

AGSuite: Software to conduct feature analysis of artificial grammar learning performance

Matthew T. Cook¹ · Chrissy M. Chubala¹ · Randall K. Jamieson¹

© Psychonomic Society, Inc. 2017

Abstract To simplify the problem of studying how people learn natural language, researchers use the artificial grammar learning (AGL) task. In this task, participants study letter strings constructed according to the rules of an artificial grammar and subsequently attempt to discriminate grammatical from ungrammatical test strings. Although the data from these experiments are usually analyzed by comparing the mean discrimination performance between experimental conditions, this practice discards information about the individual items and participants that could otherwise help uncover the particular features of strings associated with grammaticality judgments. However, feature analysis is tedious to compute, often complicated, and ill-defined in the literature. Moreover, the data violate the assumption of independence underlying standard linear regression models, leading to Type I error inflation. To solve these problems, we present AGSuite, a free Shiny application for researchers studying AGL. The suite's intuitive Web-based user interface allows researchers to generate strings from a database of published grammars, compute feature measures (e.g., Levenshtein distance) for each letter string, and conduct a feature analysis on the strings using linear mixed effects (LME) analyses. The LME analysis solves the inflation of Type I errors that afflicts more common methods of repeated measures regression analysis. Finally, the software can generate a number of graphical representations of the data to support an accurate interpretation of results. We hope the ease and availability of these tools will encourage researchers to take full advantage of item-level variance in

their datasets in the study of AGL. We moreover discuss the broader applicability of the tools for researchers looking to conduct feature analysis in any field.

Keywords Shiny application · Artificial grammar learning · Feature analysis · Linear mixed effects analysis

The ability to learn the statistical regularities of language is a hallmark of human cognition, and therefore of great interest to cognitive scientists. However, the study of language learning is greatly complicated by factors outside of experimental control: participants arrive to the laboratory with a unique and unknown pattern of exposure to linguistic stimuli. To simplify the problem, researchers often study how people learn novel artificial languages that are constructed according to rules, analogous to the grammars that define natural language. As with natural language, passing exposure to an artificial language is sufficient to produce behavior consistent with sensitivity to its regularities.

One method of studying how people learn the grammatical structure of an artificial language is the artificial-grammar learning (AGL) task (Miller, 1958; Reber, 1967). In the AGL task, participants study letter strings composed according to the rules of an artificial grammar (see Fig. 1). These rules specify which letters can begin and end strings, as well as which letters can follow one another from left to right. In a subsequent test phase, participants are shown novel strings that either conform to (i.e., *grammatical* items) or violate (i.e., *ungrammatical* items) the rules of the grammar, and they must provide a numerical judgment of grammaticality for each test item. In a typical judgment-of-grammaticality test, ratings may range from –100 to +100, where a negative rating indicates that an item is believed to be ungrammatical, and a positive rating indicates that an item is believed to be grammatical.

✉ Matthew T. Cook
cookm346@myumanitoba.ca

¹ Department of Psychology, University of Manitoba,
Winnipeg, Manitoba R3T 2N2, Canada

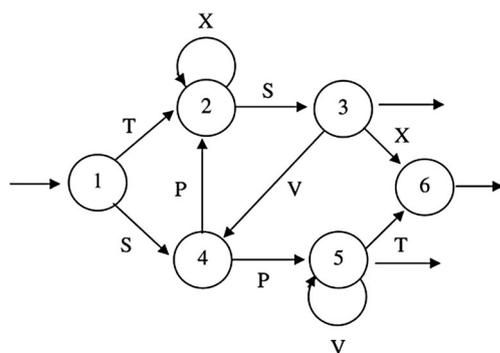


Fig. 1 A finite-state machine depicting the rules of an artificial grammar. To generate a grammatical stimulus, one enters the grammar at the leftmost node (marked 1) and follows the presented paths (indicated by arrows) until reaching one of the exit nodes on the right-hand side. When a path is taken, the associated letter is added to the end of the string. Any string that cannot be generated in this manner is ungrammatical

The typical findings in AGL experiments are that people can discriminate unstudied grammatical from ungrammatical items at a rate greater than chance, but that they cannot articulate the rules of the grammar or the basis of their judgments. Whereas a 50-year database of AGL experiments demonstrates *that* people can make this discrimination, *how* people make this discrimination remains a point of contention. One explanation is that participants implicitly internalize some of the rules of the underlying grammar and judge the test strings accordingly (e.g., Reber, 1967). A second explanation is that participants learn about fragments in the training strings (e.g., bigrams and trigrams) and subsequently rate the test strings according to their inclusion of those learned fragments (e.g., Perruchet & Pacteau, 1990). A third explanation is that participants learn the individual training strings and subsequently judge the test strings by their global similarity to the training list (e.g., Vokey & Brooks, 1992). A fourth explanation is that participants judge test strings by a combination of implicit grammatical knowledge and similarity-based inference (e.g., McAndrews & Moscovitch, 1985).

The enduring difficulty in determining the basis of participants' judgments stems not from the quantity of data collected, but from the quality of the analysis applied to the data. An AGL experiment with 50 participants, each providing judgments for 50 test items, yields 2,500 observations. Yet the standard approach to these data reduces the observations to just two numbers: the mean judgment for grammatical items, and the mean judgment for ungrammatical items. At this coarse level of analysis, all of the above explanations of AGL provide plausible accounts of the results. Consequently, a standard group-level analysis of the data cannot distinguish between the competing accounts; a more fine-grained analysis is necessary.

Consider the hypothetical test data in Table 1, which represent a plausible scenario in an AGL experiment. Although the standard finding, that grammatical test strings are judged

Table 1 Hypothetical test data from an artificial-grammar learning experiment

Grammatical strings	JOG	Ungrammatical strings	JOG
TXXSVPVVT	-11	PPSVP	57
TXSX	51	TXPVPT	13
SPSX	72	TPXXSVPV	-7
<i>Mean</i>	<i>37.33</i>	<i>Mean</i>	<i>21.00</i>

JOG = judgment of grammaticality, on a scale from -100 (*ungrammatical*) to +100 (*grammatical*).

as being more grammatical ($M = 37.33$) than ungrammatical test strings ($M = 21.00$), is reproduced, the judgments of individual items need not reflect this overall difference. In this example, the lowest rating is given to a grammatical item, whereas the second-highest rating is given to an ungrammatical item. These potentially meaningful nuances in the judgments of individual test strings are discarded in the group-level analysis.

But, instead of discarding this information, researchers can leverage it. Specifically, judgments of grammaticality for individual items can provide clues about the properties or features of strings that guide performance. For example, it is possible that the grammatical string TXXSVPVVT is judged as ungrammatical because it much longer than most of the training strings the participants studied. Likewise, it is possible that the ungrammatical string TXPVPT is judged as grammatical because it shares many bigrams (i.e., two-letter chunks) with the grammatical training strings. Other examples of string features proposed in the AGL literature include legal entry into the grammar (Redington & Chater, 1996), global similarity between the training and test strings (Vokey & Brooks, 1992), and letter chunks of varying sizes that are shared between the training and test strings (Perruchet & Pacteau, 1990). Analysis of the features that guide performance is not in itself a new idea. Indeed, the AGL literature is replete with examples of factorial experiments designed to tease apart the contributions of various string features, and thereby distinguish between the several accounts of the phenomenon (e.g., Jamieson, Nevzorova, Lee, & Mewhort, 2016; Johnstone & Shanks, 2001; Kinder, 2000; Kinder & Lotz, 2009; Vokey & Brooks, 1992). Although the value of such analyses cannot be denied, they are limited by the a priori nature of their experimental design. Namely, the factorial design assumes that the strings are equivalent in all features except those manipulated explicitly by the experimenter. Not only does this assumption fail to consider the possibility of confounding variables, it also limits experimenters to an analysis of only a small subset of the features that might define their training and test strings. Even worse, a forced and artificial manufacturing of materials can cue participants to the very features that define the orthogonalized list of materials

(Higham & Brooks, 1997). In summary, this practice discards information about the many other features that participants may be using to make their judgments, beyond those under direct examination.

We argue instead for an a posteriori feature analysis of existing data, in which each of several features (e.g., global similarity, chunk similarity, legal entry) is measured for each test string presented in an experiment. Those features are subsequently entered as predictors of participants' judgments in a regression analysis. The results of such a feature analysis provide a more nuanced understanding of AGL performance, allowing researchers to untangle the bases of participants' judgments and to understand their judgments for novel strings.

We are not the first to argue for such an analysis. Lorch and Myers (1990) proposed a method of fine-grained analysis in repeated measures designs. In their analysis, each participant's score on some dependent variable is regressed on multiple predictors, and hypothesis testing is conducted on the resulting regression coefficients. Johnstone and Shanks (1999) first introduced Lorch and Myers's technique to the AGL field, using it to discount the need for two distinct mechanisms guiding judgments, depending on the extent of exposure to the underlying grammar. The method gained some traction in the field (e.g., Scott & Dienes, 2008) but soon was shown to be statistically unsound (Baayen, Davidson, & Bates, 2008). In particular, the method fails to correctly partition variance, leading to an inflation of Type I error rates. More recently, linear mixed effects (LME) analysis has emerged within the field of psycholinguistics as a favored alternative for item-level feature analysis (Baayen, 2008). This method correctly partitions the total variability by treating items as a random rather than a fixed effect, thereby controlling Type I error inflation. Using LME analysis, researchers can assess the extent to which different candidate features of strings in an AGL experiment contribute to participants' judgments while maintaining the true Type I error rate near the nominal level.

Despite its clear theoretical benefits, there are practical barriers to implementing LME analysis. First, computing the features of the test strings, such as measures of grammaticality and similarity, by hand is time-consuming, tedious, and error-prone. Moreover, these measures have been ill-defined in the literature, such that the methods for computing them are not agreed upon. Second, although LME analysis has proven to be a statistically sound approach to feature analysis, the unfamiliarity of the method has hindered its widespread adoption within psychology; LME is an advanced statistical technique that is more difficult to understand than more familiar linear multiple regression approaches. Although the latter approaches are not suitable, due to their inflation of the Type I error rate, easily implemented alternatives for more advanced techniques such as LME are not readily available.

With these practical barriers in mind, we have developed a free Web-based software suite with the intention of promoting feature analysis in AGL experiments. The AGSuite software comprises three integrated modules, plus a fourth module that computes a Lorch–Myers (1990) style analysis that we will present later. The first module of AGSuite generates letter strings that conform to the rules of a finite-state grammar, drawing from either commonly used grammars in the literature or a user-defined grammar. A second module quickly and reliably computes 18 commonly assessed letter string features from lists of the training and test strings uploaded by the user as text files. This module also contains a link to a descriptive catalog of all 18 feature measures, as well as examples of the calculations used. In this way, we have removed any uncertainty as to the description or computation of the features. Finally, a third module merges the generated feature measures from the second module with the user's original AGL dataset and conducts LME analysis to determine the extents to which the different features capture variability in the data. As we mentioned above, LME analysis resolves the inflation of Type I errors that standard linear regression and related approaches suffer (e.g., Lorch & Myers, 1990) by expanding the error term with corrected variance partitioning. A full treatment of LME analysis and variance partitioning is beyond the scope of this article, but we invite the reader to refer to existing texts (e.g., McCulloch & Searle, 2001; Searle, Casella, & McCulloch, 1992); for a more applied focus, see Baayen (2008), who provides instructions on conducting LME analysis in the R computing language. Finally, the three modules can be used in conjunction, allowing researchers to both generate AGL stimuli and conduct a complete feature analysis on the collected data. Alternatively, the modules can be used independently as needed. We believe that the ease and flexibility of AGSuite will help researchers conduct more in-depth analysis of their AGL data, leveraging variance in the profile of item-level responding that is often ignored.

Artificial Grammar Suite (AGSuite)

AGSuite can be accessed online at https://mcook.shinyapps.io/AG_suite/. The four interconnected modules can be accessed via tabs at the top of the program. We describe the use of each module in turn before providing a demonstration. See Fig. 2 for screenshots of the program.

Generating strings

The Generate Strings module allows researchers to generate grammatical training and test strings from a database of published grammars or from a user-defined grammar. The left pane of the interface contains a dropdown menu for selecting from existing grammars. When a grammar is

a

b

Test String	String Length	Legal Entry	Mean Levenshtein	Global ACS
SPVPS	6	1	3.60	6.600000
XXSX	4	0	3.75	9.333333
PXOXS	6	0	4.85	7.600000
TXVS	4	1	3.25	1.666667
SPXOS	6	1	4.00	8.800000
TXS	3	1	3.45	10.000000
SVVT	5	1	3.75	6.750000
TPSX	4	1	3.15	4.666667
TXOSV	6	1	4.00	6.800000
TSVV	5	1	3.40	5.416667
TXOXS	6	1	4.25	8.300000
TSVPSX	6	1	3.20	12.200000
SPVST	5	1	3.55	4.500000
SPXS	4	1	2.95	10.666667
XPVVT	5	0	3.85	4.416667
TXSS	4	1	3.55	6.166667
SPBVP	5	1	2.90	14.000000
TXVPS	6	1	3.20	12.750000
SSPVT	5	1	3.75	4.750000
TXOSVP	6	1	3.50	12.300000
TSVPT	6	1	3.25	8.850000
SPVVT	6	1	3.95	7.800000
SPVX	5	1	3.20	7.333333
SPX	4	1	3.05	6.166667
SPP	3	1	3.25	5.000000

c

Linear mixed model fit by REML
 5-Data use: Restricted maximum likelihood approximation to degrees of freedom [1=Intercept]
 Formula: DV ~ fixed_effects + (1 + fixed_effects | subjects) + (1 + fixed_effects | items)

R2ML criterion at convergence: 24209.4

Fixed effects:		Estimate	Std. Error	df	t value	Pr(> t)
(Intercept)		-61.99393	14.14461	28.50000	-2.453	0.050008 **
fixed_effectsGlobal Entry		13.47619	7.32258	14.20000	2.880	0.014042 **
fixed_effectsGlobal Length		0.00597	3.02749	49.50000	0.213	0.989514
fixed_effectsGlobal Bigram ACS		6.14730	1.54271	32.49000	3.947	0.000399 ***

Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

d

Fig. 2 Screenshots of AGSuite. **(a)** The Generate Strings module. A grammar from the published artificial-grammar learning (AGL) literature is used here to generate ten training strings between four and eight letters in length. **(b)** The Compute Feature Measures module. The training and test items from Jamieson and Mewhort (2010, Exp. 3) are uploaded as text files, and the module computes 18 different feature measures for each test string. **(c)** The Linear Mixed Effects Analysis module. Judgment-of-grammaticality data are merged with feature measures (see panel

b), and a summary of the LME analysis is displayed. Outlined in the box are the results of interest: the name of each predictor included in the model, the coefficient estimates, standard errors, degrees of freedom, and the corresponding *t* and *p* values. **(d)** Items tab of the Linear Mixed Effects Analysis module. Histograms of judgments as a function of test item provide a graphical representation at the level of individual strings. The module offers similar output as a function of subjects, under the Subjects tab

selected, an accompanying diagram of the grammar appears (see Fig. 2A). Sliders allow researchers to define the minimum and maximum lengths of strings, as well as the number of strings to generate. If the desired number of strings exceeds the number that can be generated within the specified length range, a warning message notifies the user. The main (right) pane of this module is a table of the generated strings, which updates in real time as the user changes the variables in the left-hand pane. Each column is searchable and sortable. Once the set is constructed, the table can be saved as a comma-separated value (.csv) file by clicking the “Download” button. The file will be saved in the Web browser’s download folder.

Strings can also be generated from user-defined grammars by uploading a .csv file that defines the legal characters in the grammar as well as the legal transitions between them. A link above the upload button provides instructions for generating the matrix of a grammar, as well as an example of the formatting necessary for AGSuite to read it.

Computing feature measures

The Compute Feature Measures module computes feature measures for each letter string provided by the user (see Fig. 2B).¹ The user uploads the training and test strings as text (.txt) files with one string per line. Once the file is uploaded, each test item is compared to all training items. Eighteen different measures can be computed for each test item: string length; legal first letter; minimum and mean Levenshtein distance; global and anchor bigram, trigram, and overall associative chunk strength; bigram, trigram, and overall chunk novelty; bigram, trigram, and overall novel chunk proportion; first-order redundancy; and analogical similarity. The checkboxes in the left-hand pane determine which measures are displayed in the table to the right. The “Measure descriptions” link contains a catalog of each measure’s description and example calculations for each (see the Appendix). Each column of the measures table is searchable using the text box below the columns, and sortable using the arrows above the columns. Once the calculations are complete, users can download a .csv file using the “Download” button at the bottom of the left-hand pane. That file will be downloaded to the browser’s download folder and can be viewed with any standard spreadsheet program (e.g., Microsoft Excel, Apache OpenOffice, or Google Docs). These computed feature measures can be used as descriptive statistics to inspect the relationship between the test and training items. However, in the context of AGSuite, these measures serve as the predictors available to conduct an LME analysis. They are directly available to the LME module, which uses them to determine the extent to which features correlate with the profile of item-level judgments.

¹ This module is also available as a standalone R package, the source code for which can be downloaded from <https://github.com/cookm346/AGSuite>.

Conducting linear mixed effects analysis

The Linear Mixed Effects Analysis module (simply the *LME module* hereafter) allows researchers to conduct an LME analysis (see Fig. 2C) using the feature measures (e.g., string length, associative chunk strength) generated in the Compute Feature Measures module just described. To conduct the analysis, the user uploads judgment-of-grammaticality data as a .csv file using the button in the left-hand pane. These data require three variables, such that each row contains a participant identifier, a test string, and the participant’s corresponding grammaticality judgment for each test string (see Fig. 3). Once the data are uploaded, three dropdown boxes in the left-hand pane will be populated with the variable names from the data file, such that the user can define the dependent variable, the subjects variable, and the items variable from the dataset. Similarly, a dropdown menu populated with the predictors from the Compute Feature Measures module allows the user to select which predictor(s) to include in the analysis.

The statistical analysis itself is conducted by the *lme4* R package (Bates, Maechler, Bolker, & Walker, 2015); our addition of an intuitive point-and-click method allows users to quickly test different LME models without having to specify the formulae describing the model. For instruction on using, understanding, and troubleshooting issues with the *lme4* package (such as convergence failures), we recommend that users consult the *lme4* package documentation, currently located at <https://cran.r-project.org/web/packages/lme4/lme4.pdf>.

The main results pane of the LME module has three subsections, accessed by tabs. The first subsection displays a summary of the model being tested. The information of particular interest to researchers analyzing AGL data is the list of regression coefficient estimates for each predictor, along with accompanying *t* and *p* values, given in the table titled *Fixed effects*. This summary output also includes a description of the residuals from the regression analysis, titled *Scaled residuals*, a table of variance partitioning titled *Random effects*, and the correlations between predictors, titled *Correlation of Fixed Effects*. The second subsection displays tables of the coefficients for each predictor as a function of subjects, or a histogram for each individual subject’s judgments of grammaticality. The third subsection displays tables of the coefficients for each predictor as a function of test items, or a histogram for each individual item’s judged grammaticality (Fig. 2D).

Lorch–Myers (1990) analysis

Although AGSuite provides a solution to known problems with the Lorch–Myers (1990) technique for analyzing item-level judgments in within-subjects designs (Baayen et al., 2008), the Lorch–Myers method is still used in the literature. Therefore, we included a fourth, optional module that conducts an analysis of item-level judgments using Lorch and

	A	B	C
1	subject_ID	test	Judgement
2	S1	SPTVPS	76
3	S1	XXSX	-52
4	S1	PXXXXS	-69
5	S1	TXVS	15
6	S1	SPXXXX	32
7	S1	TXS	9
8	S1	SVVVT	-79
9	S1	TPSX	72
10	S1	TXSXV	-48
		.	
		.	
		.	
654	SN	SPTVPS	-14
655	SN	SPVST	67
656	SN	SPXS	-25
657	SN	SVVVT	-44
658	SN	XXPVT	66
659	SN	SPVVVT	-3
660	SN	TPSVPT	-65
661	SN	TXSVPS	-70
662	SN	TSVPSX	79

Fig. 3 Screenshot of the required data format for AGSuite. Column A identifies the subject (i.e., *subject_ID*). The subject identifiers can be in any format the researcher chooses. Column B lists the test items judged by each participant (i.e., *test*); the program requires one observation for each item per participant, but the order in which the items are listed does not need to be consistent across participants. Column C lists the judgments of grammaticality (i.e., *Judgement*). Although column headings are required so that the users of AGSuite can identify and select those variables within the program, the particular column headings that we present are arbitrary; the user can use whatever labels are preferred. The data must be saved as a .csv file

Myers's technique. The purpose of including this module, despite its known shortcomings, is twofold. First, its inclusion allows researchers the option of using this more familiar regression method for conducting feature analysis if they wish. Second, it provides a familiar point of comparison for the LME method.

This module uses the same data as the LME module and is presented in a similar layout, in which users define the variables of their dataset and the predictors to be used in the model from

dropdown menus. Unlike in the LME module, the main pane of this module presents a subject-by-predictor table of regression coefficients, including the R^2 , F , and p values for each subject's regression model. More critically, the table includes the outcomes of single-sample t tests conducted on the coefficients for each predictor over subjects. These t tests are the statistics needed to decide whether each predictor included in the analysis is correlated with participants' item-level judgments of grammaticality.² We refer the reader to Lorch and Myers's (1990) original article for the details of the statistical method, and to Johnstone and Shanks (1999) for an applied example of how the technique can be used to conduct an a posteriori feature analysis of AGL data. As with the LME module, histograms for each subject and item are available via tabs at the top of the Lorch Myers Analysis module.

In summary, AGSuite is a set of four interconnected modules. The Generate Strings module generates training strings that conform to the rules of a grammar defined by the user. The strings can be generated either from grammars in the published AGL literature or from a custom grammar uploaded to the program by the user as a matrix. The Compute Feature Measures module allows researchers to upload training and test string and generates 18 commonly used feature measures for each test string by comparing each against all of the training strings. The LME module matches the features matrix generated by the Compute Feature Measures module of the program with the judgment data to be used in LME feature analysis. This module allows researchers a point-and-click method of conducting feature analysis on their original AGL data. The fourth module allows researchers to conduct a Lorch–Myers (1990) style analysis of their data for comparison against the LME results.

An empirical demonstration

The power of AGSuite comes from the ability to analyze existing data in the original format and identify patterns in the data that were either not asked about or not detectable by the original analysis. Having described the program, we now turn to a demonstration of how AGSuite can be used to conduct a feature analysis that will enrich our understanding of existing AGL data. The reader may follow along with the demonstration at https://mcook.shinyapps.io/AG_suite_demo/.

Jamieson and Mewhort (2010, Exp. 3) conducted a standard artificial-grammar experiment in which 47 participants studied 20 grammatical training strings and then rated the grammaticality of 50 test strings: 25 grammatical and 25 ungrammatical. Each rating was collected on a scale ranging from -100 (*ungrammatical*) to $+100$ (*grammatical*). Jamieson and Mewhort used the data to argue that an exemplar-based account

² Although we present the features as predictors, it is important to note that the a posteriori nature of the analysis supports a conclusion that the predictor variable correlates with performance, but it cannot justify the conclusion that participants used the feature to judge the test items.

of implicit learning accounted for performance at the level of individual items. However, whereas their analysis provided evidence in favor of one predictor of performance, global similarity, it failed to consider or rule out other potential predictors of performance. More critically, their category-level analysis of the data (i.e., by grammatical status) discarded a good deal of the available information contained in the item estimates by making group-level comparisons.

To remedy the problem, we reanalyzed Jamieson and Mewhort's (2010, Exp. 3) data using AGSuite. We first uploaded the training and test strings from Jamieson and Mewhort's original study into the Compute Feature Measures module as text files. We then uploaded their original data into the LME module as a .csv file. After specifying which variables in the dataset corresponded to the subjects, items, and dependent variable identifiers using the dropdown menus, we specified three measures to use as predictors of peoples' judgments: legal first letter (a binary value that indicates whether a test string's first letter matches the first letter of any studied training string), string length (the number of letters in a test string), and global bigram associative chunk strength (the similarity of a test string's two-letter chunks to those of all the training strings).

Panel C of Fig. 2 presents the summary output from the analysis. As we described above, the main results pane provides a description of the residuals from the regression analysis, a table of variance partitioning, and the correlations between predictors. The information of greatest interest to us is outlined in the table titled *Fixed effects*, which provides estimated coefficients for each predictor included in the model, as well as the associated *t* and *p* values.

The results show that the legal entry predictor, which measures whether a test string includes a rule violation at the first letter of the test items, was reliably correlated with judgments of grammaticality, $\beta = 19.68, p = .016$. Thus, a rule violation by the first letter of the string reduces judgments of grammaticality by 19.68 judgment units on average. This suggests that participants are quite sensitive to rule violations at the beginning of a string. In fact, Jamieson and Mewhort (2010) noted in passing that test items beginning with a T or an S tended to be rated as more grammatical than those beginning with other letters, a fact consistent with their grammar, in which all of the training items began with T or S (Fig. 2A shows the grammar they used). By conducting a feature analysis, we not only found statistical support for their observation, but also quantified the effect in a way that was not possible with a blunt group-level analysis.³

³ As our example implies, the method is appropriate for an analysis with continuous judgments of grammaticality. We recommend caution when applying the methods to categorical judgments of grammaticality. Although several research groups have applied the Lorch–Myers (1990) technique to categorical judgments, the same limitations apply to that variation on the technique, and thus the same caution is warranted.

Likewise, global bigram associative chunk strength, a measure of bigram (i.e., two-letter chunk) similarity between the training and test strings, was reliably correlated with judgments of grammaticality, $\beta = 6.17, p < .001$. Thus, participant's judgments were correlated with differences in the frequencies with which two-letter chunks appeared in the training strings. By contrast, increasing the string length did not reliably affect judgments of grammaticality, $\beta = 0.04, p = .99$. This is not surprising, given the restricted range of the test string length (i.e., 3 to 6); had there been test items that were drastically longer than many of the training items, this feature might have correlated very strongly with participants' judgments.

In summary, whereas Jamieson and Mewhort (2010) provided evidence that global similarity of the training and test strings could account for participants' overall performance in an AGL experiment, their original analysis failed to exclude other possible explanations or to describe performance at a finer grain than the group means (i.e., hits and false alarms). AGSuite, on the other hand, allowed us to quickly and easily quantify the extent to which a variety of test string features were correlated with the judgments of grammaticality. Moreover, the obtained predictor coefficients for the three predictors included in the LME analysis can be used to make quantitative predictions about how participants might respond to strings not presented in the original experiment. More importantly, our approach supports a more inclusive analysis of judgments, because it does not require that the stimulus features of interest be included in the a priori experimental design.

General discussion

Researchers have used the artificial-grammar learning task as a simplified means of examining language acquisition (e.g., Miller, 1958; Reber, 1967). Yet, despite more than five decades of demonstrations that participants *can* discriminate grammatical from ungrammatical test strings, our understanding of *how* they do so remains debated. We argue that this is not due to a limitation of the data collected, but rather to the nature of the analyses typically applied to AGL data.

In this article, we have presented AGSuite as a user-friendly tool to conduct a more thorough and informative analysis of item-level judgments of grammaticality. Our example reanalysis of Jamieson and Mewhort's (2010) data makes the point. In a standard analysis, they were limited to the conclusion that global similarity between the training and test strings allowed participants to discriminate grammatical from ungrammatical test items. By the feature analysis we presented, we compared the influences of three string features to determine that both legal entry and global bigram associative chunk strength were correlated with judgments of grammaticality, whereas string length was not. Once participants' judgments are explained

and predicted at the level of string features, the group-level predictions follow naturally.

Several research groups have approached the exploration of features via experimental design, whereby the features of strings are crossed factorially and their relative contributions are gauged by group-level comparisons between the conditions (e.g., Jamieson, Nevzorova, et al., 2016; Johnstone & Shanks, 2001; Kinder, 2000; Kinder & Lotz, 2009; Vokey & Brooks, 1992). Others have made use of alternative statistical techniques that offer more fine-grained analyses. For instance, the technique proposed by Lorch and Myers (1990) gained some traction in the field of AGL (e.g., Johnstone & Shanks, 1999; Scott & Dienes, 2008) before it was shown to be statistically unsound (Baayen et al., 2008). LME analysis has emerged as a promising alternative to the Lorch–Myers technique, because it conducts feature analysis without inflating the Type I error rate beyond the nominal level (Baayen, 2008; McCulloch & Searle, 2001; Searle et al., 1992).

Critically, however, AGSuite simplifies what would otherwise be an arduous task: Computing numerous feature measures for each test string is time-consuming and error-prone. At the same time, the LME analysis is an unfamiliar and relatively difficult statistical technique. AGSuite solves both problems with the ease of an intuitive Web-based user interface. This freely provided suite of interconnected modules provides a simple means of generating AGL test materials from a database of known grammars, computing feature measures for each test string, and conducting LME analysis to determine the extents to which those features are correlated with participants' profiles of item-level judgments. By providing this set of tools, we aim to highlight the value of feature analysis in AGL research in particular and to provide an efficient, standardized, and explicit means to encourage its systematic adoption.

Further applications of AGSuite

Although AGSuite was developed with the study of AGL in mind, the suite's structure provides a foundation that can be modified to generalize its use to other domains of psychological investigation. Below we outline domains that would benefit from awareness and use of feature analysis, though we emphasize that the potential applicability extends even further beyond this handful of examples.

MacLeod, Gopie, Hourihan, Neary, and Ozubko (2010) presented evidence that participants remember words that they speak aloud better than words that they read silently—a phenomenon called the *production effect* (see also Jamieson, Mewhort, & Hockley, 2016; Jamieson & Spear, 2014). They explained the memory benefit for spoken words as a benefit of memorial distinctiveness conferred by the act of production (cf. Bodner, Taikh, & Fawcett, 2014). Feature analysis can be applied to data from experiments on the production effect to identify the features that participants encode when they

produce a word, as compared to the features that they encode when they read but do not produce a word. For example, it may be that phonological features predict memory for produced words, whereas orthographic features predict memory for words read silently. To solve the issue, one can correlate the item-level recognition judgments with phonological and orthographic features of the training and test words.

Feature analysis might also help uncover the basis for decisions in recognition memory experiments. For example, one might use the technique to identify the stimulus properties to which participants attend (e.g., word frequency, concreteness, or imageability) as a function of the study instructions. The technique might also be used to reevaluate the levels-of-processing theory (e.g., Craik & Lockhart, 1972) by identifying the features that participants attend to and encode under shallow- versus deep-processing instructions. A corollary of using feature analysis in such circumstances is a redefinition of verbal concepts, such as distinctiveness or deep processing, in terms of specific and quantifiable predictors already available in databases such as the MRC Psycholinguistic (Coltheart, 1981) and the SUBTLEX-UK (Van Heuven, Mandera, Keuleers, & Brysbaert, 2014) databases.

The applicability of the tool, of course, is not limited to memory. Consider an analysis of perceptual judgment tasks. One might, for example, collect participants' judgments of visual patterns (e.g., preference judgments) and subsequently use feature analysis to identify which stimulus variables (e.g., symmetry, complexity, and density) predict judgments across items. The same analysis could examine preferences, or any other ratings, concerning auditory patterns, tactile patterns, tastes, or odors.

Finally, feature analysis can be especially valuable to those who conduct experiments on populations that are difficult to recruit or that are time-consuming and costly to study. Consider, for instance, studies of language acquisition in infants, a demographic that is notoriously difficult to recruit and from which only a handful of responses can be collected (e.g., Ko, Soderstrom, & Morgan, 2009). Rather than taking the mean of responses across such participants, thereby discarding the bulk of the hard-earned data, researchers can leverage the full set of responses, using feature analysis to uncover patterns that would be otherwise difficult to detect.

In all instances, AGSuite offers an excellent foundation from which other applications can build. Consider experiments that make use of natural linguistic stimuli. The Generate Strings module of AGSuite currently generates test strings based on the rules of a preselected artificial grammar, in line with a string length range specified by the user. To generate natural linguistic stimulus sets, the program could be altered to select words from a psycholinguistic database, in line with ranges of familiarity, concreteness, imageability, or other factors specified by the user. Similarly, the Generate Feature Measures module of AGSuite currently compares test

and training items on the basis of letter and chunk similarity metrics common in the AGL literature. AGSuite could easily be modified to extract feature measures from existing linguistic databases, such as SUBTLEXus (Brysbaert, & New, 2009), the English Lexicon Project (Balota et al., 2007), or various age-of-acquisition ratings (e.g., Kuperman, Stadthagen-Gonzalez, & Brysbaert, 2012). Alternately, measures of similarity could be computed from the semantic similarity of words imported from vector space models such as latent semantic analysis (Landauer & Dumais, 1997), BEAGLE (Jones & Mewhort, 2007), or HAL (Burgess & Lund, 2000). Because it pulls directly from the generated feature measures, the Linear Mixed Effects Analysis module of AGSuite would require no major alteration in order to accommodate data from any domain, provided that the data were correctly formatted.

No matter the particular research question, a good experiment yields rich information about the very complex processes underlying cognition and behavior. The practice of averaging responses over items to obtain group means may serve a useful simplifying function, but it discards valuable data and consequently limits our understanding of the complexities of the problems we are studying. There are a number of domains in which the LME technique can help us make use of item-level responses. In the domain of AGL, the issue was noted by Dienes in 1992 and adopted in analyses by Jamieson and Mewhort (2010) and Jamieson, Vokey, and Mewhort (2017). But, as we have outlined, the same techniques can help researchers make progress in investigations of learning, memory, and other domains.

It is time we made full use of the data we have collected. This argument is in line with the statistical techniques put forward by Lorch and Myers (1990), and subsequently refined by Baayen (2008) and others. Although we have not further developed their statistical arguments, we have brought their analytic techniques into practical use by providing a software suite that solves both technical and computational problems for researchers. It is our hope that offering AGSuite as a free and user-friendly Web-based tool will raise awareness of the value of feature analysis in AGL and other cognitive domains, and will encourage researchers to consider item-level as well as group-level measurements of their data.

Author note This research was supported by a University of Manitoba Undergraduate Research Award to M.T.C., an NSERC PGS-D to C.M.C., and an NSERC Discovery Grant to R.K.J.

Appendix

For all measures other than string length and first-order redundancy, each test string is compared to all strings in the training list.

String Length

The number of characters in a particular string.

Legal Entry

“Is the first letter of this test string the first letter in any training strings?”

The entry (i.e., first) letter of all the training strings is determined and the software reports if each test string's first letter appears as the first letter in any of the training strings. To the extent that first letters in the training strings cover all legal first letters as defined by the grammar, the measurement indicates if the string enters by an illegal route.

Note that this measure is a binary variable, where 0 = illegal entry and 1 = legal entry.

Min Levenshtein (Vokey & Brooks, 1992)

“How similar is this test string to the next most similar training string?”

This is calculated by determining the Levenshtein (edit) distance of a particular test string with all training strings and then reporting the lowest Levenshtein distance. Although we use the more general term from computer science (i.e., Levenshtein distance), the measure is formally equivalent to Vokey and Brooks's (1992) measurement called *edit distance*.

Example:

The test string AAC would be compared to all training strings. The Levenshtein distances for the test string AAC with each of the following training strings would be:

ABC: 1

The letter B is substituted for the second A in the training string.

CBA: 3

The letter B is deleted from the training string resulting in CA. Then, the C and A in CA are substituted resulting in AC. Finally, the insertion of an A results in AAC.

ABCDEF: 4

Deletions of the last 3 letters D, E, and F, as well as one substitution of the B to an A is required to edit the training string ABCDEF to the test string AAC.

The minimum Levenshtein distance between AAC and the study list ABC, CBA, and ABCDEF is equal to 1.

Mean Levenshtein (Vokey & Jamieson, 2014)

“How similar is this test string on average, to all training items?”

This is calculated by determining the Levenshtein (edit) distance of a particular test string with each training string and then averaging those Levenshtein distances.

Example:

Using the examples from the min Levenshtein distance above, the mean Levenshtein distance for the test string AAC with the trainings strings ABC, CBA, and ABCDEF is $((1 + 3 + 4)/3) = 2.67$.

Global Bigram Associative Chunk Strength (ACS; Johnstone & Shanks, 2001; Knowlton & Squire, 1996)

“How many times do this test string’s bigrams (two letter strings), relative to the number of bigrams in the test string, occur in any of the training strings?”

Each test string is decomposed into its bigrams. The number of times a test string’s bigrams occur in any of the training strings is summed and then divided by the number of bigrams in that particular test string.

Example:

The test string ABCD is decomposed into three bigrams AB, BC, and CD.

Comparing ABCD to the following training strings, we count the number of times each bigram occurs in any training string:

ABC: 2 (AB, BC)
 ABCDEF: 3 (AB, BC, CD)
 ACDE: 1 (CD)
 Total: 6
 Test string’s number of bigrams: 3
 Global bigram ACS: $6/3 = 2$

Global Trigram ACS

“How many times do this test string’s trigrams (three letter strings), relative to the number of trigrams in the test string, occur in any of the training strings?”

Each test string is decomposed into its trigrams. The number of times a test string’s trigrams occur in any of the training strings is summed and then divided by the number of trigrams in that particular test string.

Example:

The test string ABCD is decomposed into two trigrams of ABC and BCD.

Comparing ABCD to the following training strings, we count the number of times each trigram occurs in any training string:

ABC: 1 (ABC)
 ABCDEF: 2 (ABC, BCD)
 ACDE: 0
 Total: 3
 Test string’s number of trigrams: 2
 Global trigram ACS: $3/2 = 1.5$

Global ACS

“How many times do this test string’s bigrams (two letter strings) or trigrams (three letter strings) occur in any of the training strings?”

First Global Bigram ACS and Global Trigram ACS are calculated. Global ACS is a sum of these two measures.

Anchor Bigram ACS

“How often do a test string’s first and last bigrams (two letter strings) appear as the first and last bigrams in the training strings?”

The number of times a test string’s first and last bigram occur as the first and last bigrams in any of the training strings is summed and then divided by two.

Example:

The test string ABCD has anchor bigrams of AB and CD.

Comparing ABCD to the following training strings, we count the number of times each bigram occurs as the first and last bigrams in all training strings:

ABCE: 1 (AB)
 ABCDEF: 1 (AB)
 ACDE: 0
 Total: 2
 Test string’s number of anchor bigrams: 2
 Anchor Bigram ACS: $2/2 = 1$

Anchor Trigram ACS

“How often do a test string’s first and last trigrams (three letter strings) appear as the first and last trigrams in the training strings?”

The number of times a test string’s first and last trigram occur as the first and last trigrams in all of the training strings is summed and then divided by two.

Example:

The test string ABCD has anchor trigrams of ABC and BCD.

Comparing ABCD to the following training strings, we count the number of times each trigram occurs as the first and last trigrams in any training string:

ABCE: 1 (ABC)
 ABCDEF: 1 (ABC)
 ACDE: 0
 Total: 2
 Test string’s number of anchor trigrams: 2
 Anchor Trigram ACS: $2/2 = 1$

Anchor ACS

“How often do a test string’s first and last bigrams (two letter strings) or trigrams (three letter strings) appear as the first and last bigrams in the training strings?”

First Anchor Bigram ACS and Anchor Trigram ACS are calculated. Anchor ACS is a sum of these two measures.

Bigram Novelty

“How many of the bigrams (two letter strings) in this particular test string are NOT in any training strings?”

Bigram Novelty is a measure of the number of bigrams in a test string that do not occur in any of the training strings.

Example:

The test string ABCDEF decomposed into its five bigrams: AB, BC, CD, DE, and EF. Each of the bigrams is checked against each of the following three training strings:

ABC: 2 (AB, BC)
 ABDC: 0 (AB – repeat)
 ACDE: 2 (CD, DE)

Four of the five bigrams from the test string are observed in the training strings. Therefore, one of the five bigrams of the test string ABCDEF were not observed in the training strings (EF). The Bigram Novelty of the test string ABCDEF with the training strings ABC, ABDC, and ACDE is therefore 1.

Trigram Novelty

“How many of the trigrams (three letter strings) in this particular test string are NOT in any training strings?”

Trigram Novelty is a measure of the number of trigrams in a test string that do not occur in any of the training strings.

Example:

The test string ABCDEF is decomposed into its four trigrams: ABC, BCD, CDE, and DEF. Each of the trigrams is checked against each of the following three training strings:

ABC: 1 (ABC)
 ABDC: 0
 ACDE: 1 (CDE)

Two of the four trigrams from the test string are observed in the training strings (ABC, CDE). Therefore, two of the four trigrams of the test string ABCDEF were not observed in the training strings (BCD and DEF). The Trigram Novelty of the test string ABCDEF with the training strings ABC, ABDC, and ACDE is therefore 2.

Chunk Novelty

“How many of the bigrams (two letter strings) or trigrams (three letter strings) in the test string are NOT in any training strings?”

First Bigram Novelty and Trigram Novelty are calculated. Chunk Novelty is a sum of these two measures.

Bigram Novel Chunk Proportion (NCP)

“How many of the bigrams (two letter strings), relative to the test string’s length, are NOT in any training strings?”

Bigram NCP is a sum of all the times a bigram occurs in a test string when that bigram does not occur in any of the training strings (bigram novelty) divided by the number of that particular string’s bigrams (i.e., the number of letters a string has minus one).

Example:

The test string ABCDEF decomposed into the five bigrams: AB, BC, CD, DE, and EF. Each of the bigrams is checked against each of the following three training strings:

ABC: 2 (AB, BC)
 ABDC: 0 (AB – repeat)
 ACDE: 2 (CD, DE)

Four of the five bigrams from the test string are observed in the training strings. Therefore, one of the five bigrams of the test string ABCDEF were not observed in the training strings (EF). The Bigram NCP of the test string ABCDEF with the training strings ABC, ABDC, and ACDE is $1/5 = 0.2$.

Trigram NCP

“How many of the trigrams (three letter strings), relative to the test string’s length, are NOT in any training strings?”

Trigram NCP is a sum of all the times a trigram occurs in a test string when that trigram does not occur in any of the training strings (Trigram Novelty) divided by the number of that particular string’s trigrams (i.e., the number of letters a string has minus two).

Example:

The test string ABCDEF is decomposed into its four trigrams: ABC, BCD, CDE, and DEF. Each of the trigrams is checked against each of the following three training strings:

ABC: 1 (ABC)
 ABDC: 0
 ACDE: 1 (CDE)

Two of the four trigrams from the test string are observed in the training strings (ABC, CDE). Therefore, two of the four trigrams of the test string ABCDEF were not observed in the training strings (BCD and DEF). The Trigram NCP of the test string ABCDEF with the training strings ABC, ABDC, and ACDE is $2/4 = 0.5$.

NCP

“How many of the bigrams (two letter strings) or trigrams (three letter strings) in this particular test string are NOT in any training strings?”

First Bigram NCP and Trigram NCP are calculated. Chunk Novelty is a sum of these two measures.

First-Order Redundancy (Jamieson, Nevzorova, et al., 2016)

“How predictable are the bigrams (two letter strings) in this particular test string?”

The test string is first decomposed into its bigrams and the probability of each bigram appearing in the string is computed. First-Order Redundancy measures the predictability of the bigrams in the test string, by computing the complement of the average information relative to the maximum information possible defined as a string of the same length made up of

all unique letters (i.e., the denominator in the following formula),

$$R_1 = 1 - \frac{-\sum_{i=1}^m \sum_{j=1}^m p_{ij} \log_2(p_{ij})}{\log_2(n-1)}$$

where, p is the probability of each bigram, m is all possible letters A...Z, and n is the length of the string. Redundancy ranges from 0 (not at all redundant/predictable) to 1 (completely redundant/predictable).

Example:

The test string ABABABAB is decomposed into two bigrams: AB and BA. AB occurs four times and BA occurs three times, with seven bigrams total. This results in the probabilities of 0.57 and 0.43 for AB and BA, respectively. Using the first-order redundancy formula from above results in:

$$R_1 = 1 - \frac{-[0.57 * \log_2(0.57) + 0.43 * \log_2(0.43)]}{\log_2(8-1)} = 0.65$$

As another example, the test string BBBB BBBB is decomposed into one bigram: BB that occurs seven times, with seven bigrams total. This results in a probability of 1 for BB. The first-order redundancy using the formula above results in:

$$R_1 = 1 - \frac{-[1 * \log_2(1)]}{\log_2(8-1)} = 1$$

Analogical Similarity (Brooks & Vokey, 1991)

"Ignoring surface structure, how often does the pattern of symbols in this test string appear in all training strings?"

This measure computes the number of training strings that have an identical analogous pattern to the test string.

Example:

The pattern of the test string CBBA can be abstracted to a pattern of 1223. This pattern is checked against all training strings. The number of training strings that contain this pattern is the Analogical Similarity:

ABCD: (pattern: 1234, match: no)
 BAAC: (pattern: 1223, match: yes)
 AABC: (pattern: 1123, match: no)

The Analogical Similarity for the test string CBBA compared to the training strings ABCD, BAAC, and AABC is therefore 1.

References

- Baayen, R. (2008). *Analyzing linguistic data: A practical introduction to statistics using R*. Cambridge, UK: Cambridge University Press.
- Baayen, R. H., Davidson, D. J., & Bates, D. M. (2008). Mixed-effects modeling with crossed random effects for subjects and items. *Journal of Memory and Language*, 59, 390–412. doi:10.1016/j.jml.2007.12.005
- Balota, D. A., Yap, M. J., Cortese, M. J., Hutchison, K. A., Kessler, B., Loftis, B., ... Treiman, R. (2007). The English Lexicon Project. *Behavior Research Methods*, 39, 445–459. doi:10.3758/BF03193014
- Bates, D., Maechler, M., Bolker, B., & Walker, S. (2015). Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, 67, 1–48. doi:10.18637/jss.v067.i01
- Bodner, G. E., Taikh, A., & Fawcett, J. M. (2014). Assessing the costs and benefits of production in recognition. *Psychonomic Bulletin & Review*, 21, 149–154. doi:10.3758/s13423-013-0485-1
- Brooks, L. R., & Vokey, J. R. (1991). Abstract analogies and abstracted grammars: Comments of Reber (1989) and Matthews et al. (1989). *Journal of Experimental Psychology: General*, 120, 316–323.
- Bryshaert, M., & New, B. (2009). Moving beyond Kučera and Francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for American English. *Behavior Research Methods*, 41, 977–990. doi:10.3758/BRM.41.4.977
- Burgess, C., & Lund, K. (2000). The dynamics of meaning in memory. In E. Dietrich & A. B. Markman (Eds.), *Cognitive dynamics: Conceptual and representational change in humans and machines* (pp. 117–156). Mahwah, NJ: Erlbaum.
- Coltheart, M. (1981). The MRC psycholinguistic database. *Quarterly Journal of Experimental Psychology*, 33, 497–505. doi:10.1080/14640748108400805
- Craik, F. I. M., & Lockhart, R. S. (1972). Levels of processing: A framework for memory research. *Journal of Verbal Learning and Verbal Behavior*, 11, 671–684. doi:10.1016/S0022-5371(72)80001-X
- Dienes, Z. (1992). Connectionist and memory array models of artificial grammar learning. *Cognitive Science*, 16, 41–79.
- Higham, P. A., & Brooks, L. (1997). Learning the experimenter's design tacit: Sensitivity to the structure of memory lists. *Quarterly Journal of Experimental Psychology*, 50A, 199–215.
- Jamieson, R. K., & Mewhort, D. J. K. (2010). Applying an exemplar model to the artificial-grammar task: String-completion and performance for individual items. *Quarterly Journal of Experimental Psychology*, 63, 1014–1039.
- Jamieson, R. K., Mewhort, D. J. K., & Hockley, W. E. (2016). A computational account of the production effect: Still playing twenty questions with nature. *Canadian Journal of Experimental Psychology*, 70, 154–164.
- Jamieson, R. K., Nevzorova, U., Lee, G., & Mewhort, D. J. K. (2016). Information theory and artificial grammar learning: Inferring grammaticality from redundancy. *Psychological Research*, 80, 195–211.
- Jamieson, R. K., & Spear, J. (2014). The offline production effect. *Canadian Journal of Experimental Psychology*, 68, 20–28.
- Jamieson, R. K., Vokey, J. R., & Mewhort, D. J. K. (2017). Implicit learning is order dependent. *Psychological Research*, 81, 204–218.
- Johnstone, T., & Shanks, D. (1999). Two mechanisms in implicit artificial grammar learning? Comment on Meulemans and Van Der Linden (1997). *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 25, 524–531.
- Johnstone, T., & Shanks, D. (2001). Abstractionist and processing accounts of implicit learning. *Cognitive Psychology*, 42, 61–112.
- Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and order information in a composite holographic lexicon. *Psychological Review*, 114, 1–37. doi:10.1037/0033-295X.114.1.1
- Kinder, A. (2000). The knowledge acquired during artificial grammar learning: Testing the predictions of two connectionist models. *Psychological Research*, 63, 95–105.
- Kinder, A., & Lotz, A. (2009). Connectionist models of artificial grammar learning: What type of knowledge is acquired? *Psychological Research*, 73, 659–673.

- Knowlton, B. J., & Squire, L. R. (1996). Artificial grammar learning depends on implicit acquisition of both abstract and exemplar-specific information. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *22*, 169–181. doi:10.1037/0278-7393.22.1.169
- Ko, E., Soderstrom, M., & Morgan, J. (2009). Development of perceptual sensitivity to extrinsic vowel duration in infants learning American English. *Journal of the Acoustical Society of America*, *126*, EL134–9.
- Kuperman, V., Stadthagen-Gonzalez, H., & Brysbaert, M. (2012). Age-of-acquisition ratings for 30,000 English words. *Behavior Research Methods*, *44*, 978–990. doi:10.3758/s13428-012-0210-4
- Landauer, T. K., & Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, *104*, 211–240. doi:10.1037/0033-295X.104.2.211
- Lorch, R. F., Jr., & Myers, J. L. (1990). Regression analyses of repeated measures data in cognitive research. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *16*, 149–157. doi:10.1037/0278-7393.16.1.149
- MacLeod, C. M., Gopie, N., Hourihan, K. L., Neary, K. R., & Ozubko, J. D. (2010). The production effect: Delineation of a phenomenon. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *36*, 671–685. doi:10.1037/a0018785
- McAndrews, M. P., & Moscovitch, M. (1985). Rule-based and exemplar-based classification in artificial grammar learning. *Memory & Cognition*, *13*, 469–475.
- McCulloch, C. E., & Searle, S. R. (2001). *Generalized, linear, and mixed models*. New York, NY: Wiley-Interscience.
- Miller, G. A. (1958). Free recall of redundant strings of letters. *Journal of Experimental Psychology*, *56*, 433–491.
- Perruchet, P., & Pacteau, C. (1990). Synthetic grammar learning: Implicit rule abstraction or explicit fragmentary knowledge. *Journal of Experimental Psychology: General*, *119*, 264–275. doi:10.1037/0096-3445.119.3.264
- Reber, A. S. (1967). Implicit learning of artificial grammars. *Journal of Verbal Learning and Verbal Behavior*, *6*, 855–863. doi:10.1016/S0022-5371(67)80149-X
- Redington, M., & Chater, N. (1996). Transfer in artificial grammar learning: A reevaluation. *Journal of Experimental Psychology: General*, *125*, 123–138.
- Scott, R. B., & Dienes, Z. (2008). The conscious, the unconscious, and familiarity. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *34*, 1264–1288.
- Searle, S. R., Casella, G., & McCulloch, C. E. (1992). *Variance components*. New York, NY: Wiley.
- Van Heuven, W. J. B., Mandera, P., Keuleers, E., & Brysbaert, M. (2014). SUBTLEX-UK: A new and improved word frequency database for British English. *Quarterly Journal of Experimental Psychology*, *67*, 1176–1190. doi:10.1080/17470218.2013.850521
- Vokey, J. R., & Brooks, L. R. (1992). Salience of item knowledge in learning artificial grammars. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *18*, 328–344.
- Vokey, J. R., & Jamieson, R. K. (2014). A visual familiarity account of evidence for orthographic processing in baboons (*Papio papio*). *Psychological Science*, *25*, 991–996.