains *Residences*, *Occupations*, and *Events*. In our study example, a fourth and final domain—the *Relationships* domain—is planned to collect information (i.e., names and relationship dates) about the participants' current and former romantic partners. In contrast to the previous domains, participants will have to name at least two events (i.e., partners). Figure 7 illustrates the item definitions of our last domain in rows 122–160 of the spreadsheet. As with the other events, each relationship is defined by four items with names following the scheme "partner_P1," "datefrom_P1," "dateto_P1," and "current_partner_P1." Please keep in mind that all partner names are stored in variables named "partner_P1," "partner_P2," "partner_P3," etc.—we will keep working with these later when we assess more information on those partners. In order to increase the number of required event entries from one to two, the four items of both the first and second relationship are not bound to a *showif* condition. Instead, the *showif* conditions are attached to the items of the relationships 3—10, starting with "count_SP > 2" in the *showif* column of the items of the third relationship. The default value of the counter variable is changed to "2."

As the extension of the R code for the Gantt chart is equivalent to the *Occupations* domain, it will not be further explained in this tutorial. Just remember that all variable names again need to be adapted in order to refer to the *Relationships* domain. At this point, our EHC is complete, and the Gantt chart displays all four domains. The survey ends with a final *submit* item.

Prospect: Adding further domains

The four domains of the example EHC are relatively generic and can probably be used for many studies dealing with relationship-related topics. Nonetheless, the content of the domains could be substituted, and researchers can readily add further domains in the same manner in which the four domains in the example study were created. For instance, a fifth or sixth calendar domain could be used to collect and visualize more specific relationship events and aspects. Naming just a few, one could assess health-related or psychosocial issues of the participant and her or his partner, sexual affairs, marriage, or the birth of children by adding corresponding domains to the calendar. As a next step, an additional survey could be created to gather more information about each partner and relationship the participants listed. The implementation of corresponding items will be explained further below. First, we will proceed with the implementation of the EHC itself.

class	type	name	showif	optional	label	value	choice1
clear right200 align_horizontally label_as_placeholder	note	instruct_relationship			####Relationships: For the final parts of the survey, we need you to name all partners …		
right80 align_horizontally label_as_placeholder	text 50	partner_P1			Name		
right80 align_horizontally label_as_placeholder	month ,now	datefrom_P1			Relationship start		
right80 align_horizontally label_as_placeholder	month ,now	dateto_P1			Relationship end		
right80 align_horizontally	mc_multiple	current_partner_P1		*			Current relationship
right80 align_horizontally label_as_placeholder	text 50	partner_P2			Name		
right80 align_horizontally label_as_placeholder	month ,now	datefrom_P2			Relation start		
right80 align_horizontally label_as_placeholder	month ,now	dateto_P2		*	Relation end		
right80 align_horizontally	mc_multiple	current_partner_P2		*			Current relationship
… repeat these four rows for the relationships 3-10. Remember to increase the number in the variable name and in the show-if condition by +1 for every new relationship.							
Counter	number 1,10	count_SP			Would you like to add a relationship?	2	
	submit	Submit06			Go on!		

1 **Variable stores partner name**

2 3 **Variables store relationship dates**

4 **Variable stores value „1“ if partner is current**

Participants have to name at least two partners: There is no showif condition for the second relationship and it starts with „count_SP > 2“ for the third relationship (not displayed) and counter default is „2“

FIGURE 7 Spreadsheet defining items for the domain *Relationships*. Participants will be required to enter at least 2 partners and a maximum of 10 partners

2.2.4 | *Step 4*: Survey upload and settings in formr

The spreadsheet created above now needs to be imported to formr. To do so, visit https://formr.org/login and enter your login credentials. Once successfully logged in, you will be

redirected to the admin section of formr. Go to *Surveys* → *Create new Survey* in the top menu. Depending on the format in which you saved the spreadsheet, you either need to upload it from your local computer or import a Google spreadsheet with a link to the online file. If you decide to import a Google spreadsheet, enter the survey name "My_EHC_survey." Please follow this naming convention as this is used to refer to variables of the spreadsheet's R code snippets and in the CSS and JavaScript code snippets, too. A change in this name would mean that you need to modify the code snippets accordingly. Moreover, please make sure that you have set appropriate permissions on your Google spreadsheet to be readable for everyone with the sheet's link. Click the *Import* button to complete the upload of the spreadsheet.

To permit users navigating back and forth within the EHC study and make changes to already given answers, we need to configure our created survey to allow *paging*. Go to *Surveys* → *My_EHC_survey*. This will lead you to the settings of that survey. Scroll to the bottom of the page, tick the *Enable Paging* checkbox, and save your settings. Please be aware of the fact that you cannot reverse this function once it is enabled. There are many survey types where participants should not be allowed to edit their answers. However, during an EHC study, this might be very useful because it is likely that participants remember events from an earlier domain when working on a later one or realize that they mixed up some dates when looking at the events' visualization.[9]

2.2.5 | *Step 5*: Setting up the EHC study: The formr run

From this point on, we will operate within the study control framework of formr. After creating and importing your questionnaire spreadsheet as a formr survey, the next step will be to set up the control structure of your study, in formr parlance a *run*. Go to *Runs* → *Create new Run*. Enter the name of the run, for example, "EHC-study." Please note that, within formr, each possible run name can only be used once—if a name is already taken, a warning message will be shown, and you need to choose another name. The above-mentioned name will be used in this tutorial in order to refer to the exemplary study. After successfully selecting a name, you will be redirected to the study control interface. Although a survey represents a part of the study, the run represents the whole study structure, which usually includes one or more surveys. It comprises subcomponents known as *run units*. Each unit carries out a specific task and can be added to the run with the click of the mouse on the appropriate run unit button located at the bottom of the study control interface. For example, an *Email* unit sends out emails, a *Survey* unit displays a questionnaire (survey), and so on. For more information on formr run units, please consult the formr documentation.[10] As a minimal setup, we will need a *Survey* unit to display our questionnaire and a *Stop Point* unit to end the study and display a message to the participant when the study is over. If you are not already there, go to *Runs* → *EHC-study*. At the bottom of the page, you should see the panel depicted in Figure 8.

Adding the run units

Click on the *Add survey* button and use the dropdown to select the name of the survey you created in the previous section. In our case, that will be "My_EHC_survey." Next, click on the

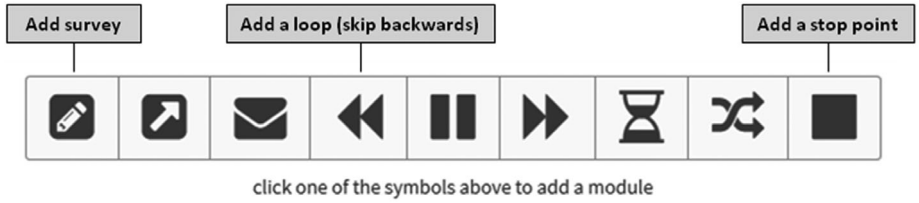

FIGURE 8 Panel with buttons that can be clicked to add units to the formr run

Add a stop point button and enter some text you would like displayed to the participant at the end of the study. It could be something as simple as "Thank you for participating!" or some complex feedback with interactive charts (written in R). It is up to the researcher what goes in there.

With the above setup of our study, we will now add some settings to the formr run. As mentioned earlier, formr gives you the possibility of adding custom CSS and JavaScript to your study. The CSS setting will enhance the look and feel of the items displayed to the user, whereas the JavaScript setting will contain code responsible for performing some client side validation and also loading the mermaid.js library responsible for displaying the EHC.

CSS setting

Still in the run control interface of your run, click on the *Settings* button located at the left menu. Go to the *CSS* tab and copy and paste the code snippet located at https://osf.io/5nzgm/. Save the CSS settings by clicking on the *Save* button in the upper right corner. This CSS code brings some of the items into position and defines the colors used in the EHC Gantt chart.

JavaScript setting

Now go to the *JS* tab of the settings page and paste the code snippet located at https://osf.io/3vzwh/. Save the JavaScript settings by clicking on the *Save* button in the upper right corner. This JavaScript code is essential to make the study work. Therefore, we will take a closer look at it. Please note that we structured the provided code snipped in a way that first defines a number of functions and then calls all of these functions in the lines following the comment "Run relevant JavaScript functions when the web page is ready." In order to recognize what lines of code refer to the same function, however, definition and call of a function will be presented and explained directly one after the other.

We are using mermaid.js to display the EHC on the browser. This JavaScript library accepts a Gantt chart definition in text form—as we produced it with the R code in the survey spreadsheet—and outputs the required graphics. The following code provides the minimal JavaScript setup, which is required to run our EHC study as it connects formr with mermaid.js:

```
**
 * Load required mermaid.js library
 * Initialize the mermaid.js library on all ".formr-mermaid" elements
 *
 * @param {String} url URL to JS library
 */
function loadMermaidJs(url) {
    // Create a <script> tag that will be used to include the mermaid
JS library in the web page
    var script = document.createElement('script');
    script.src = url;

    // Define a callback function to be called once the script is
loaded
    // Our callback function will search for all web page elements (=
interfaces between JavaScript and HTML) with class .formr-mermaid
    // and use its text content to initialize the mermaid.js library
    script.onload = function () {
        mermaid.initialize({startOnLoad:false});
        $(document).ready(function() {
            $(".item-note .formr-mermaid").each(function() {
                var $this = $(this);
                $this.text($this.text());
                $this.show();
            });
            mermaid.init({noteMargin: 10}, ".item-note .formr-
mermaid");

        });
    };

    // append the created <script> tag to the <head> element of the web
page
    document.head.appendChild(script);
}

...
// Load the mermaid JavaScript library
loadMermaidJs("https://exp.psych.bio.uni-
goettingen.de/mermaid/mermaid.min.js");
```

The code above searches for an *HTML* element in the page with the class name "formr-mermaid." This element has been generated with R code in the spreadsheet (see code defined during the spreadsheet creation) and contains the required Gantt chart definition in text form. The mermaid.js library is applied to this element, converting its text content into a graphic displaying the EHC.

Furthermore, you can edit the labels of the buttons to skip forward and backward within the EHC survey with help of the following code in the JavaScript settings of the run:

```
/**
 * Change the texts on the paginations buttons
 * Instead of the default numbering, use 'texts' representing the various
'categories of interest'
 */
function changePaginationButtonsText() {
    // Label of the text element indicating the current page/position
within the EHC
    $('.page-text').text('Page ');
    // Label of the text element next to the page labels
    $('.back-text').text('Back to ');
    // Label for each of the pages
    var pages = ['First Love', 'Instruction', 'Residences',
'Occupations', 'Life Events', 'Relationships', 'Complete Calendar'];
    var $study = $('.study-name-My_EHC_survey');
    // assignment of page labels to their corresponding page numbers
    for (var i in pages) {
        var pageNo = i*1 + 1;
        var pageLabel = ' ' + pages[i] + ' ';
        $study.find('.page-buttons .btn-page-' + pageNo).text(pageLabel);
    }
}

...

$(document).ready(function() {
    // Change the texts on the paginating buttons
    changePaginationButtonsText();
```

Every page is given a meaningful label to assist the participant navigate back and forth in the study. Moreover, you can adapt the labels "Page" and "Back to" via the first two lines of code, for example, if you would like to run your survey in a language that is not English.

Finally, we added some JavaScript code that supports the user in entering valid data. First, an event's end date is set to the event's start date as a default. This increases usability by leading the participant to the most likely dates within the date selection menu instead of having her or him clicking back from the current date over and over. Second, the code disables the selection of an end date that is earlier than the start date of the same event and thereby reduces the probability of invalid data. Third and finally, the JavaScript code performs a validation of the participant's date input, which goes beyond the validation mechanism that was defined within the spreadsheet. Roughly summarized, the validation code causes error messages to pop up if the participant forgets to either enter an end date or indicate an event as current. Moreover, users receive a note if they enter the same date for the start and end of an event and are asked whether they are sure that these dates are correct. Apart from changing the appearance of survey pages and implementing mechanisms that facilitate the users' data input, JavaScript generally gives you the flexibility to add more, sometimes fancy, details to your formr study.

2.2.6 | *Step 6*: Testing the study

Now that the run is created and all its settings are configured—here, this should be the case right after uploading the spreadsheet to formr, adding it as a survey to the run and configuring the CSS and JavaScript settings—the next step is to test the study. In the control framework of formr, you

can determine the publicness of your study. The buttons for the publicness settings, as they are depicted in the formr control framework, are shown in Figure 9. By default, publicness is set to be accessed for no one but you. With this setting, you can test and change your study without being afraid that users will see it. When testing the study, it often makes sense to pick the second option of the publicness settings, access for you and everyone with an access code that you can distribute. Again, you can edit your study without real users noticing. However, you have the possibility of sending access codes to people you know to let them run the study on trial. To test the study on your own, click on the *Test Run* button on the left side of the study control panel. This will enable you to click through the study and see how it appears and if everything is working. Deviating from real users, you can make use of the *monkey mode* and skip run sections. The monkey mode is a function in formr that automatically fills out the survey pages—this can save you a lot of time when testing large questionnaires again and again! Furthermore, the test run might display warning messages to you, which will not be apparent for real users.

Every time you start a test run, a new user/test session is created. You can view all existing test sessions by clicking on the *Old Guinea Pigs* button right under the *Test Run* button. Session codes of test users contain animal names and the letters "XXX." This will enable you to differentiate between test and real users later. Within the user overview, you can delete old test sessions, send users to certain positions in the run, and test the study again with their session code. By right clicking on the symbol in the *Action* column and choosing the option *Copy link address*, you can retrieve the access code to this test and send it to someone.

2.2.7 | *Step 7*: Running the study for real

After testing the run and finding that everything works correctly, your study is ready to go public. Depending on who you want to reach, you can either choose the inner or outer left of the two left buttons of the publicness settings (see Figure 9). In the first case, everyone with the official study link—which is found right on top of the study control panel, next to the run name—can enroll in the study. When choosing this option, you will need to distribute the link via e-mail or online platforms. Most researchers will probably prefer this publicness setting because it provides them with control over the study access. In the second case, the study link will be publicly posted to the study page in formr,[11] and everyone visiting the page can enroll. Of course, the study link can be distributed via additional ways in this case, too.

Managing users

To be directed to the user overview, click on the *Users* button on the left side of the control panel. Actually, this leads you to the same page as the *Old Guinea Pigs* button, except that the search function does not filter for the "XXX," which specifies test sessions as opposed to real

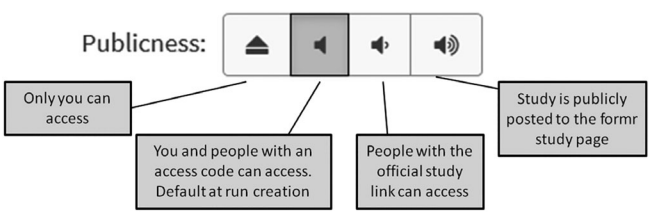

FIGURE 9 Publicness settings in formr. By default, the study can be accessed by the study administrator and people with an access code

sessions. Again, you can track the users' progress on the basis of their run position, send them to another run position, or delete their session.

Viewing and exporting results

Formr saves the user data in a result chart, with one row for every participant or observation and one column for each of the study variables from the survey spreadsheet plus information about the processing times. Results are stored separately per survey. This is especially helpful when many variables are assessed, and result tables would become very large otherwise. To access the results of one formr survey, go to the corresponding survey page. In our case, this means going to *Surveys* → *My_EHC_survey*. To view the results online, click the *Show Results* button on the left side of the page. To export the results, click on the *Export Results* button underneath and choose one of the file formats depending on your preferences or the preferences of your statistics program.

3 | BEYOND THE CALENDAR: EMBEDDING THE TOOL INTO A LARGER STUDY

3.1 | General options to use and extend the EHC

With the steps instructed so far, you know everything necessary to create a minimal setup for an EHC study. So far, the information that can be collected with this EHC would enable researchers to count the number of long-term partners and calculate the corresponding relationship durations. It would also be possible to look at the time spans of residences or occupations and check whether moves temporally coincide with breakups. As noted before, further domains could be added to collect further and more specific information.

However, you might want to embed the calendar into a larger study that includes further questionnaires. For instance, you will most likely be interested in gathering some demographic data of the participants. In addition, there a several research questions going beyond the EHC domains we have generated in our example study, thus going beyond the examples outlined above. Imagine that your goal was to assess further attributes of the partners and relationships listed in the EHC. These attributes could be compared across the relationships with respect to similarity or variability: Do the different former and current partners of a participant have certain traits in common? How (dis-)similar are the different relationships, for example, with regard to their quality or the power balance, as perceived by the participant? In general, embedding the EHC in a larger study becomes feasible by simply uploading and adding further surveys to your formr run. This can either happen in the form of additional *Survey* units or alternatively with *External link* units directing to surveys that are implemented with other survey software. In the next section, we will learn how to create a second formr survey that gathers further details about all partners and relationships a participant listed during the EHC survey.

3.2 | Gather more information about relationships listed in the calendar

Let us consider the exemplary and extended research goal from above in more detail: To find out if all current as well as former partners and relationships of one person have certain

attributes in common, information about every partner and relationship a participant listed needs to be obtained. For example, one could examine specific personality characteristics of the partners and ex-partners, ask if any children were or are planned in the relationship, or who initiated the breakup in the case of ex-relationships. In this way, the EHC would be the first step in a study by collecting the names and dates of all partners and ex-partners, followed by the assessment of more details on the corresponding relationships. The steps that are required to create a second survey that gathers more information about the relationships will be explained next. In order to distinguish these steps from the steps of implementing the EHC itself, they will be ordered by letters instead of numbers.

3.2.1 | *Step A*: Creating a new spreadsheet

In the beginning, we need to create a new spreadsheet. Please download the spreadsheet named "Relationship_details" located at https://osf.io/txpd9/ and save it locally or add it to Google Spreadsheets. Basically, this spreadsheet is just another item table (formr survey definition). After completing all the steps explained in this section, all items in the "Relationship_details" spreadsheet should be repeated for each partner a participant listed during the EHC survey. Therefore, the new survey needs to refer to our first survey called "My_EHC_survey" and retrieve some of the variables stored in the corresponding result file. When opening the "Relationship_details" spreadsheet, you will find *calculate* items in the first five rows. These items' definitions are illustrated in Figure 10 and add some useful adaptations to the survey via their functions defined in the *value* column, as will be explained below.

Defining calculate items

First, the *calculate* items enable us to insert the correct partner name for each partner about whom the questions are posed. Remember that, in our first survey, partner names were stored in variables named "partner_P1," "partner_P2," etc. The first *calculate* item in row 2 recognizes how often the survey has been processed and thus stores the value "1" for the first partner, "2" for the second partner, etc. In row 3, the second *calculate* item refers to the "My_EHC_survey"

type	name	showif	optional	label	columns choice1-4	value
calculate	person_nr					nrow(Relationship_details)
calculate	name					toString(last(My_EHC_survey[paste0("partner_P", person_nr)]))
calculate	current					ifelse(toString(last(My_EHC_survey[paste0("current_partner_P", person_nr)])) == 1, 1, 0)
calculate	had					ifelse(current == 1,"has","had")
calculate	were					ifelse(current == 1,"are","were")

Variable stores for how many partners the current survey has been processed

Variable retrieves partner names from the EHC survey, starting with name stored in the variable „partner_P1"

Variable stores value „1" if partner was marked as current in the EHC survey

Variables calculate words in present tense for current partners and words in past tense for ex-partners

FIGURE 10 Spreadsheet defining *calculate* items that are used to tailor the survey's appearance based on the partner's name and relationship status (i.e., being a current partner or ex-partner)

survey and merges the string "partner_P" with these numbers, resulting in the names of the variables that store the partner names. Together, the two *calculate* items can retrieve the correct name of each listed partner and ex-partner. Thus, when inserting the variable "name" into the spreadsheet's *label* column of an item via R code, the correct partner name will be displayed in its place when a participant processes the survey.

Second, the *calculate* items enable us to switch between present and past tense, depending on whether questions refer to a current partner or to an ex-partner. Going to row 4, the *calculate* item retrieves and stores the information regarding if a partner is a current partner or an ex-partner from the EHC survey. This information can be used by the last two *calculate* items: If partners are current, the *ifelse* conditions in the *value* column produce the words written in present tense. Else, the words in past tense are produced. Thus, when inserting the variables "had" or "were" into the spreadsheet's *label* column of an item via R code, the correct tense of the words will be displayed in its place when a participant processes the survey.

Define relationship-specific questions

After all required *calculate* items are defined, we can move on to the actual questions about the partners and relationships. To provide a temporal orientation and facilitate the recall of this information, we can reprint the complete EHC diagram at the top of the survey's page by pasting the R code into the *label* column of a *note* item, which is the first item of the survey page: This R code resembles the code that was used to print the complete calendar at the end of the EHC survey (thus visualizing the four domains *Residences*, *Occupations*, *Events*, and *Relationships*), except that it specifies the file "My_EHC_survey" as the origin of all variables. When not specified, formr would search for the variables in the current survey sheet—which is "Relationship_details"—without finding them.

Below the EHC graphic, questions are posed that will be repeated for every listed relationship. In particular, our example spreadsheet defines one set of items concerning the attributes of each partner or ex-partner in rows 8–12. For example, one item asks how trustworthy a partner is. Within the item definitions, R code refers to the *calculate* items we created before. Whenever a participant opens the survey page, the values stored in these *calculate* variables are shown in place of the R code. Thus, if a participant listed a partner named Peter, the mentioned item will read "What do you think, how trustworthy is Peter?" A second survey page in our example study starts again with the EHC diagram on top and includes a set of items concerning the relationship or ex-relationship, defined in rows 15–23 of the spreadsheet. For instance, one item asks about the power distribution within current and past relationships. With help of the *calculate* items and R code, the corresponding variable will show the verb form of "have" depending on the relationship status of the partner or ex-partner in question. If the question concerns a current partner, the item will read "In your relationship, who has more power?" Otherwise, the item will read "In your relationship, who had more power?"

Please note that the items given in the provided spreadsheet only represent examples. You can use the spreadsheet as a template for modification and replace or add further questions that fit your specific research goal.

3.2.2 | *Step B*: Survey upload

When the second spreadsheet is defined, it needs to be uploaded to formr again. Please proceed as you did with the first survey. Once again, make sure that you retain the survey name

"Relationship_details" to keep all references to the survey working. After the upload, you can decide if custom paging should be enabled in the survey about the relationships too and, correspondingly, put the tick in the survey settings or not. Within the tutorial, we go without paging in this survey as we are interested in ratings that are given spontaneously and should not be altered later on.

3.2.3 | *Step C*: Extending the formr run

In order to get the survey running, we need to add it to our already existing run. Please add a second *Survey* unit and import the "Relationship_details" survey via the dropdown of the unit. In order to let the survey run as a loop, which is repeated for each partner a participant listed in the EHC survey, we need to add a *Skip Backward* unit (see Figure 8). The following R code is stored in a file at https://osf.io/q4f9d/—please download the file and paste the code into the code field of the unit. It defines the condition on how and when another run unit is repeated; comments highlighted in black are added for explanation:

```
# check if the number of processed partners is smaller than the
number of listed partners as saved in the counter variable
nrow(Relationship_details) < My_EHC_survey$count_SP
```

After pasting this, set the position of the control panel saying *...skip backward to* to "10" and save the changes. The code checks (a) how often the "Relationship_details" survey has run and (b) if there is a further partner that has not been rated yet. When all listed partners have been rated, the loop ends, and participants are directed to the next unit, which should be the *Stop Point* unit and the study's end in our case.

Please make sure that the run units have the correct run position (otherwise, the study will not work). Run positions can be edited by increasing or decreasing the large numbers at the bottom of a run unit: The run should start with the *Survey* unit with the survey "My_EHC_survey" in run position "0," followed by the *Survey* unit with the "Relationship_details" survey in run position "10" and the *Skip Backward* unit in run position "20." Finally, the *Stop Point* unit is placed in run position "30." After editing the run positions, click on the *Reorder* button at the top of the run control panel. At this point, your extended EHC study that can gather further relationship details should be ready for testing. Please follow the steps for testing the run that were previously explained in the section referring to the EHC implementation.

In formr, there is a possibility to export the run structure defined above to a JSON file. This file can then be shared with other researchers who may be interested in collaboration or replication of the study. Such a JSON export containing a run definition can be imported to formr, and you will not need to manually perform the setup for a run as explained above. The link https://osf.io/kxdru/ will direct you to the JSON export of our example study. To test this JSON export, download the file from the above URL and save it on your local computer, create a new formr run, click on the *Import* button at the top of the run control panel, and select the downloaded JSON file. After clicking on the green *Import* button, the whole run will be built up based on the structure stored in the JSON file, and you can see what the online EHC and the survey gathering relationship details should look like.

4 | DISCUSSION

In this tutorial, we provided detailed instructions explaining how to create an online EHC using formr.org (Arslan & Tata, 2017) and mermaid.js (Sveidqvist, 2018). In our eyes, online EHCs hold the potential to make a great contribution to relationship science by combining the methodological benefits of calendar-based questionnaires and online data collection: Online EHCs enable researchers to improve the quality of retrospective data collection with the help of autobiographical memory cues (Belli, 1998; Freedman et al., 1988; Martyn & Belli, 2002; Morselli et al., 2016). At the same time, online implementation saves resources and allows data to be collected from a large sample easily (Riva et al., 2003; Selm & Jankowski, 2006; Wright, 2005). A particular benefit of the EHC online implementation introduced in this tutorial is the use of open-source software (see Kraker et al., 2011). The flexible framework of formr.org (Arslan & Tata, 2017) can be accessed without charge and supports collaborative development, as well as open methodology. Furthermore, surveys created in other survey frameworks can be integrated seamlessly into the formr study control framework.

Despite its advantages, it is important to consider some limitations of online EHCs. First, disadvantages of EHCs in general affect online EHCs as well. Probably the most important, EHCs cannot replace longitudinal studies: The use of an EHC helps to improve recall, but neither guarantees completeness of the recollections nor fully compensates for retrospective biases (Schwarz, 2007). At least in this tutorial, the data collected with the EHC are self-reports. Please keep in mind that this aspect can lead to biased ratings, too (e.g., by idealizing current partners and devaluating ex-partners; Busby, Holman, & Niehuis, 2009; Geher et al., 2005). As a solution to this problem, it is possible to add reports of informants or the partners themselves. However, the explanation of the required steps would go beyond the scope of this tutorial.

In addition, the extensive autobiographical reconstructions within an EHC can become quite time consuming for the participants. This is especially true when EHCs are combined with further questionnaires. Second, the many advantages of online studies are accompanied by disadvantages, too (Kraut et al., 2004; Selm & Jankowski, 2006; Wright, 2005). For instance, older people may not know how to operate computers and Internet (however, this population, excluded from online studies, is decreasing; Mossberger, Tolbert, & McNeal, 2007). More specific to EHCs, lack of an interviewer leaves the user of the online survey all alone with the instructions of the EHC. During an EHC that is performed as a face-to-face interview, the interviewer has the chance to double check answers or to ask for more details. When using an EHC online, it is crucial to equip the survey pages with very clear instructions that are as straightforward as possible and effectively guide participants regarding what to do. This view is supported by Morselli, Goff, and Gauthier (2018), who found that relatively good-quality data could be collected in the absence of an interviewer if instructions were clear enough. Even though this study compared self-administered paper-and-pencil EHCs completed with and without the help of an interviewer, the findings might still apply to online versions of EHCs. In addition, the conversational style of the interview in face-to-face EHCs holds the potential to improve the respondents' recall (Belli et al., 2001; Brüderl et al., 2017). In an online EHC, one can mimic the conversational style of face-to-face interviews to a certain extent, for example, by using direct questions that prompt participants to reconsider their answers. Finally, it should be noted that the current EHC online instrument has not been formally validated yet. However, user comments from the first two studies using the online EHC introduced in this tutorial support the memory structuring function of the dynamically created Gantt chart and positive participant

experience when reconstructing their relationship histories (Driebe, Penke, & Gerlach, 2019; Wieczorek, 2018).

To conclude, researchers should carefully consider the advantages and disadvantages of online studies, as well as the tradeoff between benefits and effort for participants and researchers resulting from the use of an EHC. If data quality can be improved with the help of a calendar instrument, and online collection of data is deemed appropriate, the online EHC presented in this tutorial may be a promising and flexible tool that may crucially aid in answering cutting-edge questions. This does not only apply to relationship science but also to other fields such as clinical research, criminology, or organizational psychology. Given the open-source nature of the formr-based online EHC, researchers are invited to adapt it for their own research goals and to participate in its further development and optimization.

ACKNOWLEDGMENTS

We thank Julie C. Driebe and Ruben C. Arslan for their support when setting up the first study using an online event history calendar in formr. We thank Salome Johannigmann, Kim Gloystein, and Isa Garbisch for testing the tutorial and providing feedback for its improvement.

AUTHOR CONTRIBUTIONS

Larissa L. Wieczorek and Cyril S. Tata wrote the first draft of the manuscript and created the online material for the tutorial. Tanja M. Gerlach and Lars Penke came up with the idea for the tutorial, gave conceptual input and advice regarding its conceptualization, and provided critical revisions of the manuscript.

ENDNOTES

[1] Belli et al. (2007) programmed an EHC in house, with copyright 2000 to the Regents of the University of Michigan. Brüderl et al. (2017) integrated the EHC as a Java application to their interview within the Panel Analysis of Intimate Relationships and Family Dynamics (pairfam) study. Morselli et al. (2016) implemented their online EHC with Java, making use of numerous software architectures (Spring webmvc, Dojo, Hibernate).

[2] https://mermaidjs.github.io/

[3] You can follow up this tutorial using the instance of formr (https://formr.org) hosted by the University of Goettingen. However, if you plan to conduct your own online studies on a more regular basis, we recommend setting up your own instance of formr (see the documentation at https://github.com/rubenarslan/formr.org/blob/master/INSTALLATION.md).

[4] In addition, when actually setting up your study, you might like to join the formr mailing list https://groups.google.com/forum/#!forum/formr to ask and answer questions.

[5] https://spreadsheets.google.com

[6] http://github.com/rubenarslan/formr

[7] Of course this includes the possibility of integrating software that is designed to run experimental studies online, such as the JavaScript library jsPsych (https://www.jspsych.org/) for behavioral online experiments.

[8] In our example, the number of events will be limited to 10 per domain. However, this is only a suggestion for the purpose of this tutorial, and researchers are free to set the lower and upper limits of the number of events individually.

[9] Indeed, behavioral analyses by Morselli et al. (2016) demonstrated that participants used the visualization provided by the EHC to correct previously given answers.

[10] www.formr.org/documentation

[11] https://formr.org/studies

REFERENCES

Arslan, R. C. (2018). *Formr R package (Version R Package Version 0.7.4)*. Retrieved from https://zenodo.org/record/1326523

Arslan, R. C., & Tata, C. S. (2017). *formr.org survey software (Version v0.17.14)*. https://doi.org/10.5281/zenodo.1228910

Arslan, R. C., Walther, M. P., & Tata, C. S. (2019). formr: A study framework allowing for automated feedback generation and complex longitudinal experience-sampling studies using R. *Behavior Research Methods.*, *52*, 376–387. https://doi.org/10.3758/s13428-019-01236-y

Asendorpf, J. B., Penke, L., & Back, M. D. (2011). From dating to mating and relating: Predictors of initial and long-term outcomes of speed-dating in a community sample. *European Journal of Personality*, *25*, 16–30. https://doi.org/10.1002/per.768

Axinn, W. G., Pearce, L. D., & Ghimire, D. (1999). Innovations in life history calendar applications. *Social Science Research*, *28*(3), 243–264. https://doi.org/10.1006/ssre.1998.0641

Belli, R. F. (1998). The structure of autobiographical memory and the event history calendar: Potential improvements in the quality of retrospective reports in surveys. *Memory*, *6*(4), 383–406. https://doi.org/10.1080/741942610

Belli, R. F., Shay, W. L., & Stafford, F. P. (2001). Event history calendars and question list surveys: A direct comparison of interviewing methods. *Public Opinion Quarterly*, *65*(1), 45–74. https://doi.org/10.1086/320037

Belli, R. F., Smith, L. M., Andreski, P. M., & Agrawal, S. (2007). Methodological comparisons between CATI event history calendar and standardized conventional questionnaire instruments. *Public Opinion Quarterly*, *71*(4), 603–622. https://doi.org/10.1093/poq/nfm045

Berntsen, D., & Rubin, D. C. (2004). Cultural life scripts structure recall from autobiographical memory. *Memory & Cognition*, *32*(3), 427–442. https://doi.org/10.3758/BF03195836

Bohannon, J. (2016). Psychologists grow increasingly dependent on online research subjects. *Science Magazine*, *7*. https://doi.org/10.1126/science.aag0592

Bordens, K. S., & Abbott, B. B. (2010). Using speacialized research designs. In *Research design and methods: A process approach* (8th ed., pp. 330–356). New York, NY: McGraw-Hill Humanities/Social Sciences/Languages.

Brown, N. R., & Sinclair, R. C. (1999). Estimating number of lifetime sexual partners: Men and women do it differently. *The Journal of Sex Research*, *36*(3), 292–297. https://doi.org/10.1080/00224499909551999

Brüderl, J., Castiglioni, L., Ludwig, V., Pforr, K., & Schmiedeberg, C. (2017). Collecting event history data with a panel survey: Combining an electronic event history calendar and dependent interviewing. *Methods, Data, Analyses*, *11*(1), 22. https://doi.org/10.12758/mda.2016.013

Busby, D. M., Holman, T. B., & Niehuis, S. (2009). The association between partner enhancement and self-enhancement and relationship quality outcomes. *Journal of Marriage and Family*, *71*(3), 449–464. https://doi.org/10.1111/j.1741-3737.2009.00612.x

Campbell, L., Chin, K., & Stanton, S. C. E. (2016). Initial evidence that individuals form new relationships with partners that more closely match their ideal preferences. *Collabra: Psychology*, *2*(1), 1–7. https://doi.org/10.1525/collabra.24

Driebe, J. C., Penke, L., & Gerlach, T. M. (2019, July 26). *Partner preferences—A longitudinal investigation*. Retrieved from osf.io/2ezfk

Eastwick, P. W., Harden, K. P., Shukusky, J. A., Morgan, T. A., & Joel, S. (2017). Consistency and inconsistency among romantic partners over time. *Journal of Personality and Social Psychology*, *112*(6), 838–859. https://doi.org/10.1037/pspi0000087

Ebbinghaus, H. (2013). Memory: A contribution to experimental psychology (1885). *Annals of Neurosciences*, *20*(4), 155–156. https://doi.org/10.5214/ans.0972.7531.200408

Fikowski, J., Marchand, K., Palis, H., & Oviedo-Joekes, E. (2014). Feasibility of applying the life history calendar in a population of chronic opioid users to identify patterns of drug use and addiction treatment. *Substance Abuse: Research and Treatment*, *8*, SART.S19419. https://doi.org/10.4137/SART.S19419

Freedman, D., Thornton, A., Camburn, D., Alwin, D., & Young-DeMarco, L. (1988). The life history calendar: A technique for collecting retrospective data. *Sociological Methodology*, *18*, 37–68. https://doi.org/10.2307/271044

Geher, G., Bloodworth, R., Mason, J., Stoaks, C., Downey, H. J., Renstrom, K. L., & Romero, J. F. (2005). Motivational underpinnings of romantic partner perceptions: Psychological and physiological evidence. *Journal of Social and Personal Relationships*, *22*(2), 255–281. https://doi.org/10.1177/0265407505050953

Gerlach, T. M., Arslan, R. C., Schultze, T., Reinhard, S. K., & Penke, L. (2019). Predictive validity and adjustment of ideal partner preferences across the transition into romantic relationships. *Journal of Personality and Social Psychology*, *116*(2), 313–330. https://doi.org/10.1037/pspp0000170

Glasner, T., & van der Vaart, W. (2009). Applications of calendar instruments in social surveys: A review. *Quality and Quantity*, *43*(3), 333–349. https://doi.org/10.1007/s11135-007-9129-8

Hardt, J., & Rutter, M. (2004). Validity of adult retrospective reports of adverse childhood experiences: Review of the evidence. *Journal of Child Psychology and Psychiatry*, *45*(2), 260–273. https://doi.org/10.1111/j.1469-7610.2004.00218.x

Hyman, I. E., Jr., & Pentland, J. (1996). The role of mental imagery in the creation of false childhood memories. *Journal of Memory and Language*, *35*(2), 101–117. https://doi.org/10.1006/jmla.1996.0006

Kinsey, A. C., Pomeroy, W. R., & Martin, C. E. (2003). Sexual behavior in the human male. *American Journal of Public Health*, *93*(6), 894–898. https://doi.org/10.2105/AJPH.93.6.894

Kraker, P., Leony, D., Reinhardt, W., & Beham, G. (2011). The case for an open science in technology enhanced learning. *International Journal of Technology Enhanced Learning*, *3*(6), 643–654. https://doi.org/10.1504/IJTEL.2011.045454

Kraut, R., Olson, J., Banaji, M., Bruckman, A., Cohen, J., & Couper, M. (2004). Psychological research online: Report of Board of Scientific Affairs' Advisory Group on the conduct of research on the internet. *American Psychologist*, *59*(2), 105–117. https://doi.org/10.1037/0003-066X.59.2.105

Loftus, E. F. (1979). The malleability of human memory: Information introduced after we view an incident can transform memory. *American Scientist*, *67*(3), 312–320.

Luke, N., Clark, S., & Zulu, E. M. (2011). The relationship history calendar: Improving the scope and quality of data on youth sexual behavior. *Demography*, *48*(3), 1151–1176. https://doi.org/10.1007/s13524-011-0051-2

Martyn, K. K., & Belli, R. F. (2002). Retrospective data collection using event history calendars. *Nursing Research*, *51*(4), 270–274.

Martyn, K. K., Reifsnider, E., & Murray, A. (2006). Improving adolescent sexual risk assessment with event history calendars: A feasibility study. *Journal of Pediatric Health Care*, *20*(1), 19–26. https://doi.org/10.1016/j.pedhc.2005.07.013

Mathur, A., & Evans, J. R. (2005). The value of online surveys. *Internet Research*, *15*(2), 195–219. https://doi.org/10.1108/10662240510590360

Mitchell, K. R., Mercer, C. H., Prah, P., Clifton, S., Tanton, C., Wellings, K., & Copas, A. (2018). Why do men report more opposite-sex sexual partners than women? Analysis of the gender discrepancy in a British national probability survey. *The Journal of Sex Research*, *56*, 1–8. https://doi.org/10.1080/00224499.2018.1481193

Morris, N. A., & Slocum, L. A. (2010). The validity of self-reported prevalence, frequency, and timing of arrest: An evaluation of data collected using a life event calendar. *Journal of Research in Crime and Delinquency*, *47*(2), 210–240. https://doi.org/10.1177/0022427809357719

Morselli, D., Berchtold, A., Suris Granell, J.-C., & Berchtold, A. (2016). On-line life history calendar and sensitive topics: A pilot study. *Computers in Human Behavior*, *58*, 141–149. https://doi.org/10.1016/j.chb.2015.12.068

Morselli, D., Goff, J.-M. L., & Gauthier, J.-A. (2018). Self-administered event history calendars: A possibility for surveys? *Contemporary Social Science*, *14*, 1–24. https://doi.org/10.1080/21582041.2017.1418528

Mossberger, K., Tolbert, C. J., & McNeal, R. S. (2007). Defining digital citizenship. In *Digital citizenship: The internet, society, and participation* (pp. 1–20). Cambridge, MA: The MIT Press.

Neyer, F. J., & Asendorpf, J. B. (2001). Personality-relationship transaction in young adulthood. *Journal of Personality and Social Psychology*, *81*(6), 1190–1204. https://doi.org/10.1037/0022-3514.81.6.1190

Ooms, J. (2014). The OpenCPU system: Towards a universal interface for scientific computing through separation of concerns. *ArXiv [Stat.CO]*. Retrieved from http://arxiv.org/abs/1406.4806

Open Science Collaboration. (2015). Estimating the reproducibility of psychological science. *Science*, *349*, aac4716(6251). https://doi.org/10.1126/science.aac4716

Park, Y., & MacDonald, G. (2019). Consistency between individuals' past and current romantic partners' own reports of their personalities. *Proceedings of the National Academy of Sciences*, *116*(26), 12793–12797. https://doi.org/10.1073/pnas.1902937116

Patten, S. B. (2009). Accumulation of major depressive episodes over time in a prospective study indicates that retrospectively assessed lifetime prevalence estimates are too low. *BMC Psychiatry*, *9*(1), 19. https://doi.org/10.1186/1471-244X-9-19

Penke, L., & Asendorpf, J. B. (2008). Beyond global sociosexual orientations: A more differentiated look at sociosexuality and its effects on courtship and romantic relationships. *Journal of Personality and Social Psychology*, *95*(5), 1113–1135. https://doi.org/10.1037/0022-3514.95.5.1113

Riva, G., Teruzzi, T., & Anolli, L. (2003). The use of the internet in psychological research: Comparison of online and offline questionnaires. *Cyberpsychology & Behavior*, *6*(1), 73–80. https://doi.org/10.1089/109493103321167983

Robins, R. W., Caspi, A., & Moffitt, T. E. (2002). It's not just who you're with, it's who you are: Personality and relationship experiences across multiple relationships. *Journal of Personality*, *70*(6), 925–964. https://doi.org/10.1111/1467-6494.05028

Roisman, G. I., Madsen, S. D., Hennighausen, K. H., Sroufe, L. A., & Collins, W. A. (2001). The coherence of dyadic behavior across parent-child and romantic relationships as mediated by the internalized representation of experience. *Attachment & Human Development*, *3*(2), 156–172. https://doi.org/10.1080/14616730126483

Ross, M. (1989). Relation of implicit theories to the construction of personal histories. *Psychological Review*, *96*(2), 341–357. https://doi.org/10.1037/0033-295X.96.2.341

Rubin, D. C., & Wenzel, A. E. (1996). One hundred years of forgetting: A quantitative description of retention. *Psychological Review*, *103*(4), 734–760. https://doi.org/10.1037/0033-295X.103.4.734

Schaie, K. W. (1983). What can we learn from the longitudinal study of adult psychological development? In K. W. Schaie (Ed.), *Longitudinal studies of adult psychological development* (pp. 1–19). New York, NY: Guilford Press.

Schaie, K. W. (2000). The impact of longitudinal studies on understanding development from young adulthood to old age. *International Journal of Behavioral Development*, *24*(3), 257–266. https://doi.org/10.1080/01650250050118231

Schwarz, N. (2007). Retrospective and concurrent self-reports: The rationale for real-time data capture. In A. A. Stone, S. Shiffman, A. A. Atienza, & L. Nebeling (Eds.), *The science of real-time data capture: Self-reports in health research* (pp. 11–26). New York, NY: Oxford University Press, USA.

Selm, M. V., & Jankowski, N. W. (2006). Conducting online surveys. *Quality and Quantity*, *40*(3), 435–456. https://doi.org/10.1007/s11135-005-8081-8

Sprecher, S., & Felmlee, D. (1997). The balance of power in romantic heterosexual couples over time from "his" and "her" perspectives. *Sex Roles*, *37*(5–6), 361–379. https://doi.org/10.1023/A:1025601423031

Sue, V. M., & Ritter, L. A. (2011). *Conducting online surveys* (2nd ed.). Thousand Oaks, CA: Sage.

Sveidqvist, K. (2018). Mermaid (Version 0.5.0).

Turner, C. F., Ku, L., Rogers, S. M., Lindberg, L. D., Pleck, J. H., & Sonenstein, F. L. (1998). Adolescent sexual behavior, drug use, and violence: Increased reporting with computer survey technology. *Science*, *280*(5365), 867–873. https://doi.org/10.1126/science.280.5365.867

Wieczorek, L. L. (2018). *Make a match: The role of partner and power preferences in the consistency of partner choice* (unpublished master's thesis). Georg-August-University, Goettingen, Germany.

Wright, K. B. (2005). Researching internet-based populations: Advantages and disadvantages of online survey research, online questionnaire authoring software packages, and web survey services. *Journal of Computer-Mediated Communication*, *10*(3), 00–00. https://doi.org/10.1111/j.1083-6101.2005.tb00259.x

Yoshihama, M., Gillespie, B., Hammock, A. C., Belli, R. F., & Tolman, R. M. (2005). Does the life history calendar method facilitate the recall of intimate partner violence? Comparison of two methods of data collection. *Social Work Research*, *29*(3), 151–163. https://doi.org/10.1093/swr/29.3.151

How to cite this article: Wieczorek LL, Tata CS, Penke L, Gerlach TM. Online implementation of an event history calendar with formr: A tutorial. *Pers Relationship*. 2020;1–33. https://doi.org/10.1111/pere.12305