JOURNAL OF LEARNING ANALYTICS

SOLAR SOCIETY for LEARNING ANALYTICS RESEARCH

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

Microgenetic Learning Analytics Methods: Workshop Report

Ani Aghababyan
Digital Platform Group, McGraw-Hill Education, USA
anie.aghababyan@gmail.com

Taylor Martin
Principal Learning Scientist at O'Reilly Media, USA

Philip Janisiewicz
Agile Dynamics, USA

Kevin Close
Instructional Technology and Learning Sciences, Utah State University, USA

ABSTRACT: Learning analytics is an emerging discipline and, as such, benefits from new tools and methodological approaches. This work reviews and summarizes our workshop on microgenetic data analysis techniques using R, held at the second annual Learning Analytics Summer Institute in Cambridge, Massachusetts, on 30 June 2014. Specifically, this paper introduces educational researchers to our experience using data analysis techniques with the RStudio development environment to analyze temporal records of 52 elementary students' affective and behavioural responses to a digital learning environment. In the RStudio development environment, we used methods such as hierarchical clustering and sequential pattern mining. We also used RStudio to create effective data visualizations of our complex data. The scope of the workshop, and this paper, assumes little prior knowledge of the R programming language, and thus, covers everything from data import and cleanup to advanced microgenetic analysis techniques. Additionally, readers will be introduced to software setup, R data types, and visualizations. This paper not only adds to the toolbox for learning analytics researchers (particularly when analyzing time series data), but also shares our experience interpreting a unique and complex dataset.

Keywords: Microgenetic analysis, R, RStudio, learning analytics, data mining, hierarchical clustering, sequential pattern mining

1 BACKGROUND

Microgenetic analysis is a technique that enables the discovery of learning progression components and their influence on learning outcomes (Kuhn, 1995; Martin et al. 2013; Stigler, Givvin, & Thompson, 2010). Microgenetic research investigates processes of learning (Kuhn, 1995; Siegler & Crowley, 1991). This research method reveals how processes of activity change over time, how differential processes and patterns change across individuals, and how processes are related to critical outcomes, such as success, growth, or engagement. Three main elements distinguish microgenetic research designs: 1) the

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

time span of the research occurs when a competency is likely to develop or be learned, 2) observations of learning behaviour are as dense as possible within this window, and 3) analysis of learning behaviour is conducted on an instance-by-instance basis (Fazio & Siegler, 2013). For example, in microgenetic studies of children learning addition, researchers met with participants every week for 11 weeks and had them solve simple addition problems (e.g., 2+4) in interviews (Siegler & Jenkins, 1989). They recorded both accuracy and learner strategy. This research demonstrated how less efficient strategies transitioned to more efficient strategies over time. In addition, it showed that children did not maintain successful new strategies immediately. Instead, a new successful strategy appeared at first sporadically, then gradually became the predominant strategy (though less efficient or incorrect strategies still appeared). This method has recently been employed in understanding self-explanation (Cheshire, Ball, & Lewis, 2005), the interaction between speech and gesture (Pine, Lufkin, Kirk, & Messer, 2007), and representations of numerical magnitude (Opfer & Thompson, 2008). These pioneering works seek to explain the mechanisms of learning hidden in the black box of activity between pre-test and post-test.

To date, the grain size for these studies has been fairly large (e.g., at the level of a problem on an assessment) and the number of time points has been relatively small (e.g., ten interviews over the course of ten weeks). These elements can be greatly improved using Learning Analytics (LA) and Educational Data Mining (EDM) methods. Specifically, researchers can increase the *density of observations* and the *number of instances for analysis* — two of Fazio and Siegler's (2013) criteria for microgenetic designs for research. Some researchers have begun expanding microgenetic methodology using LA and EDM methods (e.g., Baker, Hershkovitz, Rossi, Goldstein, & Gowda, 2013; Blikstein et al., 2014; Martin et al., 2014), but their work is still in early stages. Their methods take advantage of techniques such as cluster analysis, classification, and sequence and pattern mining to investigate log data from digital learning environments, corpora of texts from students' writing, transcriptions of video or audio-recorded data, and data manually and automatically collected from face-to-face learning environments. For example, with the moment-by-moment learning curve, Baker et al. (2013) demonstrated the effectiveness of techniques for determining the probability that a student had learned a given concept at a given moment of activity in a genetics intelligent tutor program. In addition, many EDM researchers use methods and conduct analyses that could inform microgenetic research (even if they do not place their work within the microgenetic paradigm). Many of these approaches come under the umbrella of process and sequence mining.

Process Mining techniques in EDM lead to descriptions of overall processes such as the process of learning, of change in systems, or of resources and hints use (Trčka, Pechenizkiy, & van der Aalst, 2010). However, process mining is a broad term (and set of methodologies). For example, one can identify overall characteristics of a process without paying explicit attention to sequences. For example, Berland, Martin, Benton, Petrick Smith, & Davis (2013) found the probability of programmers transitioning from one type of program to another over a single programming session (types discovered using cluster analysis). We suggest that, as researchers develop a microgenetic research line within EDM, sequences should be a basic characteristic of microgenetic research design.

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

Sequential pattern mining is an example of a technique that focuses on how events are ordered and how different orderings can be consequential (Agrawal & Srikant, 1995; Zhou, Xu, Nesbit, & Winne, 2010). For example, Perera, Kay, Koprinska, Yacef, and Zaïane (2009) used sequential pattern mining to improve their original predictions based on cluster analysis that did not consider time or order in a study on group work. They found that different patterns of interactivity for groups related to successful performance and less successful performance. Additionally, Kinnebrew and Biswas (2012) employed differential sequence mining techniques to understand patterns of reading behaviours and how these patterns differentiate between more and less successful students.

In this workshop, we presented how data collected with a microgenetic analysis approach can be analyzed using several different learning analytics methods. All the data wrangling, clean up, and data analyses included in this report were conducted using R programming language.

2 WORKSHOP

2.1 Software

For this workshop, attendees were asked to download and install the latest versions of R[1] and RStudio[2] software. Both software are compatible with a wide range of operating systems such as Windows, Mac OS X, Fedora/RedHat, and others. After acquiring the required technological components, attendees were presented with the content of our workshop as follows:

- Introduction to R (Section 3)
- Main analysis in R (Section 4)

2.2 Data Description

Data selected for this workshop was composed of human-coded field observations of a group of elementary school students engaged in a science-teaching educational digital game while being observed for their emotional response to the game. This data was collected as part of Ani Aghababyan's (2014) doctoral thesis under the supervision of Dr. Taylor Martin.

The dataset contained temporal records of students' affective and behavioural responses to the learning environment. Table 1 presents a fraction of this dataset. In this table, we have two rows of information about the same user. Row one describes the student's first observed behaviour as on task and his/her first observed emotion as concentrating. These observations are organized in temporal order (column 5, observation rank). The same student's second affect and behaviour observation is slightly different, showing that his/her emotion changed from concentration to confusion. This dataset contains observations for 52 students over a period of nine days with nine consecutive observations per student

[1] http://www.r-project.org/
[2] http://www.rstudio.com/products/rstudio/download/

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

per each day that they were observed (only about 15 students were observed per day due to limited time).

Table 1. Dataset details

Username	Timestamp	Behaviour	Affect	Observation Rank
Day1-1	12886	On Task	Concentrating	1
Day1-1	34323	On Task	Confused	2

3 INTRODUCTION TO R

In this section of the workshop, we presented introductory topics for first-time R users. As part of this section, we explained and demonstrated data-import procedures, data-cleanup methods, differences in data types, data aggregations, and data-visualization methods. Below we present snippets of the code used in the workshop for each of these subsections. Most of the code can be rewritten in alternate ways; hence, if you are an experiences R user, feel free to alter the code to your style. Otherwise, follow the steps to replicate the workshop.

3.1 Environment Setup

Before using the programming code for these analyses, we recommended that all attendees download necessary R packages (otherwise called libraries). This list of libraries contains packages such as TraMineR (sequence mining and graphical tool; Gabadinho, Ritschard, Müller, & Studer, 2011), cluster (built-in cluster analysis methods; Maechler, Rousseeuw, Struyf, Hubert, & Hornik, 2015), arulesSequences (mining tool for frequent sequences; Buchta, Hahsler, Buchta, & Matrix, 2007), and the necessary graphics tools for these temporal analyses (see the code block below). Below, each line of functional code follows a line of instructional comment or explanation of what the code does: green lines are instructional comments or explanations in R (these follow the hashtag "#" sign) while blue lines are functional code blocks (these follow the ">" sign). Finally, we chose to use the "=" sign as the assignment operator; however, you are welcome to use the original "<–" R assignment operator.

Importing R libraries (green = R comments, blue = R functional code):

```
# Download necessary libraries
# TraMineR is an R-package for mining sequential data
> if(!require(TraMineR)){install.packages(TraMineR)}
# Built-in Cluster Analysis Methods
> if(!require(cluster)){install.packages(cluster)}
# Graphics package for different visualizations
> if(!require(graphics)){install.packages(graphics)}
```

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

```
# An alternative to Data Frame
> if(!require(data.table)){install.packages(data.table)}
# Frequent sequence mining package that uses CSPADE algorithm
> if(!require(arulesSequences)){install.packages(arulesSequences)}
# Package for data manipulation and reshape
> if(!require(reshape)){install.packages(reshape)}
# Package for splitting, applying, and combining data
> if(!require(plyr)){install.packages(plyr)}
```

The above code both installs and calls all the required libraries. However, for users who had previously downloaded these libraries, we recommended replacing the code above with the following code:

```
# Call already downloaded libraries (these three packages only serve as examples).
> library(cluster)
> library(data.table)
> library(plyr)
etc.
```

3.2 Data types

The first step in our data analysis was to introduce our audience to the basic data types built into R programming language. Some of these data types are as follows: *numeric*, *character*, *factor*, and *logical*. In addition, we introduced the concept of data structures such as *vectors* and *data frames*, which are the initial steps for every data analysis. As an additional resource, we directed all attendees to an online R-tutorial at http://www.r-tutor.com/r-introduction/.

Numeric data types are variables that contain numeric values. See the code below:

```
# Variable “x” equals 1
> x = 1
# Variable “x” equals 0.25
> x = .25
# Variable “x” equals the sum of 1 and 0.25
> x = 1+0.25
# Variable “x” equals the quotient of 1 divided by 3
> x = 1/3
```

Characters, otherwise called strings, are any characters surrounded by either single or double quotation marks. See the code below:

```
# Variable “x” equals a string composed of characters that reads “hello”
> x = "hello"
# Variable “x” equals the combination of strings “hello” and “LASI”: “hello LASI”
```

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

```
> x = paste("hello","lasi")
```

Logical data types are variables with either True or False values. See the code below:

```
# Create variable “a” that equals to Boolean True
> a = TRUE
# Create variable “b” that equals to Boolean False
> b = FALSE
# According to Boolean logic “a and b” equals False
> a && b
# According to Boolean logic “a or b” equals True
> a || b
```

Vectors are one-dimensional arrays. See the code below:

```
# Create vector “v” with elements 1, 2, 3, and 4
> v = c(1,2,3,4)
# Create vector “w” with elements 2, 4, 6, and 8
> w = c(2,4,6,8)
# Add the elements of vector “v” to the elements of vector “w,” which will result in a vector with elements 3, 6, 9, and 12
> v + w
# Multiply the elements of vector “v” with the elements of vector “w,” which will result in a vector with elements 2, 8, 18, and 32
> v*w
```

Factors are categorized variables, otherwise called nominal variables. See the code below:

```
# Create a gender variable where there are two factors: male and female. Here, c() is the function for concatenation and rep() is the function for replication.
# Create a character variable
> gender = c(rep("male",5), rep("female", 4))
# Convert the character variable into factor
> df_ft = factor(gender)
```

Data Frames are lists of vectors of equal length. They have columns and rows, which can contain different data types (i.e., numeric, string, or Boolean data types). See the code below:

```
# Create a data frame composed of a) numerical data, b) string data, and c) Boolean data
a.  > d = c(1,2,3,4)
b.  > e = c("red", "white", "red", NA)
c.  > f = c(TRUE,TRUE,TRUE,FALSE)
> mydata = data.frame(d,e,f)
```

The following line of code accesses a specific value located in row 1, column 2 from a data frame:

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

```
# access the value within the cell on row 1 and column 2
> mtcars[1, 2]
```

3.3 Visualizations

One of the biggest advantages of R programing language is its graphical capabilities. R visualizations easily allow the user to create any statistical figure with elaborate customizations. Hence, for this section, we introduced a set of graphical functions in R.

Scatterplot matrices, as opposed to simple scatterplots, compare several factors. To demonstrate an example of scatterplot matrices, we used "mtcars" data set, which is an internal data set available within R. It is a data frame composed of 32 observations (i.e., rows) on 11 variables (i.e., columns). Variables contain information about the average miles per gallon, number of cylinders, weight, horsepower, and other measures of several cars. The scatterplot matrices display all 11 variables compared (see Figure 1).

```
# Scatterplot matrices
> pairs(mtcars, main = "mtcars data")
```

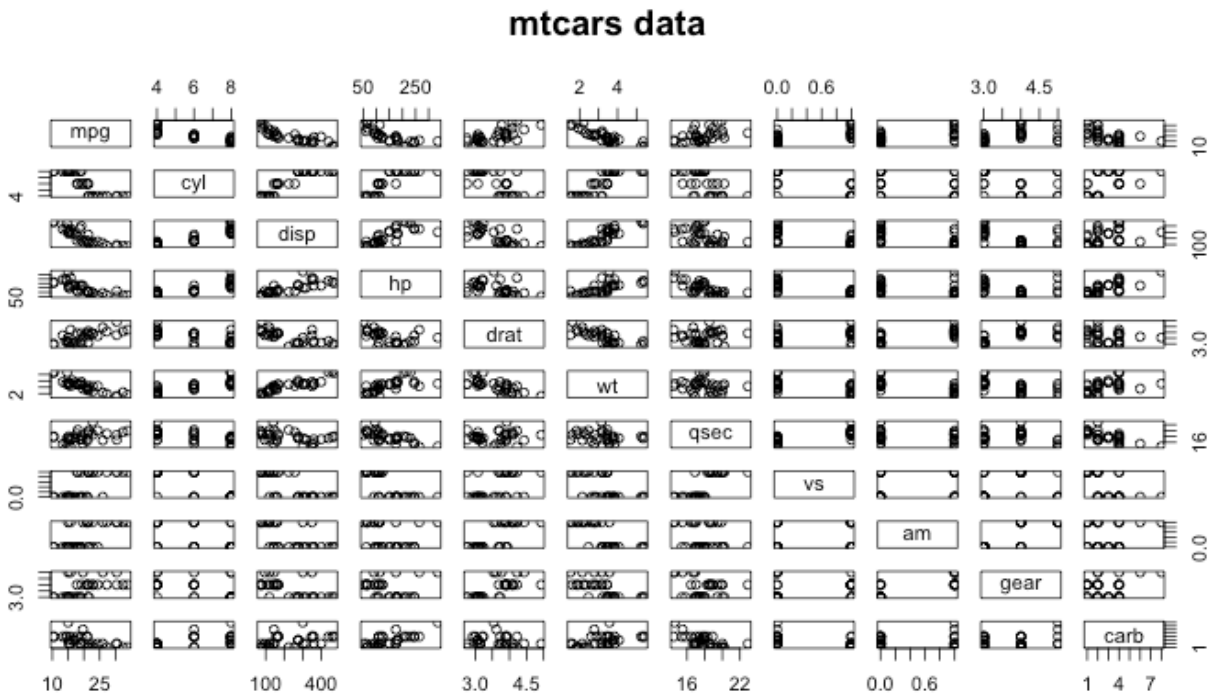

Figure 1. Scatterplot matrices

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

Conditional plots specify a key factor (a variable) in the dataset for the comparison of other variables. In the workshop (see the code below), we used the number of cylinders as a factor for the comparison of miles/gallon and displacement (i.e., engine volume).

```
# Conditional plots
> coplot(mpg ~ disp | as.factor(cyl), data = mtcars, panel = panel.smooth, rows = 1)
```

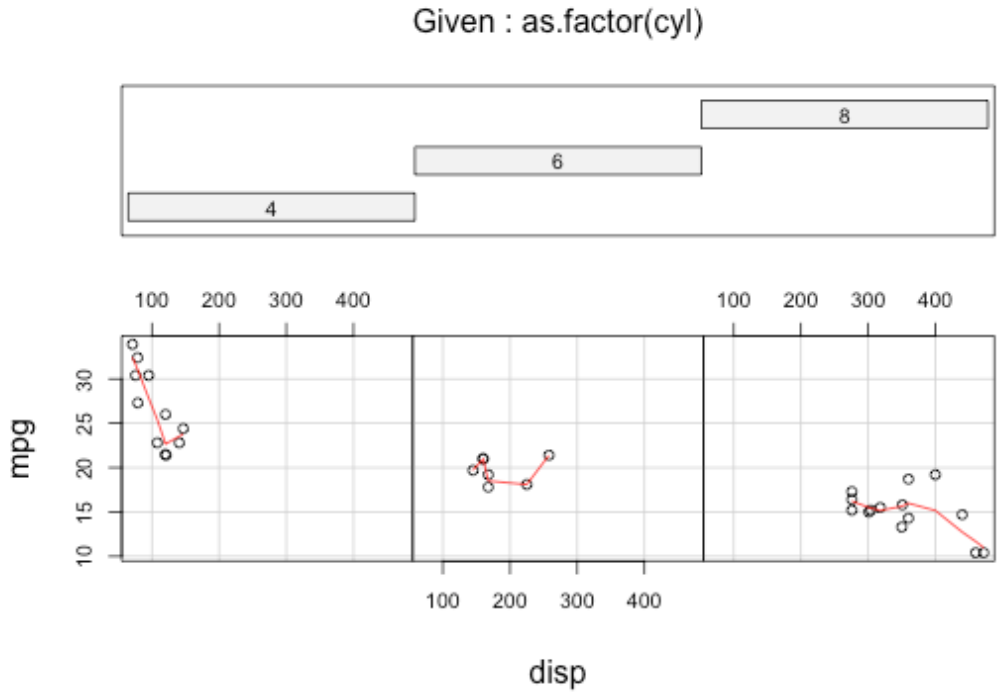

Figure 2. Conditional plots

4 MAIN ANALYSES IN R: MICROGENETIC METHODS FOR LEARNING ANALYTICS

In this section, we present our sequence analysis in detail.

4.1 Data Import

The first step in our data analysis was to explain the process of importing the data file. First, we explained how to change the working directory to the workshop folder (see the code below):

```
# Change the working directory (given the workshop folder is within the "lasi" folder in the home directory)
> setwd("~/lasi/workshop")
# Import csv file
> behavior_affect_data = read.csv("/behavior_affect_data.csv")
```

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

```
# Review first few rows of the data file
> head(behavior_affect_data)
```

Table 2, below, presents the first 10 rows of the data file, which we used for the rest of our workshop. This dataset contains student behaviour and affect data collected over a period of nine days. Each student's observation data (i.e., behaviour and affect) for each day was appended to a separate array with the date and username information. For example, Table 2, Column 1 shows nine rows of data for the first student recorded during day 1 of observations (Day1-1). The next row, row 10, displays data for the second student recorded during day 1 of observations (Day1-2). Each day, each of the observed student's affect and behaviour was recorded nine times in a row, thus creating an array of nine paired elements (nine pairs of affect and behaviour). These arrays were broken down into one row per observation and ranked according to their timestamp (see Table 2, Columns 2 and 5). The timestamp allowed us to reorder all the data according to its temporal attribute. Column 5 (observation rank) is generated based on the timestamp, where rank = 1 identified the first element in the array and rank = 9 identified the last element in the same array. Columns 3 and 4 represent the behaviour and affect states, respectively. In this data, there were six behaviour categories: 1) on task (OT), 2) other on conversation (OOC), 3) off task (OfT), 4) giving help (GH), 5) receiving help (RH), and 6) unknown behaviour (?). There were also eight affect categories: 1) bored (B), 2) confused (CF), 3) concentrating (C), 4) frustrated (F), 5) delighted (D), 6) surprised (S), 7) experiencing a eureka moment (E), and 8) unknown affect (?).

Table 2. First 10 rows of the behaviour–affect data file

Username	Timestamp	Behaviour	Affect	Observation Rank
Day1-1	12886	OT	C	1
Day1-1	34323	OT	C	2
Day1-1	98789	OT	C	3
Day1-1	161540	OfT	CF	4
Day1-1	199791	OT	CF	5
Day1-1	239859	OT	CF	6
Day1-1	274391	OT	CF	7
Day1-1	310893	OOC	CF	8
Day1-1	331876	OT	CF	9
Day1-2	18852	OT	C	1

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

In general, for effective time series or sequence analyses, we recommend using datasets that contain order information. This dataset had a time variable and a rank variable, which both organize the events into chronological order by user.

Before starting any analysis, it is good practice to become familiar with the data set. Therefore, we presented a few simple approaches for collecting summary information. See the code below:

```
# Output the first 10 rows of the data
> head(behavior_affect_data, n = 10)
# Count the number of unique usernames in the data file
> length(unique(behavior_affect_data$username))
```

4.2 Data cleanup and/or manipulations

Real data, more often than not, requires extensive cleanup and, possibly, feature engineering or data wrangling. Depending on the analysis, the required data manipulation procedure may vary. For our sequence analysis, we concatenated behaviour and affect records with the following code.

```
# Concatenate behaviour and affect variables and create a new column with this new information
> behavior_affect_data$a_b = do.call(paste, c(behavior_affect_data[c("affect", "behavior")], sep = "_"))
```

4.3 Reshape for Sequence Analysis

To begin the sequence analysis, we showed our audience how to create sequences using the R *reshape* function. Guided by domain knowledge and the fact that there is a gap between each data collection session larger than 24 hours, we decided to treat each day's recordings as separate sequences and treat each student as a different student when recorded again on another day.

```
# Wide reshape of the data from a sequence of time points to a set of sequences shaped by observation
rank and grouped by username
> behavior_affect_sequence = reshape(behavior_affect_data[,c("username", "obs_rank", "a_b")],
        timevar = "obs_rank",
        idvar = c("username"),
        direction = "wide")
```

According to the reshape function, the user must first pass the data set that they are trying to convert into sequences. The "timevar" argument is then used to select the order variable, which will help reshape the data without altering the original chronology of events. The "idvar" argument then identifies the variables the user wants to group, in this case the username. Finally, users must also specify the direction of the output file (i.e., whether the data will be "wide" and have one user ID and set of observations per line or "long" with multiple rows per subject).

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

Prior to this data reshape process, we had 2480 single observations of affect–behaviour combination (e.g., user1: C-OT; user1: CF-OT; user2: F-OfT, etc.). After this aggregation, we created 273 sequences (e.g., user1: C-OT, C-OT, C-OT, CF-OfT, CF-OT, CF-OT, CF-OT, CF-OOC, CF-OT; user2: C-OT, CF-OT, CF-OOC, C-OT, C-OT, C-OT, CF-OT, C-RH, C-OT). This transformation allowed us to reformat our sequence data into actual sequences.

4.4 Creating States

In this section of our analyses, we introduced our audience to R's TraMineR package. This package provides a comprehensive toolbox for mining categorical sequence data, specifically formulated to account for the unique features of social science data. TraMineR provides tools for data preparation, rendering of sequences, computing the distance between the sequences using several different algorithms, conducting hierarchical clustering on dissimilarity data, and, finally, visualizing these sequences graphically.

In our workshop, we used the "seqstatl" function, which creates a list of distinct states/events found in a data frame (otherwise called the "alphabet").

```
# Returns list of distinct states/events (alphabet) in sequence data
> seqstatl(behavior_affect_sequence[, 2:length(behavior_affect_sequence)])
```

To explore the "seqstatl" function further, we displayed "data.frame.alphabet," "data.frame.labels," and "data.frame.scode," which are the values, the labels, and the short descriptions, respectively, for each state. Below are the code blocks for each of these state characteristics. It is important to note that in our workshop, we did not use all of the core elements of the TraMineR package such as the labels list for assigning long labels or the scodes for assigning short labels to the states defined within the alphabet attribute. However, we presented them below in the code, so you can borrow the benefits of these attributes (see each explained below). For more information on how to use these attributes, see the package documentation below.
https://cran.r-project.org/web/packages/TraMineR/vignettes/TraMineR-state-sequence.pdf

```
# These are the values within the data frame sequence columns.
> behavior_affect_sequence.alphabet =
seqstatl(behavior_affect_sequence[, 2:length(behavior_affect_sequence)])

# These are the long labels for each state that could be used for colour legends.
> behavior_affect_sequence.labels = behavior_affect_sequence.alphabet

# These are the short codes or descriptors for each state. These may be a few characters in length to keep the output cleaner when printing.
> behavior_affect_sequence.scodes = behavior_affect_sequence.alphabet
```

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

Once the number of unique states was identified, we assigned colours to each of these states. The selection of colours was arbitrary. However, we grouped the similar states into a similar colour scheme, thus ensuring that all the states with concentration are in one colour group (shades of blue), while all the states with confusion are in another (shades of red). Users must have a sufficient set of colours to identify and separate each state (RColorBrewer may be a useful resource; Neuwirth & Neuwirth, 2007). We selected 28 distinct colours for our 28 states.

```
# Define the colours for each state
> colors = c(
 "gray55", "gray73",
 "lightpink4", "lightpink3", "lightpink2", "lightpink",
 "dodgerblue4", "dodgerblue3", "deepskyblue3", "cadetblue3", "cadetblue1",
 "brown4", "brown3", "firebrick1", "coral1", "lightcoral",
 "yellow4", "yellow3", "yellow1", "khaki1",
 "seagreen4", "seagreen1",
 "violetred4", "violetred", "violetred1", "orchid1",
 "navajowhite3", "moccasin")
```

Next, we created the sequences with the scode, label, alphabet, and colour parameters that were set in the previous few code blocks.

```
# Create sequences with given parameters
> behavior_affect_traminer_sequence = seqdef(behavior_affect_sequence,
        2:length(behavior_affect_sequence),
        alphabet = behavior_affect_sequence.alphabet,
        states = behavior_affect_sequence.scodes,
        labels = behavior_affect_sequence.labels,
        xtstep = 1,
        cpal=colors)
```

4.5 Visualizing Sequences

This section of sequence analysis allows the researcher to see the overall patterns in the data. In the next few blocks of R code, we presented some useful TraMineR graphics.

```
# Set to view the graphics in a 1 row and 1 column format (optional step)
> par(mfrow = c(1, 1), cex.main=0.7)

# Plot the legend for state colours
> seqlegend(behavior_affect_traminer_sequence, fontsize = 0.8)
```

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

```
# Plot all sequences
> seqIplot(behavior_affect_traminer_sequence, sortv = "from.start", withlegend = FALSE, ylab=NULL, xlab="Observation")
> seqIplot(behavior_affect_traminer_sequence, sortv = "from.end", withlegend = FALSE, ylab=NULL, xlab="Observation")

# Plot the 10 most frequent sequences
> seqfplot(behavior_affect_traminer_sequence, withlegend = FALSE, border=NA, title="10 Most Frequent Affect_Behavior Sequences", xlab="Observation")

# Plot the transversal entropy index (see TraMineR package for more details).
> seqHtplot(behavior_affect_traminer_sequence)

# Plot the mean time spent in each state of the alphabet.
> seqmtplot(behavior_affect_traminer_sequence, withlegend = FALSE)
```

4.6 R Sequence Mining Algorithm (aRulesSequences)

The Sequential PAttern Discovery using Equivalence classes (SPADE) algorithm allows us to mine for frequent sequential patterns using temporal joins (Zaki, 2001). This analysis allows the researcher to discover inter-event patterns (i.e., sequences) within many different input-sequences. To accomplish this analysis, we used the arulesSequences package. Using this package, initially we created a basket format with one row per each state as a sequence of 1. Afterword, the state ID and size information was used to create a temporal object, which served as an input in CSPADE function. The algorithm found subsequences from initial temporal object sequences and provided frequencies for each of the subsequences. This resulted in new sequence data, which can subsequently be used to conduct inter-sequence distance analysis and hierarchical clustering. However, for the purposes of this workshop and the ease of implementation, in section 4.7, we chose to continue using the "behaviour affect traminer sequence" data file previously used to visualize sequences in section 4.5.

```
# Create a size of 1
> behavior_affect_data$size = 1

# Store the data
> write.table(behavior_affect_data[,c("username","obs_rank","size","a_b")],
file="~/observation_table.csv", row.names = FALSE, col.names = FALSE, sep = ",")

# Create a temporal object
> trans = read_baskets("~/observation_table.csv", sep = ",", info = c("sequenceID", "eventID", "SIZE"))

# CSPADE analysis
> cs = cspade(trans, parameter = list(support = 0, maxgap = 1), control = list(verbose = TRUE))
```

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

```
> observed_sequences = as(cs, "data.frame")
> observed_sequences = observed_sequences[c("support", "sequence")]
> observed_sequences = observed_sequences[ order(observed_sequences[, "support"], decreasing =
TRUE), ]
> row.names(observed_sequences) = NULL

# Display the summary of CSPADE results
> summary(cs)

# Store the data
write.table(cs, "behavior_affect_traminer_sequence.csv", row.names = FALSE, col.names = TRUE, sep =
",")
```

4.7 Optimal Matching and Clustering

In this section, we computed the distances between our sequences using optimal-matching (Abbott & Tsay, 2000). To do so, we decided on a substitution–cost matrix using the "seqsubm" function. This function creates the substitution matrix using either a constant or the transition rates. In our case, we selected a constant substitution cost of 1. This meant that, in order to assimilate two sequences, we could either delete or substitute states. Each change would cost 1. Following the optimal matching calculation, we decided on a clustering algorithm. Given our data, we decided to use Ward's (1963) hierarchical clustering method. In this example, we selected a 3-cluster solution. However, workshop participants could alter that selection by changing the value of "k." Once we chose the cluster solution, we converted that number into a factor and plotted the sequences. It is important to highlight that the literature in the area does not provide a distinctive guideline as to what is the appropriate cost (Wu, 2000); however, to learn more about some of the limitations on cost choice, see Chapter 5 of Aghababyan (2014) regarding the transformation cost.

```
# Create constant substitution matrix cost of 1
> substitution_matix =
seqsubm(behavior_affect_traminer_sequence, method="CONSTANT", cval=1)

# Create the distance matrix based on the substitution cost with insert/deletion cost 1
> dist.om.const =
seqdist(behavior_affect_traminer_sequence, method="OM",indel=1,sm=substitution_matix)

# Cluster the distance matrix using Ward's method
> clusterward = agnes(dist.om.const, diss = TRUE, method = "ward")

# Generate a dendrogram
> plot(clusterward, which.plot = 2)

# Select the optimal number of clusters and store the solution
```

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

```
> wards3 = cutree(clusterward, k = 3)

# View cluster memberships. Use these numbers for the y-axis in the seqIplot below.
> table(wards3)

# Convert the cluster numbers into factors
> wards3fac = factor(wards3, labels = paste("Cluster", 1:3))

# Add the clusters to the data frame
> behavior_affect_sequence$wards3 = wards3

# Plot all sequences from the 3-cluster solution
> seqIplot(behavior_affect_traminer_sequence,
        cex.plot = .8,
        group = wards3fac,
        sortv = "from.end",
        las = 2,
        ylim = c(0,30)
        )
```

4.8 R Cluster Visualizations

As the final step to our sequence analysis, we visualized the results from our cluster analysis. The code below is based on Ward's 3-cluster solution. However, users could change the cluster visualization by choosing another solution based on need.

```
# Create a function for sequence visualization
> visualize_sequence <– function(sequence) {
        par(mfrow = c(2, 2))
        seqdplot(sequence, withlegend = FALSE, border = NA)
        seqHtplot(sequence)
        seqmsplot(sequence, withlegend = FALSE, border = NA)
        seqmtplot(sequence, withlegend = FALSE, ylim=c(0,7))
        }

# Create visualizations for clustered sequences
> visualize_sequence(behavior_affect_traminer_sequence[which(behavior_affect_sequence$wards3 ==
1),])
> visualize_sequence(behavior_affect_traminer_sequence[which(behavior_affect_sequence$wards3 ==
2),])
> visualize_sequence(behavior_affect_traminer_sequence[which(behavior_affect_sequence$wards3 ==
3),])
```

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

4.9 Hands on Experience

Finally, we encouraged each participant either to analyze their own temporal datasets or to select one of the additional datasets provided at the workshop. It is important to highlight that, while the code provided could easily adapt to most sequential data files (which we were able to demonstrate with a secondary, unrelated data set), each dataset will require unique domain-specific alterations. Section 5 expands more on the results of this workshop, its contributions, and the importance of this methodology for the Data Mining and Learning Analytics fields.

5 DISCUSSION & CONCLUSIONS

In the field of Educational Data Mining, it has been noted that the issue of time and sequence analysis within context plays an important role (Baker, 2010). Time series analyses have been extensively studied in statistics, especially for applications in biology (Low-Kam, Raïssi, Kaytoue, & Pei, 2013). However, the transition to the field of education has been difficult due to the complexities of student data, especially game datasets (Pahl & Donnellan, 2003; Sanjeev & Zytkow, 1995; Shen, Yang, & Han, 2003; Zaïane & Luo, 2001). There are different approaches to sequential pattern mining. In this workshop, we explored an itemset approach, where we tried to find statistically relevant patterns in our data considering each item as a state in the order they appeared. The implications of such analysis depend on its application. In our example, our goal was to observe affect patterns among students to compare with theoretical assumptions about affective state transitions. Our aim was to observe the transition from the state of frustration to the next affective state. Understanding sequence patterns in affect could help determine when games should adapt to learners' affective states. However, as with any other method, sequence mining also has its limitations. The major limitation for our approach was the time gap between each observation (Aghababyan, 2014).

This workshop covered not only technical nuances of analyses in R but also discussed educational data intricacies. The workshop contained both lecture and hands-on elements, which facilitated the tutorial session and ensured engagement despite varying levels of expertise. We believe that this workshop session successfully introduced researchers to several data-mining approaches and introduced them to R.

6 ACKNOWLEDGMENTS

This material is based upon work supported by the Bill and Melinda Gates Foundation and the National Science Foundation Grants EEC 1329438 and DRL 1338176 (while serving at) the National Science Foundation. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Bill and Melinda Gates Foundation or the National Science Foundation.

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

REFERENCES

Abbott, A., & Tsay, A. (2000). Sequence analysis and optimal matching methods in sociology. *Review and Prospect Sociological Methods & Research*, *29,* 3–33. http://dx.doi.org/10.1177/0049124100029001001

Agrawal, R., & Srikant, R. (1995). Mining sequential patterns. *Proceedings of the 11th International Conference on Data Engineering*, 6–10 March 1995, Taipei, Taiwan (pp. 3–14). http://dx.doi.org/10.1109/ICDE.1995.380415

Aghababyan, A. (2014). *E3: Emotions, engagement, and educational digital games*. All Graduate Theses and Dissertations. Paper 4031. http://digitalcommons.usu.edu/etd/4031

Baker, R. S. J. d. (2010). Data mining for education. In B. McGaw, P. Peterson, E. Baker (Eds.), *International Encyclopedia of Education* (3rd ed.), vol. 7, pp. 112–118. Oxford, UK: Elsevier.

Baker, R. S. J. d., Hershkovitz, A., Rossi, L. M., Goldstein, A. B., & Gowda, S. M. (2013). Predicting robust learning with the visual form of the moment-by-moment learning curve. *Journal of the Learning Sciences*, *22*(4), 639–666. http://dx.doi.org/10.1080/10508406.2013.836653

Berland, M., Martin, T., Benton, T., Petrick Smith, C., & Davis, D. (2013). Using learning analytics to understand the learning pathways of novice programmers. *Journal of the Learning Sciences*, *22*(4), 564–599.http://dx.doi.org/10.1080/10508406.2013.836655

Blikstein, P., Worsley, M., Piech, C., Sahami, M., Cooper, S., & Koller, D. (2014). Programming pluralism: Using learning analytics to detect patterns in the learning of computer programming. *Journal of the Learning Sciences*, *23*(4), 561–599. http://dx.doi.org/10.1080/10508406.2014.954750

Buchta, C., Hahsler, M., Buchta, M. C., & Matrix, I. (2007). The arulesSequences Package. https://cran.r-project.org/.

Cheshire, A., Ball, L. J., & Lewis, C. N. (2005). Self-explanation, feedback and the development of analogical reasoning skills: Microgenetic evidence for a metacognitive processing account. In B. G. Bara, L. Barsalou, & M. Bucciarelli (Eds.), *Proceedings of the 27th Annual Conference of the Cognitive Science Society* (CogSci 2005), 21–23 July 2005, Stresa, Italy (pp. 435–441). Mahwah, NJ: Erlbaum.

Fazio, L. K., & Siegler, R. S. (2013). Microgenetic learning analysis: A distinction without a difference. *Human Development*, *56*(1), 52–58. http://dx.doi.org/10.1159/000345542

Gabadinho, A., Ritschard, G., Müller, N. S., & Studer, M. (2011). Analyzing and visualizing state sequences in R with TraMineR. *Journal of Statistical Software, 40*(4), 1–37. http://dx.doi.org/10.18637/jss.v040.i04

Kinnebrew, J., & Biswas, G. (2012). Identifying learning behaviors by contextualizing differential sequence mining with action features and performance evolution. In K. Yacef, O. Zaïane, A. Hershkovitz, M. Yudelson, & J. Stamper (Eds.), *Proceedings of the 5th International Conference on Educational Data Mining* (EDM 2012), 19–21 June, 2012, Chania, Greece (pp. 57–64). International Educational Data Mining Society.

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114. http://dx.doi.org/10.18608/jla.2016.33.6

Kuhn, D. (1995). Microgenetic study of change: What has it told us? *Psychological Science*, *6*(3), 133–139. http://dx.doi.org/10.1111/j.1467-9280.1995.tb00322.x

Low-Kam, C., Raïssi, C., Kaytoue, M., & Pei, J. (2013). Mining statistically significant sequential patterns. *Proceedings of the 13th International Conference on Data Mining* (ICDM 2013), 7–10 December 2013, Dallas TX, USA (pp. 488–497). IEEE. http://dx.doi.org/10.1109/ICDM.2013.124

Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M., & Hornik, K. (2015). Cluster: Cluster analysis basics and extensions. R package version 2.0.3. https://cran.r-project.org/

Martin, T., Philips, R., Smith, C., Horstman, T., Janisiewicz, P., & Aghababyan, A. (2013). Maximizing the use of human coders and automated techniques to study learning in educational games. Paper presented at the 15th Biennial Conference of the European Association for Research in Learning and Instruction (EARLI 2013), 27–30 August 2013, Munich, Germany.

Martin, T., Forsgren Velasquez, N., Aghababyan, A., Maughan, J., & Janisiewicz, P. (2014). Microgenetic designs for educational data mining research. Poster presented at the 7th International Conference on Educational Data Mining (EDM 2014), 4–7 July, London, UK.

Neuwirth, E., & Neuwirth, M. E. (2007). The RColorBrewer Package. https://cran.r-project.org/

Opfer, J. E., & Thompson, C. A. (2008). The trouble with transfer: Insights from microgenetic changes in the representation of numerical magnitude. *Child Development*, *79*(3), 788–804. http://dx.doi.org/10.1111/j.1467-8624.2008.01158.x

Pahl, C., & Donnellan, C. (2003). Data mining technology for the evaluation of web-based teaching and learning systems. In M. Driscoll & T. Reeves (Eds.), *Proceedings of the World Congress on E-Learning in Corporate, Government, Healthcare, and Higher Education* (E-Learn 2013), 21 October 2013, Montreal, QC, Canada (pp. 1–7). Chesapeake, VA: Association for the Advancement of Computing in Education.

Perera, D., Kay, J., Koprinska, I., Yacef, K., & Zaïane, O. R. (2009). Clustering and sequential pattern mining of online collaborative learning data. *IEEE Transactions on Knowledge and Data Engineering*, *21*(6), 759-772. http://dx.doi.org/10.1109/TKDE.2008.138

Pine, K. J., Lufkin, N., Kirk, E., & Messer, D. (2007). A microgenetic analysis of the relationship between speech and gesture in children: Evidence for semantic and temporal asynchrony. *Language and Cognitive Processes*, *22*(2), 234–246. http://dx.doi.org/10.1080/01690960600630881

Sanjeev, A. P., & Zytkow, J. M. (1995). Discovering enrollment knowledge in university databases. *Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining* (KDD-95), 20–21 August 1995, Montreal, QC, Canada (pp. 246–251).

Shen, R. M., Yang, F., & Han, P. (2003). A dynamic self-organizing e-learner communities with improved multi-agent matchmaking algorithm. *Lecture Notes in Computer Science, 2903,* 590–600. http://dx.doi.org/10.1007/978-3-540-24581-0_50

Siegler, R. S., & Crowley, K. (1991). The microgenetic method: A direct means for studying cognitive development. *American Psychologist*, *46*(6), 606–620. http://dx.doi.org/10.1037/0003-066X.46.6.606

Siegler, R. S., & Jenkins, E. (1989). How children discover new strategies. Hillsdale, NJ: Erlbaum.

(2016). Microgenetic learning analytics methods: Workshop report. *Journal of Learning Analytics*, *3*(3), 96–114.
http://dx.doi.org/10.18608/jla.2016.33.6

Stigler, J., Givvin, K., & Thompson, B. (2010). What community college developmental mathematics students understand about mathematics. *MathAMATYC Educator, 1*(3), 4–16.

Trčka, N., Pechenizkiy, M., & van der Aalst, W. (2010). Process mining from educational data. In C. Romero, S. Ventura, M. Pechenizkiy, & R. S. J. d. Baker (Eds.), *Handbook of educational data mining* (pp. 123–142). Boca Raton, FL: Taylor & Francis.

Ward, J. H., Jr. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association, 58*, 236–244.

Wu, L. L. (2000). Some comments on "Sequence analysis and optimal matching methods in sociology: Review and prospect." *Sociological Methods & Research, 29,* 41–64.

Zaïane, O. R., & Luo, J. (2001). Towards evaluating learners' behaviour in a web-based distance learning environment. *Proceedings of the IEEE International Conference on Advanced Learning Technologies* (ICALT 2001), 6–8 August 2001, Madison WI, USA (pp. 357–360). http://dx.doi.org/10.1109/ICALT.2001.943944

Zaki, M. J. (2001). SPADE: An efficient algorithm for mining frequent sequences. *Machine Learning Journal, 42*, 31–60. http://dx.doi.org/10.1023/A:1007652502315

Zhou, M., Xu, Y., Nesbit, J. C., & Winne, P. H. (2010). Sequential pattern analysis of learning logs: Methodology and applications. In C. Romero, S. Ventura, M. Pechenizkiy, & R. S. J. d. Baker (Eds.), *Handbook of educational data mining* (pp. 107–121). Boca Raton, FL: Taylor & Francis.