5

Representing observational data

5.1 Representing versus recording

When observational data are being recorded, it is reasonable that practical concerns dictate what is done. One uses whatever equipment is available and convenient. One records in ways that are easy and natural for the human observers. One tries to preserve in some form the information that seems important. Thus recorded data can appear in literally a multitude of forms. Some of these were described in chapter 3; however, we recognize that what investigators actually do often combines the simple forms described there into more complex, hybrid forms.

However, the form used for data recording – the data as collected – should not become a straitjacket for the analysis to follow. A format that works well for data recording may not work so well for data analysis, and a format that works well for one analysis may not work well for another analysis. The solution is to figure out simple ways to represent data, ways that make different kinds of analysis simple and straightforward. There is nothing especially sacrosanct about the form of the recorded data, after all, and there is no merit in preserving that form when it proves awkward for subsequent uses.

It would be very useful if just a few relatively standard forms for representing observational data could be defined. Not only would this help to standardize terminology with respect to sequential analysis, thus facilitating communication, it would also make analysis easier and would facilitate designing and writing general-purpose computer programs for sequential analysis. We assume that most investigators use computers for their data analysis tasks, but even if they do not, we think that representing the data by use of one (or more) of the five standard forms defined in this chapter will make both thinking about and doing data analysis a simpler and more straightforward affair. Further, there is nothing exclusive about these five forms. Depending on how data were recorded, investigators can, and probably often will, extract different representations from the same recorded data for different purposes.

The first four forms are defined by Bakeman and Quera's (1992, 1995a) Sequential Data Interchange Standard (SDIS), which defines a standard form for event, state, timed-event, and interval sequences, respectively. Sequential data represented by any of these forms can be analyzed with the Generalized Sequential Querier (GSEQ; Bakeman & Quera, 1995a). The fifth form is an application of the standard cases by variables rectangular matrix and is useful for analyzing cross-classified events, including contingency table data produced by the GSEQ program.

5.2 Event sequences

The simplest way to represent sequential behavior is as event sequences. As an example, imagine that observers used the following mnemonic codes for the parallel play study described in chapter 1: Un = Unoccupied, Sol = Solitary, Tg = Together, Par = Parallel, and Gr = Group. A child is observed for just a minute or two. First she is unoccupied, then she shifts into together play, than back to unoccupied, back to together, then into solitary play, and back to together again. If each code represents a behavioral state, then the event sequence data for this brief observation would look like this:

Un Tg Un Tg Sol Tg . . .

In this case, there are no Par or Gr codes because the child was not observed in parallel or group play during this observation session.

The data might have been recorded directly in this form or they might have been recorded in a more complex form and only later reduced to event sequences. The behavior to be coded could be thought of as behavioral states, as in this example, or as discrete events. Behavior other than just the states or events that form the event sequences might have been recorded but ignored when forming event sequences for subsequent analysis. In general, if the investigator can define a set of mutually exclusive and exhaustive codes of interest, and if the sequence in which those codes occurred can be extracted from the data as recorded, and if reasonable continuity between successive codes can be assumed (as discussed in section 3.4), then some (if not all) of the information present in the data can be represented as event sequences.

This is often very desirable to do. For one thing, the form is very simple. A single stream of codes is presented without any information concerning time, whether onsets or offsets. Lines of an event-sequential data file consist simply of codes for the various events, ordered as they occurred in time, along with information identifying the participant(s) and sessions. Thus event sequential data are appropriate when observed behavior is reduced

to a single stream of coded events (which are thus mutually exclusive and exhaustive by definition), and when information about time (such as the duration of events) is not of interest. Event sequences are both simple and limited. Yet applying techniques described in chapter 7 to event sequences, Bakeman and Brownlee were able to conclude that parallel play often preceded group play among 3-year-old children.

5.3 State sequences

For some analyses, the duration of particular behavioral states or events may matter, either because the investigator wants to know how long a particular kind of event lasted on the average or because the investigator wants to know what proportion of the observation time was devoted to a particular kind of event. For example, it may be important to know that the mean length for a bout of parallel play was 34 seconds and that children spent 28% of their time engaged in parallel play, on the average. In such cases, the form in which data are represented needs to include information about how long each event or behavioral state lasted.

As we define matters, state sequences are identical to time sequences with the simple addition of timing information. The terminology is somewhat arbitrary, and in the next section we discuss timed-event sequences, but our intent is to provide a few simple forms for representing sequential data, some of which are simpler than others, so that investigators can choose a form that is no more complex than required for their work. For example, if duration were important, the SDIS state-sequential representation for the sequence given earlier would be:

Un = 12 Tg = 8 Un = 21 Tg = 11
Sol = 34 Tg = 6 ···

Assuming a time unit of a second, this indicates that Unoccupied lasted 12 seconds, followed by 8 seconds of Together, 21 more seconds of Unoccupied, etc. The same sequence can also be represented as:

Un,8:01 Tg,8:13 Un,8:21
Tg,8:42 Sol,8:53 Tg,9:27 ,9:33 ···

which indicates an onset time for Unoccupied of 8 minutes and 1 second, for the first Together of 8 minutes and 13 seconds, etc. The offset time for the session is 9 minutes and 33 seconds; because the first onset time was 8:01, the entire session lasted 92 seconds.

State sequences, like simple event sequences, provide a useful way to represent aspects of the recorded data when a single stream of mutually

exclusive and exhaustive (ME&E) coded states (or events) captures information of interest. It would be used instead of (or in addition to) state sequences when information concerning proportion of time devoted to a behavior (e.g., percentage of time spent in parallel play) or other timing information (e.g., average bout length for group play) is desired. Additionally, it is possible to define multiple streams of ME&E states; for details see Bakeman and Quera (1995a). In sum, both simple event and state sequences are useful for identifying sequential patterns, given a simple ME&E coding scheme. But they are not useful for identifying concurrent patterns (unless multiple streams of states are defined) or for answering relatively specific questions, given more complex coding schemes. In such cases, the timed-event sequences described in the next section may be more useful.

5.4 Timed-event sequences

If codes can cooccur, and if their onset and offset times were recorded, then the data as collected can be represented as timed-event sequences. This is a useful and general-purpose format. Once data are represented in this form, an investigator can determine quite easily such things as how often specific behavioral codes cooccur (does the baby smile mainly when the mother is looking at him or her?), or whether certain behavioral codes tend to follow (or precede) other codes in systematic ways (does the mother respond to her baby's smiling within 5 seconds?).

For example, imagine that two people engaged in conversation were videotaped, and four codes were defined: *Alook,* meaning person A looks at person B; *Blook*, meaning B looks at A; *Atalk*, and *Btalk*. A brief segment of the coded conversation, depicted as though it had been recorded with an old-fashioned, deflected-pen, rolling-paper event recorder, is shown in Figure 5.1. Following SDIS conventions for timed-event sequences, this same segment would be represented as follows:

,1 Alook,2–4 Atalk,3–10 Blook,4–7 Alook,6
Blook,8–10 Alook,12–16 Btalk,12–19
Blook,13 Alook,17–20 Atalk18 , 21

Assuming time units are a second, this session began at second 1 and ended at second 21; thus the session lasted 20 seconds. Person A first began looking at second 2 and stopped at second 4; thus A's first looking bout lasted 2 seconds, etc. By convention, when offset times are omitted, durations are assumed to be one time unit; thus *Alook, 6* implies an offset time of 7. As you can see, offset times are assumed to be exclusive, but

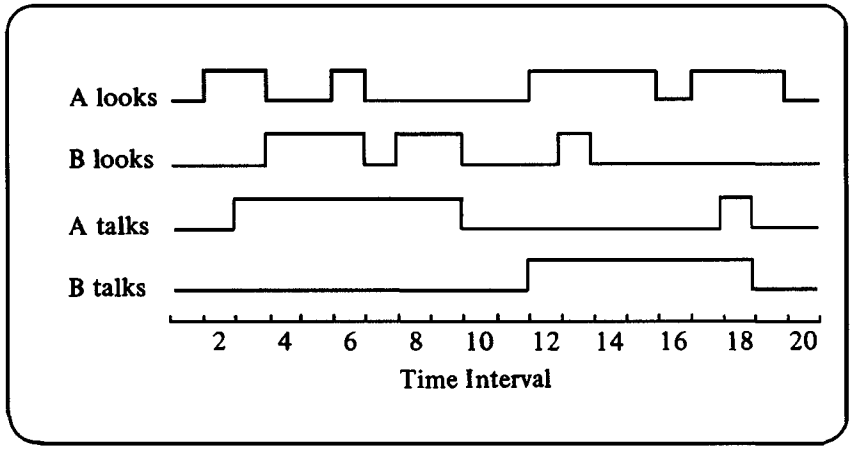

Figure 5.1. An example of the kind of output produced by a deflected-pen event recorder.

SDIS conventions also allow inclusive offset times (signaled by a right parenthesis). Thus the segment just given could also be represented as:

,1 Alook, 2–3) Atalk, 3–9) Blook, 4 – 6) Alook, 6
Blook, 8–9) Alook, 12–15) Btalk, 12–18)
Blook, 13 Alook, 17–19) Atalk, 18 ,20)

which some investigators may find more convenient. Other conventions for timed-event sequences, including inferred offset times and context codes, are detailed in Bakeman and Quera (1995a).

The flexibility of the timed-event format is very useful. As in other SDIS formats, codes can be defined for any number of behaviors (although, as a practical matter, the SDIS program limits the number to 95), but unlike event sequence data, timed-event sequential data (TSD) preserves onset and offset times, and unlike a single stream of state sequences, TSD preserves cooccurrences among behaviors. This allows for the level of complexity represented in the data to approach "real life," and it certainly imposes a minimum of constraints on the questions investigators can ask.

5.5 Interval sequences

The fourth format defined by SDIS, interval sequences, is designed to accommodate interval recording in a simple and straightforward way. Codes

are simply listed as they occur and interval boundaries are represented by commas. For example:

```
Interval = 5;
, , Ivoc, Ivoc, Aofr, Rain Aofr Ivoc Ismi, Rain, ...
```

indicates that intervals have a duration of 5 time units. No behavior is coded for the first two intervals. The infant vocalizes in intervals 3 and 4 and an adult offers in interval 5. The adult continues to (or again) offers in interval 6, at which time the infant both vocalizes and smiles and it begins to rain. The rain continues in interval 7, etc. Other conventions for interval sequences, including context codes, are detailed in Bakeman and Quera (1995a).

If the interval width for interval sequences were 1 second, for example, and if the time unit for timed-event sequences were likewise 1 second, then the same observed sequence could be represented with either timed-event sequences or interval sequences. Both allow behavioral codes to cooccur. But when an investigator begins with an interval recording strategy, it is often easier (and requires fewer key strokes) to represent such data directly as interval sequences, which is why we included this format as part of SDIS.

Many computer-based data collection devices and programs in common use produce data in their own format. For example, timed-event sequences in SDIS place the code first, followed by a comma, followed by the time the code occurred, whereas at least a few data collection systems with which we are familiar place the time before the code. This is hardly problematic. In such cases, it is relatively easy to write a simple computer program (in Pascal, for example) that reformats the data as initially collected into SDIS format.

No matter the particular form – whether event, state, timed-event, or interval sequences – the advantages of a few standard forms for sequential data are considerable. They make it easier, and more worthwhile, to develop general-purpose programs for sequential data – such as GSEQ (Bakeman & Quera, 1995a) – that compute, not just simple frequencies and percentages for different codes, but a variety of conditional probabilities and sequential and other statistics as well. Such general-purpose programs can then be shared among different laboratories. In addition, relying on a standard form for data representation should enhance the development of special-purpose software within a given laboratory as well.

5.6 Cross-classified events

In chapter 3, we distinguished between two general approaches to data collection: intermittent versus continuous recording. Because this is a book about sequential analysis, we stressed continuous recording strategies, defining four particular ones: (a) coding events, (b) recording onset and offset times, (c) timing pattern changes, and (d) coding intervals. We also said that a fifth strategy – cross-classifying events – could result in sequential data if the major categories used to cross-classify the event could be arranged in a clear temporal order.

When continuous recording strategies are used, there is some choice as to how data should be represented. When events are cross-classified, however, there seems only one obvious way to do it: Each line represents an event, each column a major category. For example, as described in section 2.13, Bakeman and Brownlee coded object struggles. The first major category was prior possession, the second was resistance, and the third was success. Thus (if 1 = yes, 2 = no) the following

```
1 1 1
2 1 2
```

would code two events. In the first, the child attempting to take the object had had prior possession (1), his take attempt was resisted (1), but he succeeded in taking the object (1). In the second, the attempted taker had not had prior possession (2), he was likewise resisted (1), and in this case, the other child retained possession (2).

Unlike the SDIS data formats discussed in the previous several sections, data files that contain cross-classified event data are no different from the usual cases by variables rectangular data files analyzed by the standard statistical packages such as SPSS and SAS. Typically, cross-classified data are next subjected to log-linear analyses. When events have been detected and cross-classified in the first place, the data file could be passed, more or less unaltered, to the log-linear routines within SPSS or other standard packages, or to a program like ILOG (Bakeman & Robinson, 1994) designed specifically for log-linear analysis. Moreover, often analyses of SDIS data result in contingency table data, which likewise can be subjected to log-linear analysis with any of the standard log-linear programs. In fact, GSEQ is designed to examine sequential data and produce contingency table summaries in a highly flexible way; consequently it allows for the export of such data into files that can subsequently be read by SPSS, ILOG, or other programs.

Table 5.1. *Relationship between data recording and data representation*

This data recording strategy	Allows for this data representation
Coding events, no time	Events sequences
Recording onset and offset times, or timing pattern changes, or coding intervals	Events, state, timed-event, or interval sequences
Cross-classifying events	Cross-classified events

5.7 Transforming representations

Early in this chapter, we suggested that the data as collected should not become a straitjacket for subsequent analyses; that it was important to put the data into a form convenient for analysis. A corollary is that the data as collected may take various forms for different analyses and that one form may be transformed into another. There are limits on possible transmutations, of course. Silken detail cannot be extracted from sow-ear coarseness. Still, especially when onset and offset times have been recorded, the data as collected can be treated as a data "gold mine" from which pieces in various forms can be extracted, tailored to particular analyses.

The relationship between data recording strategies and data representation form is presented in Table 5.1. As can be seen, when onset and offset or pattern-change times are recorded, aspects of that collected data can be represented as event, state, timed-event, or interval sequences. Further data represented initially as state or timed-event sequences can be transformed into event or even interval sequences; again, see Bakeman and Quera (1995a) for details.

One example of the potential usefulness of data transformation is provided by Bakeman and Brown's (1977) study of early mother–infant interaction. Desiring to establish an "ethogram" of early interactive behavior in the context of infant feeding, they defined an extensive number of detailed behavioral codes, more than 40 for the infant and 60 for the mother. Some of these were duration events, some momentary, the whole represented as timed-event sequences. For one series of analyses, Bakeman and Brown wanted to examine the usefulness of viewing interaction as a "behavioral dialogue." To this end, they defined some of the mother codes and some of the infant codes as representing "communicative acts," actions that seemed potentially communicative or important to the partner.

This done, they then proceeded to extract interval sequences from the timed-event sequential data. Each successive interval (they used a 5-second

interval, but any "width" interval could have been used) was categorized as follows: (a) interval contains neither mother nor infant "communicative act" codes, (b) interval contains some mother but no infant codes, (c) interval contains some infant but no mother codes, and (d) interval contains both mother and infant codes. This scheme was inspired by others who had investigated adult talk (Jaffe & Feldstein, 1970) and infant vocalization and gaze (Stern, 1974). With it, Bakeman and Brown were able to show, in a subsequent study, differences between mothers interacting with preterm and full-term infants (Brown & Bakeman, 1980). But the point for now is to raise the possibility, and suggest the usefulness, of extracting more than one representation from the data as originally recorded.

A second example of data transformation is provided by the Bakeman and Brownlee (1980) study of parallel play described in chapter 1. There an interval recording strategy was used. Each successive 15-second interval was coded for predominant play state as follows: (a) Unoccupied, (b) Solitary, (c) Together, (d) Parallel, or (e) Group play. Thus the data as collected were already in interval sequential form and were analyzed in this form in order to determine percentage of intervals assigned to the different play states.

However, to determine if these play states were sequenced in any systematic way, Bakeman and Brownlee transformed the interval sequence data into event sequences, arguing that they were concerned with which play states followed other states, not with how long the preceding or following states lasted. But once again, the moral is that for different questions, different representations of the data are appropriate.

Throughout this book, the emphasis has been on nominal-scale measurement or categorization. Our usual assumption is that some entity–an event or a time interval – is assigned some code defined by the investigator's coding scheme. But quantitative measurement can also be useful, although such data call for analytic techniques not discussed here. (Such techniques are discussed in Gottman, 1981.) The purpose of this final example of data transformation is to show how categorical data can be transformed into quantitative time-series data.

Tronick, Brazelton, and their co-workers have been interested in the rhythmic and apparently reciprocal way in which periods of attention and nonattention, of quiet and excitation, seem to mesh and merge with each other in the face-to-face interaction of mothers with their young infants (e.g., Tronick, Als, & Brazelton, 1977). They videotaped mothers and infants interacting and then subjected those tapes to painstaking coding, using an interval coding strategy. Several major categories were defined, each containing a number of different codes. The major categories included, for example, vocalizations, facial expressions, gaze directions, and body

movement for both mother and infant. The tapes were viewed repeatedly, often in slow motion. After each second of real time, the observers would decide on the appropriate code for each of the major categories. The end result was interval sequential data, with each interval representing 1 second and each containing a specific code for each major category.

Next, each code within each of the major categories was assigned a number, or weight, reflecting the amount of involvement (negative or positive) Tronick thought that code represented. In effect, the codes within each major category were ordered and scaled. Then, the weights for each category were summed for each second. This was done separately for mother and infant codes so that the final result was two parallel strings of numbers, or two times series, in which each number represented either the mother's or infant's degree of involvement for that second. Now analyzing two time series for mutual influence is a fairly classic problem, more so in astronomy and economics than psychology, but transforming observational data in this way allowed Gottman and Ringland (1981) to test directly and quantitatively the notion that mother and infant were mutually influencing each other.

5.8 Summary

Five standard forms for representing observational data are presented here. The first, event sequences, consists simply of codes for the events, ordered as they occurred. The second, state sequences, adds onset times so that information such as proportions of time devoted to different codes and average bout durations can be computed. The third, timed-event sequences, allows for events to cooccur and is more open-ended; momentary and duration behaviors are indicated along with their onset and offset times, as required. The fourth, interval sequences, provides a convenient way to represent interval recorded data. And the fifth form is for cross-classified events.

An important point to keep in mind is that data as collected can be represented in various ways, depending on the needs of a particular analysis. Several examples of this were presented in the last section. The final example, in fact, suggested a sixth data representation form: time series. Ways to analyze all six forms of data are discussed in the next chapters, although the emphasis is on the first five. (For time-series analyses, see Gottman, 1981.) Some of the analyses can be done by hand, but most are facilitated by using computers. An advantage of casting data into these standard forms is that such standardization facilitates the development and sharing of computer software to do the sorts of sequential analyses described throughout the rest of this book. Indeed, GSEQ (Bakeman & Quera, 1995a) was developed to analyze sequential data represented according to SDIS conventions.