Behavior Research Methods
2008, 40 (1), 21-32
doi: 10.3758/BRM.40.1.21

RAP: A computer program for exploring similarities in behavior sequences using random projections

VICENÇ QUERA
Universidad de Barcelona, Barcelona, Spain

A computer program (*RAP*, for random projection) for exploring similarities between and within sequences of behavior is presented. Given a time window of a sequence, the program calculates a *signature*, a real-valued vector that is a random projection of the contents of the window (i.e., the codes occurring within it and their relative location, or onset and offset times) into an arbitrary *K*-dimensional space. Then, given two different time windows from the same sequence or from different sequences, their similarity is computed as an inverse function of the Euclidean distance between their respective signatures. By defining moving (overlapped or not overlapped) windows along each sequence and calculating similarities between every pair of windows from the two sequences, a map of similarities or possible recurrent patterns is obtained; the RAP program represents them as gray-level lattices, which are displayed as mouse-sensitive images in an HTML file. Computation of similarities is based on the random projection method, as presented by Mannila and Seppänen (2001), for the analysis of sequences of events. The program reads sequence data files in Sequential Data Interchange Standard (SDIS) format (Bakeman & Quera, 1992, 1995a).

A pattern in a behavior sequence exists when there tend to be identical or similar repetitions of certain portions of it, which makes it possible to predict future behaviors from past ones, within some probability interval. Common methods for detecting such patterns include the fitting of Markov models in continuous and discrete time (e.g., Gardner & Hartmann, 1984), survival analysis in continuous time (e.g., Griffin & Gardner, 1989; Stoolmiller & Snyder, 2006), lag-sequential analysis (e.g., Bakeman, 1978; Bakeman & Quera, 1995a; Sackett, 1979), log-linear model fitting to multidimensional contingency lag tables (e.g., Bakeman, Adamson, & Strisik, 1995; Bakeman & Quera, 1995b), and, more recently, analysis of sequence organization based on proximity coefficients among behavioral codes (Taylor, 2006). What these methods have in common is (1) the aim of summarizing relationships in the sequences by means of quantitative global measures, and (2) the use of asymptotic statistical techniques for obtaining *p* values that indicate whether or not the sequences contain patterns, and for pointing to statistically significant temporal relationships among behavioral codes.

Alternative methods for the analysis of sequences have been developed that provide information about similarities within or between sequences, and many of them allow for visual exploration of possible patterns as well. These methods can be viewed as complementary to those described above, since it is usually highly advisable to explore the sequences of behavior for possible patterns before carrying out an analysis based on inferential statistics. Methods that search for similarities and repetitions in sequences abound in biological sequence analysis, which consists mainly of aligning nucleotide sequences by means of dynamic programming algorithms (e.g., Durbin, Eddy, Krogh, & Mitchison, 1998); these algorithms have also been applied when researchers have been searching for similarities between sequences of sociological events (Abbott & Barman, 1997), aligning action protocols collected in human–computer interaction settings (Fu, 2001), and calculating observer agreement for event sequences (Quera, Bakeman, & Gnisci, 2007). Specific algorithms have been proposed for searching for repeated patterns in the analysis of interactive behavior (T-patterns, Theme software; Magnusson, 2000), in genomic analysis (maximal repeats, REPuter software; Kurtz et al., 2001), and in the data mining of sequences of alarms in telecommunication networks (e.g., Mannila & Toivonen, 1996; Moen, 2000).

The use of dynamic-programming algorithms and repetition-searching algorithms in both cases requires extensive calculation; the former have the advantage of providing optimal alignments between sequences (i.e., alignments that minimize disagreements between them), whereas some of the latter (specifically, *Theme* and *REPuter*) yield visual displays indicating where patterns occur in the sequences being compared. Mannila and Seppänen (2001) proposed a method for exploring repetition of similar situations

V. Quera, vquera@ub.edu

(i.e., similar temporal organization of codes) in sequences of alarms in telecommunication networks. It is based on representing each code by a random vector in a space of an arbitrary number of dimensions and then defining time windows in the sequences and representing them by vectorial functions of the codes occurring within them and their relative locations; finally, Euclidean distances between the points representing the windows are computed. A similar method is used for calculating the distance between genes (e.g., Kang, 2005). When a certain number of codes are represented by random points in a space of arbitrary dimensions, it is said that codes are randomly projected into that space; if the number of dimensions of the space is lower than the number of codes, a dimensionality reduction is carried out. A dimensionality reduction is a powerful technique used in the analysis of multidimensional data that aims to represent a great number of variables by a subset of variables while capturing as much of the variation in the original data as possible. Applications of random projection include image and text processing (Achlioptas, 2003; Bingham & Mannila, 2001; Sahlgren, 2005), machine learning (Blum, 2006), and data mining in general. Although the most common purpose of random projection is reducing dimensionality while preserving essential properties of the data, "another, perhaps less intuitive scenario, is when projection to a lower-dimensional space actually *highlights* essential properties" (Vempala, 2004, p. 4). In this case, the purpose of using random projection to compute similarities between time windows in sequences is to highlight the parts of the sequences that tend to repeat and, thus, to highlight where patterns can probably be found.

In this article, a program is presented that computes similarities among sections defined in two sequences of behavior, according to the method outlined by Mannila and Seppänen (2001), and displays them as cells in a rectangular lattice, using gray levels to indicate degrees of similarity; this kind of graph is similar to the one used in dot matrix methods to explore similarities between amino acid sequences in genome analysis (e.g., States & Boguski, 1991). The program reads sequences of behavior from a data file and displays similarity lattices for pairs of sequences specified by the user. The program is written in Borland C++ and Delphi Pascal and runs on Windows systems.

BEHAVIOR SEQUENCES

The random projection (RAP) program accepts SDIS data files, a format for representing sequences of behavior that defines five types of data: event, state, timed event, interval, and multievent sequences (Bakeman, 2000; Bakeman, Deckner, & Quera, 2005; Bakeman & Quera, 1992, 1995a; Quera & Bakeman, 2000). Following is an overview of the first three data types.

Let C be a coding scheme, $\mathbf{C} = \{c_1, c_2, \ldots, c_M\}$, where c_i is a code representing a discrete behavioral state and M is the number of different mutually exclusive and exhaustive (ME&E) codes. For example, in a study on joint attention during adult–infant interaction, the infant's verbal behavior could be represented by this coding scheme (based on Mas, 2003): C = {O (the infant utters an onomatopoeia), W (the infant utters a word), P (the infant utters a phrase), N (the infant does not respond)}. Suppose that behavior was coded using this scheme and that only the order in which the codes occurred was considered relevant, but not their durations and onset times. During an observation session, an event sequence would have been obtained, S: $s_1\ s_2 \ldots s_n$, where s_i are codes belonging to C and n is the length of the sequence—for example, S: W W P N W W N O N.

However, if code durations or onset and offset times were considered important, the following timed sequence would have been obtained: S: $s_1,t_1\text{–}u_1\ s_2,t_2\text{–}u_2 \ldots s_n,t_n\text{–}u_n$, where dashes separate onset and offset times for each code (respectively, t_i and u_i), expressed in appropriate time units, and code s_i's duration is given by $d_i = u_i - t_i$. If the codes are ME&E, that is a state sequence in which the onset time for a code equals the offset time for the code that immediately precedes it (and therefore, $t_{i+1} > t_i$); total duration for sequence S equals $d = u_n - t_1$—for example, S: W,0–2 W,2–3 P,3–6 N,6–15 W,15–17 N,17–28 O,28–29 N,29–41, where time units are seconds. Note that in a state sequence, only onset times for the codes are in fact necessary, and the offset time for the last code must be explicitly stated.

If not all codes in C are mutually exclusive (i.e., some of them may occur simultaneously) or if they are not exhaustive (i.e., some time units have no codes assigned), they can be represented as a timed event sequence. In the example above, if infant gestural codes are included and code N is removed, the coding scheme could be C = {O, W, P, S (simple, or spontaneous, gesture), R (relational gesture), J (joint gesture—e.g., pointing)}. In fact, two separate coding schemes could be defined for verbal and gestural behavior: $\mathbf{C}_V$ = {O, W, P} and $\mathbf{C}_G$ = {S, R, J}. Codes in each scheme are mutually exclusive, but gestures and verbal utterances are not, since they can occur simultaneously; also, since the infant may remain silent or not gesture during certain time periods, the codes are not exhaustive. An example of a timed event sequence based on these coding schemes is S: W,0–2 J,0–6 P,3–6 S,10–12 W,15–17 J,16–17 P,25–30. In the example, several codes may overlap in time. Therefore, in these cases, either $t_{i+1} \leq u_i$ or $t_{i+1} > u_i$. In the example, W,0–2 and J,0–6 overlap for 2 time units, whereas J,0–6 and P,3–6 overlap for 3 time units. There are also three different time gaps during which no code occurs, from 6 through 10, from 12 through 15, and from 17 through 25.

SIMILARITY BETWEEN TIME WINDOWS

In order to compute the similarity between two sections, or time windows, belonging to different sequences $\mathbf{S}_1$ and $\mathbf{S}_2$, the program first represents each section by a point in a K-dimensional space; then it computes the distance between the two points and transforms it into a similarity measure by means of an appropriate function. The steps will be detailed in the following paragraphs.

Mapping the Codes

For each code c_i in the coding scheme, a mapping vector $\rho_k(c_i)$ with K dimensions is defined, and its components are

Table 1
Two Alternative Sets of Mapping Vectors in 10 Dimensions for Six Infant Verbal and Gestural Behavioral Codes

Code	1	2	3	4	5	6	7	8	9	10
					A. 10-Dimensional Coordinates, Gaussian Distribution					
O	1.0972	0.9482	−1.6819	−0.6289	−0.2642	−0.2462	−0.1101	−2.2061	−0.8089	0.3262
W	−0.2128	0.4906	−0.6881	−2.3814	−0.2231	−1.2982	0.6998	−0.0371	0.8672	−0.9148
P	1.0315	−0.0886	0.8352	1.2860	0.9168	−0.5167	−1.9647	−0.2173	1.6315	−1.1443
S	−2.0327	0.8232	1.5112	−0.9692	−1.1082	−0.6330	−1.9665	−1.0093	−0.8230	1.2079
R	0.2854	0.9361	−0.3157	−0.4925	−0.3920	−1.0767	−0.5467	0.3619	0.5237	−1.7015
J	0.8646	0.6140	−0.7695	1.4214	0.2499	−0.4185	−0.3891	0.9090	−0.4800	−1.0040
					B. 10-Dimensional Coordinates, Three-Valued Sparse Distribution					
O	0.0000	−1.7321	0.0000	−1.7321	0.0000	0.0000	1.7321	−1.7321	0.0000	0.0000
W	0.0000	−1.7321	0.0000	1.7321	1.7321	0.0000	0.0000	−1.7321	0.0000	0.0000
P	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.7321	0.0000	−1.7321	0.0000
S	−1.7321	0.0000	−1.7321	0.0000	−1.7321	0.0000	0.0000	−1.7321	0.0000	0.0000
R	0.0000	−1.7321	0.0000	1.7321	−1.7321	0.0000	0.0000	−1.7321	0.0000	−1.7321
J	0.0000	−1.7321	0.0000	0.0000	−1.7321	1.7321	−1.7321	−1.7321	−1.7321	−1.7321

assigned random real values that are sampled from some distribution, as the Gaussian or unit normal distribution, with a mean of zero and a standard deviation of one. Vector components are coordinates in a K-dimensional space, and thus we say that codes are projected into that space, where each code is represented by one point. If $K < M$, the projection reduces the dimensionality of the data. Table 1A shows six mapping vectors in $K = 10$ dimensions for each of the codes in the infant verbal and gestural coding scheme; coordinate values were sampled from the normal unit distribution. For each code, the vector components define one point in a 10-D space. Other distributions are possible besides the normal unit one; Achlioptas (2003) proposed two sparse distributions: a distribution with only two possible values for the random variable, $x = +1$ or -1, with $p = 1/2$; and a distribution with only three values, $x = +\sqrt{3}$, 0, or $-\sqrt{3}$, with $p = 1/6$, 2/3, and 1/6, respectively. The sparse distributions have the advantage of being computationally simpler than the normal one and still provide sensible results. Table 1B shows six mapping vectors in $K = 10$ dimensions, sampled from the three-valued sparse distribution for each code in the infant verbal and gestural coding scheme. As Bingham and Mannila (2001) indicate, vectors $\rho_k(c_i)$ are not linearly independent, and thus the projections are not orthogonal; however, obtaining orthogonal vectors is computationally expensive, and "in a high-dimensional space, there exists a much larger number of almost orthogonal than orthogonal directions. Thus, vectors having random directions might be sufficiently close to orthogonal" (according to Hetch-Nielsen, 1994; the quote is from Bingham & Mannila, 2001, p. 246).

Defining Time Windows

Two sequence locations are specified in sequences S_1 and S_2, marking the onset of the two sections, and window width w is defined. Both the locations and the width are expressed in either events or time units, depending on the SDIS data type. For example, given the following timed event sequences,

S_1: W,0–2 R,1–6 P,3–6 S,10–12 W,15–17 S,16–17 P,25–30

and

S_2: W,5–8 R,8–9 P,9–13 S,15–18 W,20–23 J,22–23 R,24–25 P,31–33,

we can define an arbitrary window for the first sequence starting at 8 and another window for the second one starting at 12, and set $w = 25$ time units. Window contents are thus V_1: 8[S,10–12 W,15–17 S,16–17 P,25–30]33 and V_2: 12[S,15–18 W,20–23 J,22–23 R,24–25 P,31–33]37, where square brackets enclose the window contents, and the onset and offset times of the windows are indicated outside the brackets; offset times are exclusive.

Computing Window Signatures

A signature is computed for each time window by multiplying the components of the mapping vectors that correspond to each code occurrence within the window by a function of the relative locations of the codes; for window V_i ($i = 1, 2$), the kth coordinate is given by

$$y_{ik} = \sum_{j=1}^{R_i} \rho_k\left(s_{ij}\right) \cdot f\left(t_{ij}, u_{ij}\right),$$

where the sum is over the R_i code occurrences within the window V_i ($i = 1, 2$), s_{ij} is the jth code occurring in it, and t_{ij} and u_{ij} are its onset and offset times. The signature is thus a K-dimensional, real-valued vector that identifies the contents of the window quantitatively; in other words, the signature is a point in a K-dimensional space representing the contents of the window. The choice of K is arbitrary; when it is increased, high similarities stand out and low ones tend to fade away. No guideline exists as to which is the best value of K (see below); a minimum of $K = 5$ or $K = 10$ would seem to be advisable (Mannila & Seppänen, 2001). In matrix form, the signature for window V_i is defined as $\mathbf{y}_i = \mathbf{M}_i \cdot \mathbf{f}_i$, where $\mathbf{y}_i$ is the $K \times 1$ signature vector, $\mathbf{M}_i$ is a $K \times R$ matrix containing the components of the mapping vectors (rows) corresponding to each code occurrence within window V_i (columns), and $\mathbf{f}_i$ is an $R \times 1$ vector whose components are projection functions.

$$\mathbf{y}_1 = \mathbf{M}_1 \cdot \mathbf{f}_1 = \begin{bmatrix} y_{11} \\ y_{12} \\ y_{13} \\ y_{14} \\ y_{15} \\ y_{16} \\ y_{17} \\ y_{18} \\ y_{19} \\ y_{1,10} \end{bmatrix} = \begin{bmatrix} -2.033 & -0.213 & -2.033 & +1.031 \\ +0.823 & +0.491 & +0.823 & -0.089 \\ +1.511 & -0.688 & +1.511 & +0.835 \\ -0.969 & -2.381 & -0.969 & +1.286 \\ -1.108 & -0.223 & -1.108 & +0.917 \\ -0.633 & -1.298 & -0.633 & -0.517 \\ -1.966 & +0.700 & -1.966 & -1.965 \\ -1.009 & -0.037 & -1.009 & -0.217 \\ -0.823 & +0.867 & -0.823 & +1.631 \\ +1.208 & -0.915 & +1.208 & -1.144 \end{bmatrix} \cdot \begin{bmatrix} f(10,12) \\ f(15,17) \\ f(16,17) \\ f(25,30) \end{bmatrix}$$

Thus, from the mapping vectors shown in Table 1, the signature for window $\mathbf{V}_1$ is given in the matrix above. Columns in matrix $\mathbf{M}_1$ contain the mapping vectors for codes S, W, S, and P in the order in which they occur within window $\mathbf{V}_1$. A linear projection function (LPF) that takes into account the onset time can be used:

$$f\left(t_{ij}, u_{ij}\right) = \frac{T - t_{ij}}{w}, \qquad (1)$$

where T is the window offset time. Choosing T as the window offset time is arbitrary. Other points, such as the window onset time, could be chosen instead. In any case, what matters is that for each code occurrence, the function returns a value representing the relative position of the code's onset time within the window.

Equation 1 yields values that are greater than 0 and less than or equal to 1 (0 would indicate that the code occurs very near the window termination, whereas 1 would indicate that it occurs when the window starts). For example, for the occurrence W,15–17, the projection function yields (33 − 15)/25 = .72. The (transposed) signature $\mathbf{y}_1$ for window $\mathbf{V}_1$ equals

$$\begin{bmatrix} -3.075 & +1.642 & +2.190 & -2.854 & -1.640 \\ -2.113 & -3.271 & -1.711 & -0.170 & +0.908 \end{bmatrix}.$$

Likewise, in order to obtain the signature for window $\mathbf{V}_2$, a matrix $\mathbf{M}_2$ with $R = 5$ columns, one per code occurrence (S, W, J, R, P), and a vector $\mathbf{f}_2$ with five projection functions are necessary. The signature $\mathbf{y}_2$ is then

$$\begin{bmatrix} -1.019 & +1.892 & +0.436 & -1.567 & -0.961 \\ -2.375 & -2.244 & -0.232 & +0.241 & -1.321 \end{bmatrix}.$$

Alternatively, a projection function taking code durations into account, and not only their onset times, could be used by applying LPF to every time unit when the code occurs—that is,

$$f\left(t_{ij}, u_{ij}\right) = \sum_{m=t_{ij}}^{u_{ij}-1} (T - m) / w.$$

Other possible projection functions are the exponential, $f(t_{ij},u_{ij}) = \exp[(T - t_{ij})/w - 1]$ (Mannila & Seppänen, 2001), which weighs codes higher as their onset times approach the end of the window; the sequential order-projection function (SPF), $f(t_{ij},u_{ij}) = r$, which simply assigns each code occurrence its sequential position r within the window and ignores its onset time; and the occurrence function, $f(t_{ij},u_{ij}) = 1$, which weighs all codes occurring within the window equally and, thus, for which only the fact that a code occurs, but neither its sequential order nor its exact onset and offset times, is considered important.

Computing Window Similarity

Given the signatures for the two windows, the Euclidean distance between them is computed:

$$D\left(\mathbf{V}_1, \mathbf{V}_2\right) = \sqrt{\sum_{k=1}^{K} \left(y_{1k} - y_{2k}\right)^2}.$$

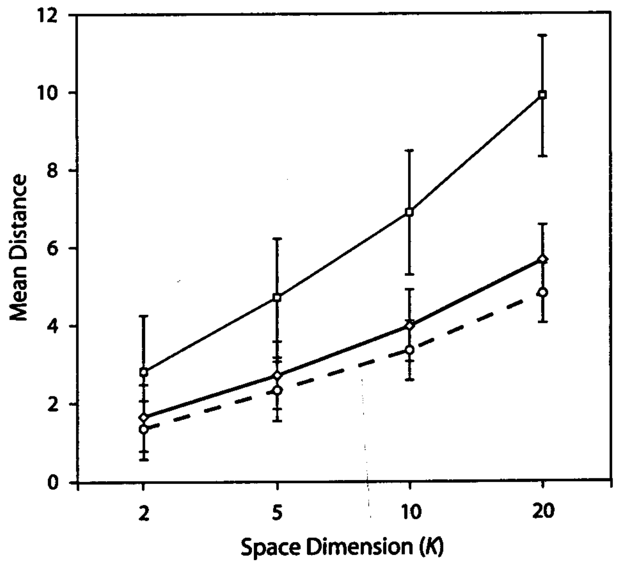


Figure 1. Mean distances among three windows ($\mathbf{V}_1$, $\mathbf{V}_2$, and $\mathbf{V}_3$; see the text), for space dimensions K = 2, 5, 10, and 20. For each K, mapping vectors were assigned random Gaussian coordinates 1,000 times independently, and distances were computed between the resulting window signatures. The top, middle, and bottom lines correspond to mean $D(\mathbf{V}_1, \mathbf{V}_3)$, $D(\mathbf{V}_2, \mathbf{V}_3)$, and $D(\mathbf{V}_1, \mathbf{V}_2)$, respectively; vertical segments indicate standard deviations.

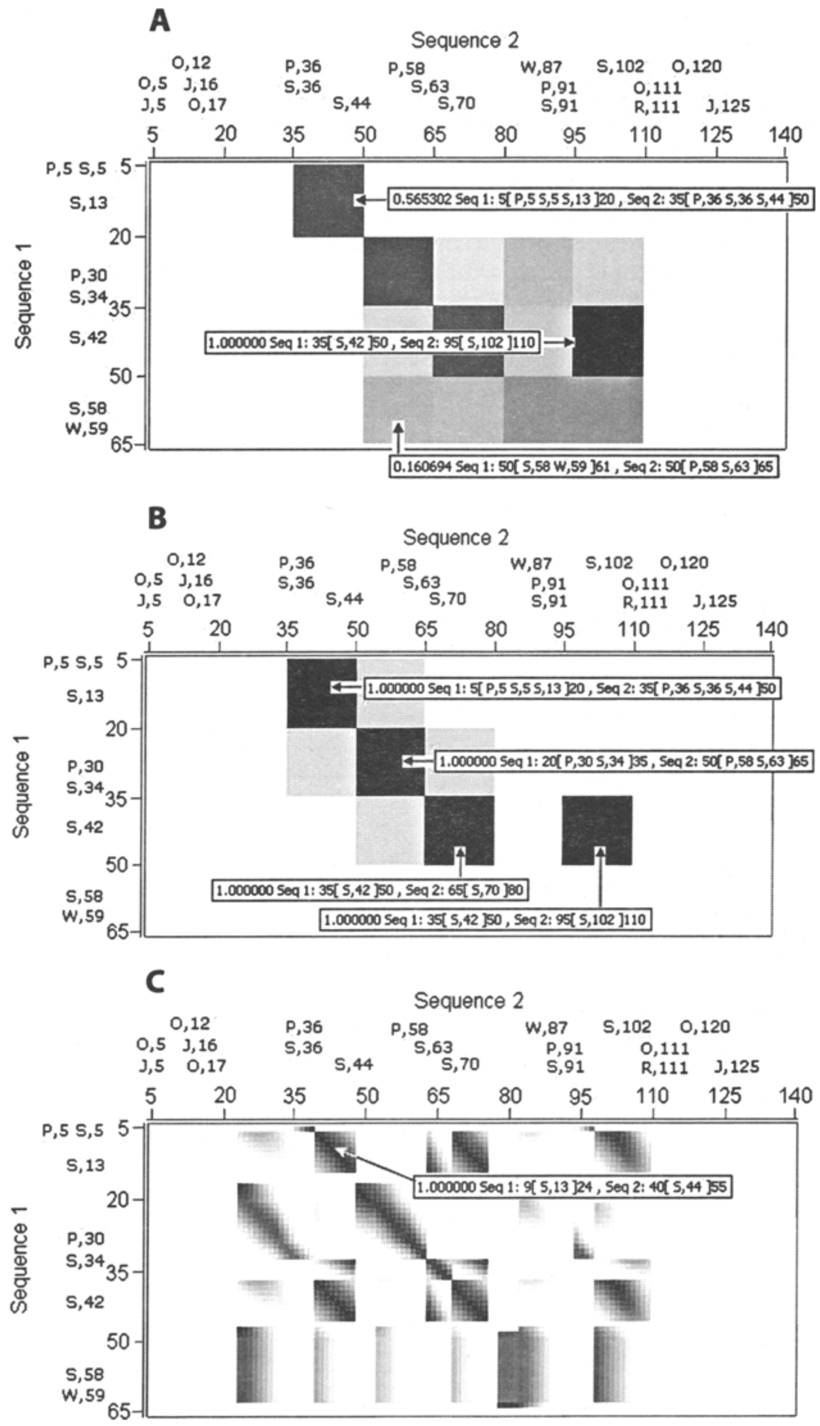


Figure 2. Similarity lattices for two timed event sequences of infant verbal and gestural behavior, obtained using a Gaussian random projection with $K = 10$ dimensions. Window width was set to 15, and starting times to 5 for both sequences. (A) Linear projection function, with shift equal to 15. (B) Sequential order projection function, with shift equal to 15. (C) Linear projection function, with shift equal to 1 and, thus, overlapping successive windows. Codes and their onset times are shown for each sequence. Similarities between several different pairs of time windows are indicated.

The distance between two windows with identical contents equals zero, whereas any differences regarding the code occurrences, their relative locations within the windows, or both (depending on the kind of projection function used) will increase the distance. Using LPF, the distance between the two previous windows is $D(V_1,V_2) = 4.235$. Distances can be converted into similarity measures by means of an inverse function, such as $G(V_1,V_2) = \exp[-D(V_1,V_2)]$. The negative exponential function is particularly useful because it highlights small distances by transforming them into large similarities, whereas big distances are transformed into negligible similarities; moreover, the resulting similarities are bounded between 0 and 1. Two windows with identical contents have $G = 1$, whereas any differences make G decrease toward 0. The similarity between the two previous windows is, thus, $G(V_1,V_2) = \exp(-4.235) = .014$.

Since the mapping vector coordinates are random values, resampling them will result in different window signatures. Therefore, the actual distances between windows are likely to vary if the mapping vectors are resampled. Figure 1 shows mean distances among three windows—V_3: 55[J,50–52 R,53 P,60–70 S,69]80, and previous V_1 and V_2—for different space dimensions (K = 2, 5, 10, and 20); for each K, mapping vectors were assigned random Gaussian coordinates 1,000 times independently, LPF was used, and distances were computed between the resulting window signatures. Mean distances increase when the dimensionality increases and consistently indicate that, on average, $D(V_1, V_3)$ is greater than both $D(V_2, V_3)$ and $D(V_1, V_2)$. However, the three distributions of distances largely overlap when K is low, as Figure 1 indicates. Percentages of cases in which $D(V_1, V_3) > D(V_2, V_3) > D(V_1, V_2)$ are 61.1%, 70.2%, 80.7%, and 87.4% for K = 2, 5, 10, and 20, respectively, whereas percentages of cases in which $D(V_1, V_3) > D(V_1, V_2) > D(V_2, V_3)$ are 31.4%, 28.7%, 19.3%, and 12.6%, respectively. Therefore, increasing the space dimensionality tends to remove overlapping and maintain the order of distances.

SIMILARITY LATTICES

In order to search for a defined pattern (a short-sequence S_1 with a total duration d_1) in a target sequence (a longer sequence S_2 with a total duration d_2, where $d_1 \ll d_2$), a time window of width $w = d_1$ is moved along the target sequence; each time the moving window is applied to S_2, a similarity is calculated between that window and S_1. If similarities were represented graphically against the onset time of the successive windows, peaks in the graph would indicate high similarity regions where the pattern sequence tends to appear. Successive moving windows can overlap or not; two overlapped windows in the target sequence will probably have almost identical similarities with respect to the pattern. In order to avoid such redundancy, it may be judged preferable to use nonoverlapping windows or windows that overlap in a small portion of their widths. It must, therefore, be specified how many time units (s) the successive windows must be shifted forward along the target sequence: (1) If $s < w$, successive windows will be overlapped; (2) if $s = w$, they will be concatenated and not overlapped; (3) if $s > w$, they will not be concatenated, and some regions in the target sequences will be skipped.

However, the exploration of sequences does not need to be directional; that is, given two sequences of comparable durations, we may be interested in exploring which sections in the first sequence are similar to sections in the second one, and vice versa. Even more, the two sequences can be the same sequence, and we may want to explore which sections in it tend to repeat. More generally, given two sequences S_1 and S_2 with total durations d_1 and d_2, window width w, and shift s, both much smaller than d_1 and d_2, and a starting time t_p for sequence p (p = 1, 2), a total of n_p successive windows can be explored for sequence p whose onset times are $t_p, t_p + s, t_p + 2s, \ldots t_p + (n_p - 1)s$ within the sequence; therefore, in total, $n_1 \cdot n_2$ similarities can be calculated. For example, given the following timed event sequences (where offset times are omitted when they equal the corresponding onset time plus one),

S_1: P,5–18 S,5 S,13–20 P,30–54 S,34–37 S,42–46 S,58–61 W,59–61

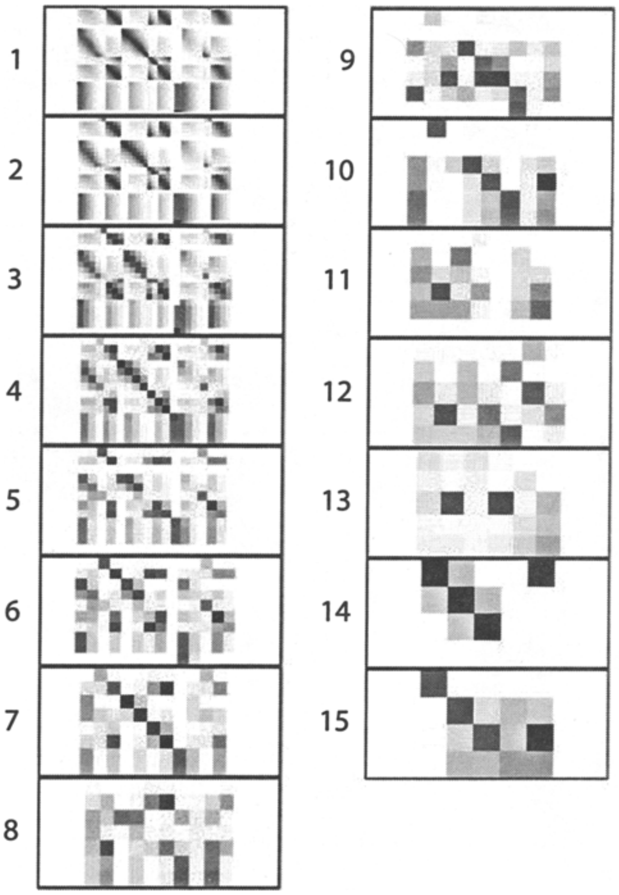


Figure 3. Similarity lattices for two timed event sequences of infant verbal and gestural behavior, obtained using a Gaussian random projection with K = 10 dimensions, and the linear projection function for a series of window widths, 1 through 15. Window shift was set to 1 for all cases; thus, successive windows overlap.

S_2: O,5 J,5 O,12 J,16–18 O,17–25 P,36–47 S,36 S,44–50 P,58–83 S,63–66 S,70–74 W,87–89 S,91–94 P,91–105 S,102–105 O,111 R,111–123 O,120–126 J,125–131,

we set $w = s = 15$ and starting times $t_1 = t_2 = 5$; therefore, the number of windows are $n_1 = 4$ and $n_2 = 9$, with onset times of 5, 20, 35, and 50 for sequence S_1 and 5, 20, 35, . . . , 125 for sequence S_2. In total, $4 \cdot 9 = 36$ similarities are computed. Similarities can be represented in a 3-D graph in which window onset times for sequences S_1 and S_2 correspond to the x- and y-axes, respectively, and window similarities correspond to the z-axis; alternatively, an $n_1 \times n_2$ lattice with cells representing similarity values as gray levels can be used.

Figure 2A shows the lattice obtained by the RAP program corresponding to the comparison between previous sequences S_1 and S_2, using LPF and Gaussian mapping vectors with $K = 10$. Each cell represents a similarity between two windows; the darker the cell, the higher the similarity; white cells indicate either a very low similarity or a pair of windows with no behavioral codes in common (e.g., the cell in the top left corner). In the figure, only the onset times of the codes are shown, since their offset times are disregarded in Equation 1; information about similarity and window contents is shown for three cells, with high, medium, and low similarities; a group of three diagonal cells with medium similarities can be seen, which correspond to three consecutive windows in both sequences containing identical codes in similar relative locations: windows 5[P,5 S,5 S,13]20 and 35[P,36 S,36 S,44]50; windows 20[P,30 S,34]35 and 50[P,58 S,63]65; and windows 35[S,42]60 and 65[S,70]80, from sequences S_1 and S_2, respectively. Figure 2B shows the similarity lattice for the same sequences, with $w = s = 15$ and starting times $t_1 = t_2 = 5$, using SPF. Similarities represented in Figure 2B tend to be more extreme than those in Figure 2A, because SPF is less strict than LPF; in Figure 2B, four cells indicate maximum similarity between windows containing codes that occur in identical orders within their corresponding windows.

Detection of similar patterns is highly dependent on the window width chosen. If w is big, as compared with the

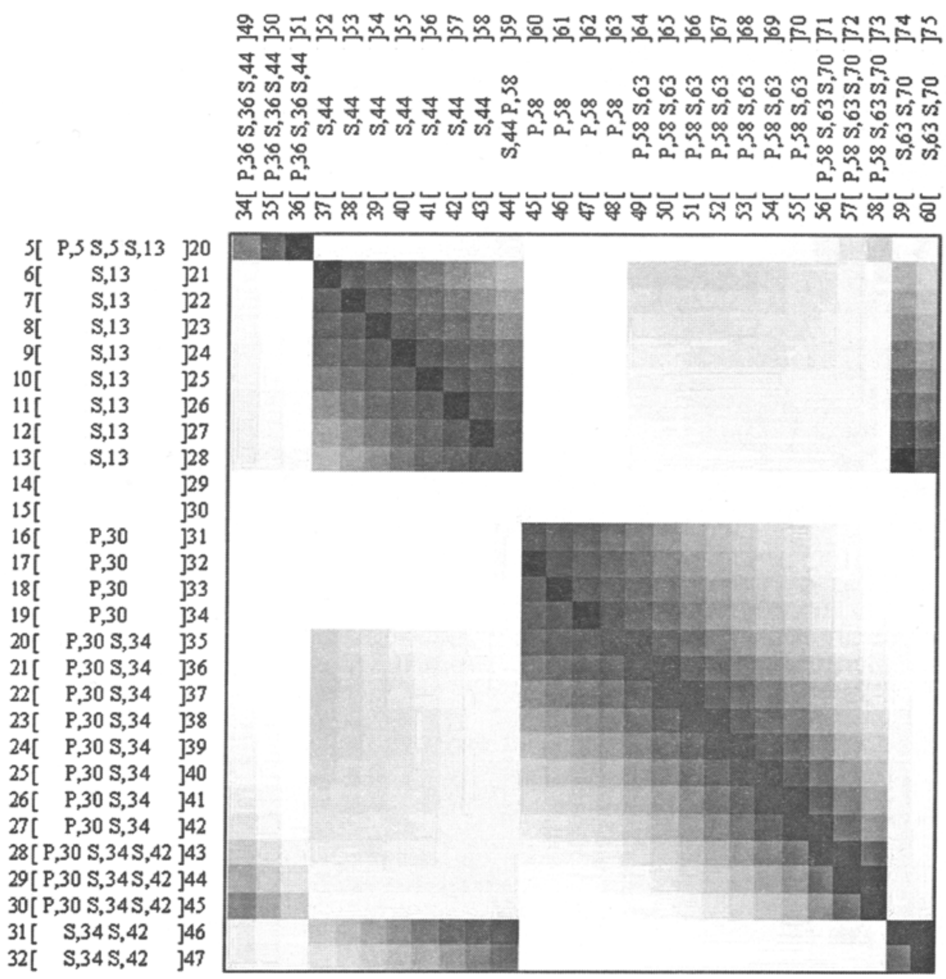


Figure 4. Similarity lattice, a zoomed section from Figure 2C. Rows and columns correspond to time windows in the first and second sequences, respectively. Window width was set to 15, and shift was set to 1; thus, successive windows overlap. Contents for each window are shown, including its onset and offset times, the codes occurring within the window, and their onset times.

total sequence duration, few pairs of windows in the two sequences are likely to contain identical or highly similar codes (with identical or highly similar relative onset times) unless the two sequences as a whole are identical or nearly identical; on the other hand, if w is small, the windows will probably contain few codes, and therefore, long patterns cannot be detected.

Overlapped Windows

As was stated above, successive overlapping windows are likely to contain very similar information. Figure 2C shows one lattice for previous sequences S_1 and S_2, obtained using LPF, with $w = 15$ and $s = 1$. Since successive windows overlap and are shifted one time unit, the number of windows for each sequence equals the total duration of the sequence (i.e., $n_1 = d_1 = 56$, and $n_2 = d_2 = 126$); therefore, both lattices are 56×126. Comparing Figures 2A and 2C, it is obvious that overlapping windows yield a much more precise representation of similarity. As the window shift increases, that fine-grained picture progressively disappears, as Figure 3 indicates; in that figure, lattices corresponding to $w = 15$, and $s = 1$ through 15 are displayed using LPF; lattice size decreases from 56×126 (top) to 4×9 (bottom).

By increasing shift, regular samples of the lattice corresponding to $s = 1$ are obtained. When $s = 2$, the 1st, 2nd, 3rd, and 4th windows of a sequence correspond to its 1st, 3rd, 5th, and 7th windows for $s = 1$; when $s = 3$, the

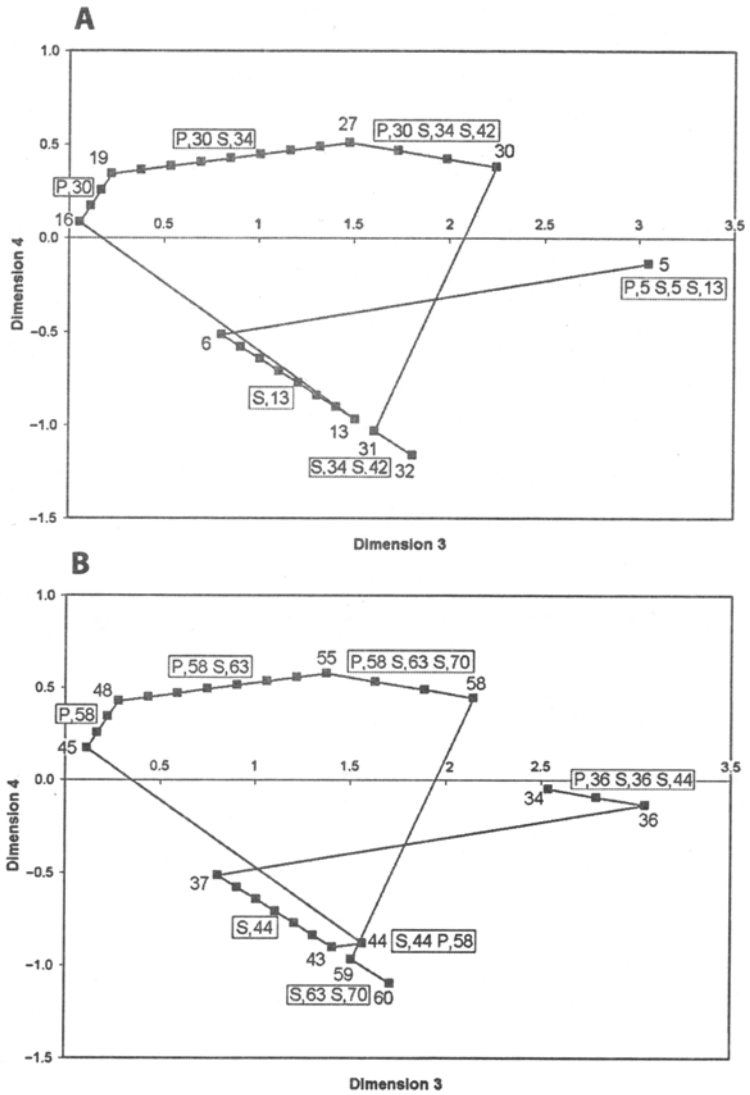


Figure 5. Signatures of two series of time windows, originally 10-D, projected onto a 2-D space. The time windows correspond to those compared in Figure 4. (A) Signatures for time windows in the first sequence, from window 5[P,5 S,5 S,13]20 through window 32[S,34 S,42]47. (B) Signatures for time windows in the second sequence, from window 34[P,36 S,36 S,44]49 through window 60[S,63 S,70]75. Distances between similar trajectories correspond to darker cells in Figure 4.

1st, 2nd, 3rd, and 4th windows correspond to the 1st, 4th, 7th, and 10th windows for $s = 1$, and so on. Consequently, given two shifts, s and s', where s' is a multiple of s, the lattice obtained for s' is a regular sample of the one obtained for s. For example, in Figure 3, the lattices for $s = 4$ are regular samples of those for $s = 2$, and the lattices for $s = 10$ are regular samples of those for $s = 5$.

Zooming In on the Lattice

As Figure 2C shows, when successive windows overlap and shift is small, smooth similarity gradients between neighbor cells are obtained when LPF is used. In particular, rectangular regions in the lattice with high similarities values along their diagonals are found. For example, a zoomed section of Figure 2C is displayed in Figure 4: from window 5[P,5 S,5 S,13]20 through window 32[S,34 S,42]47 (rows, sequence S_1), and from window 34[P,36 S,36 S,44]49 through window 60[S,63 S,70]75 (columns, sequence S_2). The upper left 8 × 8 square with grayed cells indicates similarities between windows containing event S,13 from sequence S_1 and event S,44 from sequence S_2. Black cells appear along the diagonal because the relative locations of code S within the respective row and column windows are identical: event S,13 is located 8 time units before the offset of window 6[S,13]21 (row), and likewise, event S,44 is located 8 time units before the offset of window 37[S,44]52 (column). Moreover, they are both located 7 time units before the offsets of windows 7[S,13]22 (row) and 38[S,44]53 (column), respectively, and so on. Gray off-diagonal cells indicate that the corresponding row and column windows contain the same events, but their locations within their windows are not identical; for instance, event S,13 is located 10 time units before the offset of window 7[S,13]23 (row), whereas event S,44 is located 13 time units before the offset of window 41[S,44]56 (column). The lower right diagonal cell in that 8 × 8 square is gray but not black, because window 44[S,44 P,58]59 (column) contains both events S,44 and P,58, whereas window 13[S,13]28 (row) contains only event S,13.

Figure 5 provides more insight as to how the similarities represented in Figure 4 are obtained from the window signatures. Signatures for the windows in sequence S_1, from 5[P,5 S,5 S,13]20 through 32[S,34 S,42]47, are represented as points in a 2-D space in Figure 5A; for simplicity, the 10-D signatures have been projected into the third and fourth dimensions of the space. Some points are identified by the onset time of the corresponding windows (5, 6, 13, 16, and so on), and groups of points on a single line are identified by the codes that are shared by all of them (e.g., S,13 for all points between those marked 6 and 13). Likewise, signatures for the windows in sequence S_2, from 34[P,36 S,36 S,44]49 through 60[S,63 S,70]75, are represented in Figure 5B. The similarities previously shown in Figure 4 had been transformed from the distances from every point in Figure 5A to every other point in Figure 5B (albeit taking all $K = 10$ dimensions into account); note that, as certain sections in the trajectories in Figures 5A and 5B are almost identical (e.g., points marked 6 to 13 in Figure 5A and points marked 37 to 43 in Figure 5B), the distances between them are small or even null, thus yielding high similarities (e.g., black cells along the diagonal of the upper left 8 × 8 square; Figure 4).

Exploring Event Sequences

Event sequences can be viewed as a particular case of timed event sequences in which one time unit corresponds to one behavior occurrence and every time unit contains one and only one code. As a result, an event sequence like W W P N W N O N can be alternatively represented as the following timed event sequence where times are actually sequential orders, W,1 W,2 P,3 N,4 W,5 N,6 O,7 N,8 (offset times are omitted). For such sequences, every time unit within a window will contain one code, which is unlikely for regular timed event sequences.

As an example, consider the following event sequences shown in Figure 6, which represent verbal interaction in a couple. Each verbal utterance of wife and husband is assigned one code, and only the order in which the

```
Event;
% Sequence 1
HApp WEmp WEmo HEmp WEmp WNeg WNeg HApp WEmo HEmo WEmp HEmp WCom HCom WCom HCom
WEmo HEmo WEmo HEmo WEmp HEmo WEmo HApp WEmp WEmo HApp HApp WApp HEmo WEmp HEmp
HEmo WEmo WEmo HApp HCom WNeg WNeg WCom HEmo WEmo HEmp WEmp HEmp WEmo HEmo WEmo
WEmo HEmp HNeg HOth HCom WOth WEmo WApp HApp WEmp WEmo HApp WEmp HEmp WCom HEmo
WEmo WCom HEmo WEmp HEmp WEmp HEmp WEmp HCom WCom HCom WNeg HCom WEmp HEmp WEmp
HEmp HEmo WEmo WEmp ;
% Sequence 2
WEmo HEmo HEmp WEmo WApp HApp WEmp HEmo WEmo HEmo WEmp HNeg WEmp HCom WCom HNeg
HCom HNeg WEmp HEmp WEmp HEmp WEmp HEmp WEmo HEmo WEmo HEmo WEmo HApp WApp HApp
HApp WEmp HEmp WCom HEmp WEmp HEmo HNeg HApp HEmp WEmp HEmo HApp WApp WApp HEmp
WApp HCom WCom HCom HNeg WEmo HEmo WEmo HEmp WEmp ;
% Sequence 3
HCom WEmo HEmo HEmp WEmp HApp WEmo HEmp WEmp HEmo WEmo HEmo WApp WApp HApp WEmp
HEmp WEmp WEmo HEmp HEmp WEmp WEmp HEmp WEmp HApp HApp WApp HApp WEmp HEmp HApp
WApp HApp WEmo HEmo WEmp HApp WApp HEmo HEmp WEmo HEmp WEmp HEmp WOth HApp WEmp
HApp WEmp HEmo HCom HEmp WEmp WEmo HEmo WEmo HEmp WEmp HEmp WEmp HEmp WEmo HEmo
WEmo /
```

Figure 6. Event sequences obtained for verbal interaction in a couple.

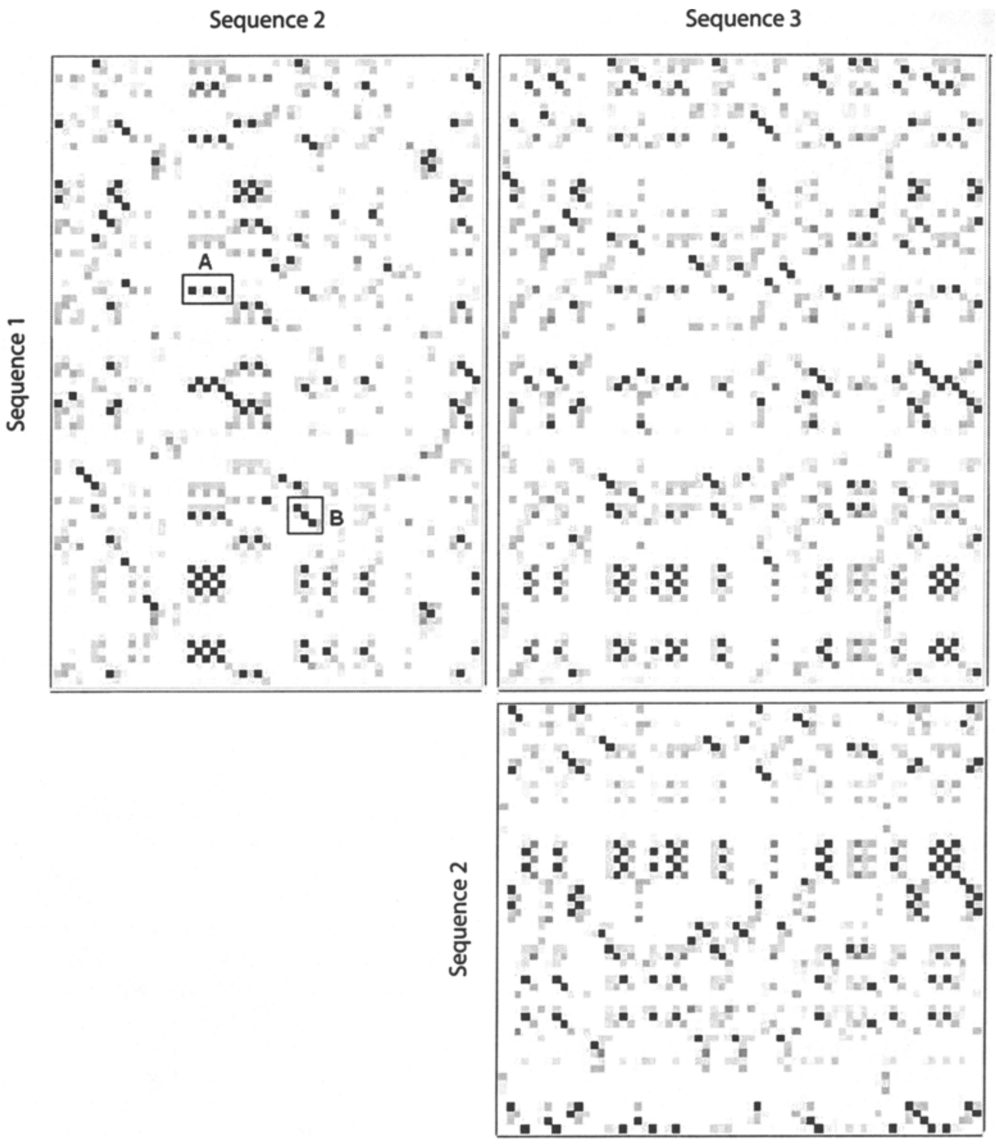


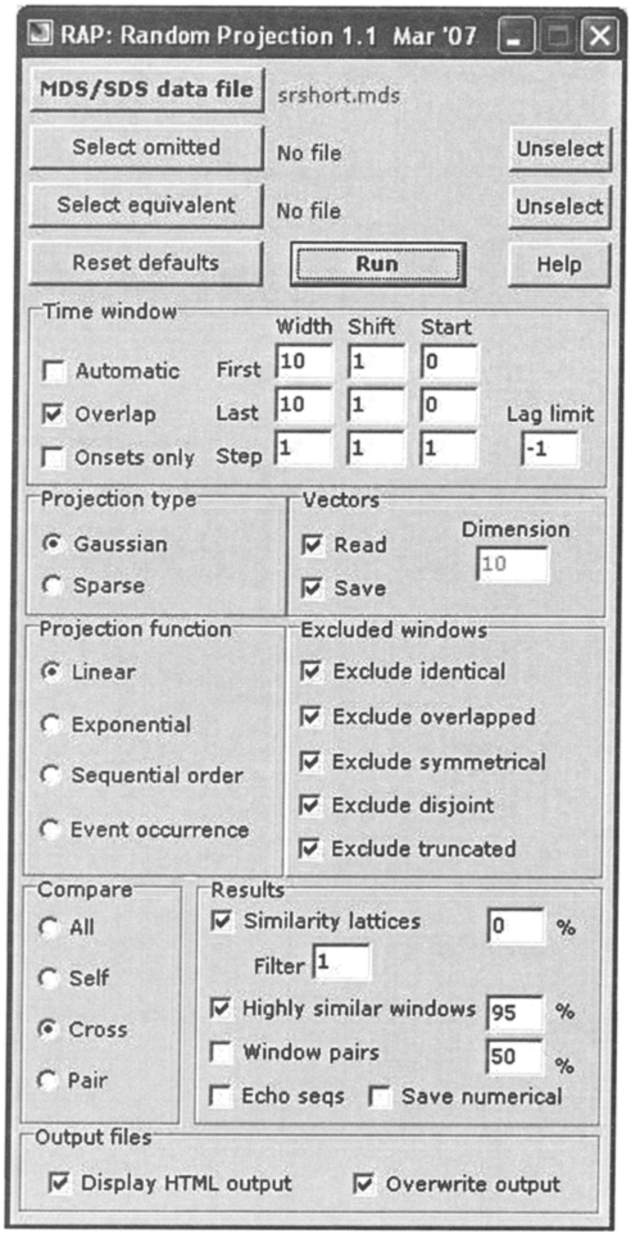
Figure 7. Similarity lattices comparing three event sequences of a couple's verbal interaction, obtained using a Gaussian random projection with $K = 10$ dimensions and the linear projection function. Window width was set to 2, and shift was set to 1; thus, successive windows overlap. See the text for an explanation.

codes occur is of interest. The codes are the following: WCom (wife complains), WEmo (wife emotes), WEmp (wife empathizes), WApp (wife approves), WNeg (wife negates), WOth (other wife utterances), HCom (husband complains), HEmo (husband emotes), HEmp (husband empathizes), HApp (husband approves), HNeg (husband negates), and HOth (other husband utterances). The couple was observed in three different occasions, and the sequences, shown in SDIS format in Figure 6, were obtained.

Similarity lattices for those sequences are displayed in Figure 7. Window width and shift are $w = 2, s = 1$; mapping vectors are Gaussian with $K = 10$, and LPF is used. Three lattices are shown: one for each comparison among the three sequences. For event sequences, using $w = 2$ has the effect of highlighting repetitions of pairs of codes that occur successively; for example, the pair WEmp HEmp occurs at Locations 31 and 32 in Sequence 1 and repeats at Locations 19 and 20, 21 and 22, and 23 and 24 in Sequence 2. These repetitions are indicated by three black cells in a horizontal line (A) in Figure 7. Strings of codes that are identical in both sequences are indicated by a group of diagonal cells; for example, the string HApp WEmp HEmp WCom occurs starting at Location 60 in Sequence 1 and at Location 33 in Sequence 2, and therefore, three overlapped pairs are identical in both sequences, HApp WEmp, WEmp HEmp, and HEmp WCom, as indicated by three diagonal black cells (B) in Figure 7.

RUNNING THE RAP PROGRAM

The RAP program can be downloaded from www.ub.es/comporta/vquera. Its user interface is a dialog window for selecting the data files and the parameters that run the process (Figure 8). Either a raw sequence SDIS data file (extension SDS) or a compiled data file (extension MDS) can be selected; if a raw data file is selected, RAP first invokes the SDIS compiler in order to compile it and create the corresponding MDS file. The SDIS compiler is a module originally in the GSEQ software (Bakeman & Quera, 1995a; www.ub.es/comporta/sg.htm) and is included in the RAP program as well. After selecting the data file and setting the parameters (window width and shift, space dimension, projection function, and so on), clicking on the Run button starts the computation. The user can optionally define groups of codes as equivalent (i.e., recode them as a single code) and can request that some codes be omitted from the computation of similarity; this may be helpful for focusing and enhancing the exploration. A help file with details about parameters and options is included with the program. Results are saved in HTML files; RAP launches the system's default Internet browser to display them. Similarity lattices are saved as BMP images, which are inserted in the HTML text, one image per pair of sequences being compared. Image maps for the BMP files are saved in the HTML code, making the images mouse sensitive. If requested, the program can apply a filter to the resulting images in order to highlight groups of contiguous lattice cells having high similarity values.


RAP: Random Projection 1.1 Mar '07

MDS/SDS data file: srshort.mds
Select omitted: No file
Select equivalent: No file

Time window

		Width	Shift	Start	
☐ Automatic	First	10	1	0	
[✓] Overlap	Last	10	1	0	Lag limit
☐ Onsets only	Step	1	1	1	-1

Projection type: [✓ Gaussian] Sparse
Vectors: [✓] Read [✓] Save; Dimension 10
Projection function: [✓ Linear] Exponential Sequential order Event occurrence
Excluded windows: [✓] Exclude identical [✓] Exclude overlapped [✓] Exclude symmetrical [✓] Exclude disjoint [✓] Exclude truncated
Compare: All Self [✓ Cross] Pair
Results: [✓] Similarity lattices 0 %; Filter 1; [✓] Highly similar windows 95 %; ☐ Window pairs 50 %; ☐ Echo seqs ☐ Save numerical
Output files: [✓] Display HTML output [✓] Overwrite output

Figure 8. User interface of the RAP program.

AUTHOR NOTE

This research project was supported by Grant 2005SGR00098 from the Directorate General for Research of the Government of Catalonia, Spain. Correspondence concerning this article should be addressed to V. Quera, Departamento de Metodología de las Ciencias del Comportamiento, Universidad de Barcelona, Campus Mundet, Paseo Valle de Hebrón, 171, E-08035 Barcelona, Spain (e-mail: vquera@ub.edu).

REFERENCES

Abbott, A., & Barman, E. (1997). Sequence comparison via alignment and Gibbs sampling: A formal analysis of the emergence of the modern sociological article. *Sociological Methodology*, **27**, 47-87.

Achlioptas, D. (2003). Database-friendly random projections. *Journal of Computer & System Sciences*, **66**, 671-687.

Bakeman, R. (1978). Untangling streams of behavior: Sequential analysis of observation data. In G. P. Sackett (Ed.), *Observing behavior: Data collection and analysis methods* (Vol. 2, pp. 63-78). Baltimore: University Park Press.

Bakeman, R. (2000). Behavioral observation and coding. In H. T. Reis & C. M. Judd (Eds.), *Handbook of research methods in social and personality psychology* (pp. 138-159). Cambridge: Cambridge University Press.

Bakeman, R., Adamson, L. B., & Strisik, P. (1995). Lags and logs: Statistical approaches to interaction (SPSS version). In J. M. Gottman (Ed.), *The analysis of change* (pp. 279-308). Mahwah, NJ: Erlbaum.

Bakeman, R., Deckner, D. F., & Quera, V. (2005). Analysis of behavioral streams. In D. M. Teti (Ed.), *Handbook of research methods in developmental science* (pp. 394-420). Oxford: Blackwell.

Bakeman, R., & Quera, V. (1992). SDIS: A sequential data interchange standard. *Behavior Research Methods, Instruments, & Computers*, **24**, 554-559.

Bakeman, R., & Quera, V. (1995a). *Analyzing interaction: Sequential analysis with SDIS and GSEQ*. Cambridge: Cambridge University Press.

Bakeman, R., & Quera, V. (1995b). Log-linear approaches to lag-sequential analysis when consecutive codes may and cannot repeat. *Psychological Bulletin*, **118**, 272-284.

Bingham, E., & Mannila, H. (2001). Random projection in dimensionality reduction: Applications to image and text data. In F. Provost & R. Srikant (Eds.), *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2001* (pp. 245-250). New York: ACM Press.

Blum, A. (2006). Random projection, margins, kernels, and feature selection. In C. Saunders, M. Grobelnik, J. Stefan, S. Gunn, & J. Shawe-Taylor (Eds.), *Subspace, latent structure and feature selection: Statistical and Optimization Perspectives Workshop, SLSFS 2005* (pp. 52-68). Berlin: Springer.

Durbin, R., Eddy, S., Krogh, A., & Mitchison, G. (1998). *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. Cambridge: Cambridge University Press.

FU, W.-T. (2001). ACT-PRO action protocol analyzer: A tool for analyzing discrete action protocols. *Behavior Research Methods, Instruments, & Computers*, **33**, 149-158.

GARDNER, W., & HARTMANN, D. P. (1984). On Markov dependence in the analysis of social interaction. *Behavioral Assessment*, **6**, 229-236.

GRIFFIN, W. A., & GARDNER, W. (1989). Analysis of behavioral durations in observational studies of social interaction. *Psychological Bulletin*, **106**, 497-502.

KANG, J. (2005). *Data models for exploratory analysis of heterogeneous microarray data* (Tech. Rep. TR-2005-30). Raleigh: North Carolina State University.

KURTZ, S., CHOUDHURI, J. V., OHLEBUSCH, E., SCHLEIERMACHER, C., STOYE, J., & GIEGERICH, R. (2001). REPuter: The manifold applications of repeat analysis on a genomic scale. *Nucleic Acids Research*, **29**, 4633-4642.

MAGNUSSON, M. S. (2000). Discovering hidden time patterns in behavior: T-patterns and their detection. *Behavior Research Methods, Instruments, & Computers*, **32**, 93-110.

MANNILA, H., & SEPPÄNEN, J. (2001). Recognizing similar situations from event sequences. In *Proceedings of the First SIAM Conference on Data Mining*, Chicago. Available at www.cs.helsinki.fi/~mannila/postscripts/mannilaseppanensiam.pdf.

MANNILA, H., & TOIVONEN, H. (1996). Discovering generalized episodes using minimal occurrences. In E. Simoudis, J. Han, & U. M. Fayyad (Eds.), *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (pp. 146-151). Portland, OR: AAAI Press.

MAS, M. T. (2003). *L'atenció conjunta dels 10 als 28 mesos d'edat de l'infant* [Joint attention in 10- to 28-month-old children]. Unpublished doctoral thesis. Facultat de Psicologia, Universitat Autònoma de Barcelona.

MOEN, P. (2000). *Attribute, event sequence, and event type similarity notions for data mining* (Rep. A-2000-1). Helsinki: University of Helsinki, Department of Computer Science.

QUERA, V., & BAKEMAN, R. (2000). Quantification strategies in behavioral observation research. In T. Thompson, D. Felce, & F. Symons (Eds.), *Behavioral observation: Technology and applications in developmental disabilities* (pp. 297-315). Baltimore: Brookes.

QUERA, V., BAKEMAN, R., & GNISCI, A. (2007). Observer agreement for event sequences: Methods and software for sequence alignment and reliability estimates. *Behavior Research Methods*, **39**, 39-49.

SACKETT, G. P. (1979). The lag sequential analysis of contingency and cyclicity in behavioral interaction research. In J. D. Osofsky (Ed.), *Handbook of infant development* (1st ed., pp. 623-649). New York: Wiley.

SAHLGREN, M. (2005). An introduction to random indexing. In H. Witschel (Ed.), *Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering*, August 16, 2005, Copenhagen. Available at www.sics.se/~mange/papers/RI_intro.pdf.

STATES, D. J., & BOGUSKI, M. S. (1991). Similarity and homology. In M. Gribskov & J. Devereux (Eds.), *Sequence analysis primer* (pp. 89-157). New York: Stockton.

STOOLMILLER, M., & SNYDER, J. (2006). Modeling heterogeneity in social interaction processes using multilevel survival analysis. *Psychological Methods*, **11**, 164-177.

TAYLOR, P. J. (2006). Proximity coefficients as a measure of interrelationships in sequences of behavior. *Behavior Research Methods*, **38**, 42-50.

VEMPALA, S. (2004). *The random projection method*. Providence, RI: American Mathematical Society.

(Manuscript received February 13, 2007;
accepted for publication March 13, 2007.)