Using generalized additive (mixed) models to analyze single case designs☆

William R. Shadish a,⁎, Alain F. Zuur b, Kristynn J. Sullivan a

a School of Social Sciences, Humanities and Arts, University of California, Merced, USA
b Highland Statistics, Ltd., Newburgh, UK

This article shows how to apply generalized additive models and generalized additive mixed models to single-case design data. These models excel at detecting the functional form between two variables (often called trend), that is, whether trend exists, and if it does, what its shape is (e.g., linear and nonlinear). In many respects, however, these models are also an ideal vehicle for analyzing single-case designs because they can consider level, trend, variability, overlap, immediacy of effect, and phase consistency that single-case design researchers examine when interpreting a functional relation. We show how these models can be implemented in a wide variety of ways to test whether treatment is effective, whether cases differ from each other, whether treatment effects vary over cases, and whether trend varies over cases. We illustrate diagnostic statistics and graphs, and we discuss overdispersion of data in detail, with examples of quasibinomial models for overdispersed data, including how to compute dispersion and quasi-AIC fit indices in generalized additive models. We show how generalized additive mixed models can be used to estimate autoregressive models and random effects and discuss the limitations of the mixed models compared to generalized additive models. We provide extensive annotated syntax for doing all these analyses in the free computer program R.

© 2013 Society for the Study of School Psychology. Published by Elsevier Ltd. All rights reserved.

1. Introduction

Proposals for the statistical analysis of single-case designs (SCDs) have increased greatly in the last decade (Edgington & Onghena, 2007; Ferron & Levin, in press; Kratochwill & Levin, 2010; Maggin et al., 2011; Parker, Vannest, Davis, & Sauber, 2011; Shadish et al., in press). These include effect size estimators, ordinary regression, multilevel models, Bayesian statistics, and randomization tests. All of those analyses must deal with a key issue in all longitudinal data: systematic increases or decreases in the outcome over time in the absence of treatment, commonly referred to as trend. Most proposed SCD analyses either assume no trend or model trend as linear even when they could model nonlinear trend, despite the fact that cases exist in which nonlinear models fit the data better than linear models (Heimberg & Becker, 2002; Nishith, Resick, & Griffin, 2002). Failure to model trend correctly can bias both coefficients and their standard errors and also result in artifactual autocorrelation given that any model misspecification can affect the error structure of the model. To be clear, we are not saying that trend is a problem to be eliminated.
Trend, and changes in it, are substantively interesting in their own right. For example, changes in trend may be part of the treatment effect. The issue is modeling trend correctly. In that sense, we share the interests of prior researchers interested in addressing trend in both quantitative and visual analyses (Parker, Cryer, & Byrns, 2006; Parker, Vannest, & Davis, 2012).

The method we elaborate in this article, called generalized additive models (GAMs\(^1\)), should be of substantial use to all SCD researchers interested in the correct modeling of trend. GAMs have been applied to a wide array of longitudinal data (Brown, 1993; Chilcoat & Schütz, 1996; Dominici, McDermott, Zeger, & Sammet, 2002; Hemmi, Schneider, Mülller, Meyer, & Wilhelm, 2011; Zajacova & Burgard, 2012; Zeger & Diggle, 1994; Zhang, Xie, & Li, 2012) but never to SCDs. They have a key advantage over the usual parametric regression methods. Parametric methods require the researcher to decide the functional form of the trend that may be present in the data (e.g., linear, quadratic, and logistic). When the researcher knows that form, parametric methods are the best. Unfortunately, the researcher rarely knows that form with any confidence. GAMs help solve this problem.

In previous work (Sullivan, Shadish, & Steiner, in press), we introduced the use of GAMs for the analysis of SCDs, reported a computer simulation showing that they performed well compared to the usual generalized linear models (GLMs) when the functional form is unknown (as it nearly always is), showed how to apply some basic GAMs to SCDs on three example cases, and discussed some of the problems and prospects of GAMs. In this article, we extend that work by showing additional ways of analyzing SCDs with GAMs as well as with generalized additive mixed models (GAMMs). We provide extensive annotated syntax for doing all these analyses in the Appendix. While we focus initially on the modeling of trend as the unique feature of GAMs, in the Discussion section, we describe how these models also take into account level, variability, overlap, immediacy of effect, and phase consistency—all key issues that SCD researchers consider when they make judgments about detecting a functional relationship.

Unlike some other articles in this special section, we do not illustrate a meta-analysis using the results from GA(M)Ms. That has been a lower priority for us as we learn first how to apply GA(M)Ms to SCD data. In principle, GA(M)Ms could be used to do meta-analysis in two ways. One could use the methods of Swaminathan, Rogers, and Horner (2013–this issue) to standardize the raw effect by the total variance (within case plus between case). However, it is less clear that a standardized mean difference statistic is the appropriate effect size metric for SCD data that are counts or rates, where metrics like an odds ratio may be more appropriate. We prefer to study this issue more carefully before proposing an effect size metric for non-normal data in SCD research. Second, we could use the method of Moeyaert, Ferron, Beretvas, and Van Den Noortgate (2014–this issue), limiting the meta-analysis to cases that have identical outcomes over studies so that standardization is not needed and completing the meta-analysis with a three-level multilevel model.

However, both of these approaches would encounter problems. The Moeyaert et al. (2014–this issue) approach cannot include all the data when diverse outcomes are present. For the Swaminathan, Rogers, and Horner (2013–this issue) approach, obtaining the conditional variance from the Bayesian software would be challenging for the ordinary user, though it is possible. For the Moeyaert et al. (2014–this issue) approach, cognate analyses to those of Shadish, Hedges, and Pustejovsky (this issue) could be developed but are not yet available. One could, for example, use standard regression diagnostics (Fox & Weisberg, 2011) to check model assumptions and complete tests for predictors using ordinary multilevel model methods. However, it is not clear how one would complete publication bias tests that were demonstrated in Shadish, Hedges, and Pustejovsky (this issue), engage in cumulative meta-analysis, or create forest plots. Clearly the use of any of these models, including GA(M)Ms, to facilitate meta-analysis needs more attention.

1. Generalized additive models

The reader who is mostly interested in applications of GA(M)Ms can skim this section, reading in more detail only the parts that seem intuitive (like the first three or four paragraphs), or skip this section and go directly to section 2. The reader who wants even more detail than we provide here can refer to a number of excellent works (Hastie & Tibshirani, 1986; Hothorn & Everitt, 2010; Keele, 2008; Ruppert, Wand, & Carroll, 2003; Sullivan et al., in press; Wood, 2006; Zuur, 2012).

1.2. A conceptual and statistical introduction to GAMs

GAMs expand GLMs by replacing one or more GLM terms with a smoothing function. Many researchers are already familiar with smoothing functions in the form of loess (or its predecessor lowess) smoothers (Keele, 2008), which are widely available in many standard computer programs. Consider Fig. 1, for example (R syntax to construct that figure is in the Appendix). The graph at upper left shows the raw data for Case A1 from Lambert et al. (2006). The remaining three graphs fit three different lines to the data. In the graph at upper right, a straight line connects each dot. That is hardly a model of the data, however. Models should simplify data with an analysis that one hopes captures the key features of the data without simply connecting the dots. The lower

---

\(^1\) When talking about the statistical models in this paper, we will refer to generalized additive models as GAMs, and generalized additive mixed models as GAMMs. We use GA(M)M when we wish to refer to both of those statistical models simultaneously. When talking about the names of the computer programs used to estimate that model, we will refer to gam or gamm in different font.
left graph shows a linear regression line fit to each phase. It does simplify the representation of the data, but perhaps too much because it clearly does not match up to the dots well.

The lower right graph is the loess smoother. This approach computes linear or simple polynomial regressions in "neighborhoods" of sessions closest to each other; the size of the neighborhood is called a bandwidth that the researcher can specify (Keele, 2008). For example, if the bandwidth is 8, one regression would be on sessions 1 to 8, another on 2 to 9, and so forth. If the bandwidth is as long as the total length from the first to the last session (e.g., 30), the resulting regression line will be at its smoothest. Conversely, if the bandwidth is only 1, the result will be like the figure in the upper right of Fig. 1. In any case, the regressions are then connected together to interpolate curves in between the dots. The resulting loess smoother seems visually to represent the dots better than the simple linear regression.

GAMs produce lines that are visually similar to those from the loess smoother, but they use different methods for producing the smoothers. The smoothing functions used in GAMs are called splines. The word spline describes a flexible strip that can be fixed at certain points (knots) and then bent in a smooth curve around those points. They are used to draw curves in drafting or carpentry. Statistics adopted that word for many of its smoothers. Statistical splines improve on loess and related smoothers by having a stronger analytic basis, being better at preventing oversmoothing, having superior software implementations, and being easier to make part of GA(M)Ms.

Splines are complex to define in statistical terms. Keele (2008) provides the most accessible presentation for the interested reader, and this discussion of splines is largely drawn from that book. He gives the example of a data set that first decreases linearly and then increases linearly. The point at which it changes direction (bends) is called a knot (c). To oversimplify, splines fit separate regression lines in the regions between knots and then join them together continuously. We can express this very simple case joining two linear regressions together into a spline as

\[
y = \begin{cases} 
\alpha + \beta_1 x & \text{if } x \leq c \\
\alpha + \beta_1 x + \beta_2 (x-c) & \text{if } x > c
\end{cases}
\]

Then the predicted values (y) will change depending on whether the observations (x) are above or below the knot.

Estimating this model requires a basis function. Basis functions add predictors to model the regions between the knots. Continuing with Keele’s (2008) example, imagine the dataset contains \( n = 100 \) observations, and the knot is placed at \( c = 60 \). If so, then the basis function for the left side (where the line decreases) is

\[
B_L(x) = \begin{cases} 
c - x & \text{if } x < c \\
0 & \text{otherwise}
\end{cases}
\]

The basis function for the right side (where the line then starts to increase linearly) is

\[
B_R(x) = \begin{cases} 
x - c & \text{if } x > c \\
0 & \text{otherwise}
\end{cases}
\]
These bases are then added to the regression model design matrix. In ordinary linear regression, that matrix would have two columns, the first a column of ones to represent the intercept, and the second a column of session numbers (x’s). To model the spline in regression, we replace the column of session numbers with two columns representing the two basis functions:

\[
X = \begin{bmatrix}
1 & (60-x_1) & 0 \\
& \vdots & \vdots \\
1 & (60-x_{50}) & 0 \\
1 & 0 & 0 \\
1 & 0 & (x_{51}-60) \\
& \vdots & \vdots \\
1 & 0 & (x_{100}-60)
\end{bmatrix}.
\]

This matrix is then used to create the predicted values fit to the spline as

\[
\hat{y} = [X'X]^{-1}X'y.
\]

The preceding description of splines and smoothing is oversimplified for three reasons. First, bases are rarely linear, as in the model above; rather, bases are more often quadratic or cubic to allow curves. Second, many kinds of splines exist, for example, cubic, natural, and B-splines, and each has its own statistical representation. Keele (2008) outlines their advantages and disadvantages. Third, to avoid overfitting a curve to the data, the estimation of splines typically involves penalizing the model for having too many parameters. The penalty is like the difference between \(R^2\) and adjusted \(R^2\) in multiple regression. The adjusted \(R^2\) penalizes model fit for each additional parameter in the regression.

Sullivan et al. (in press) used cubic regression splines in the GAMs in their article. They are a series of separate cubic polynomial curves joined together at knots. In this article, we mostly use thin plate splines. The main difference is that cubic splines place the smoothing penalty on the second derivative of the smoothed fit; thin plate splines can penalize any derivative of the smoothed fit, so they are more general. Thin plate splines also have better numerical characteristics (e.g., computing speed and optimization).

Regarding knots, the researcher can place them manually based on inspection of the data. However, evenly spacing knots (the computer program default) usually captures the shape of any nonlinearity quite well. GAM results are generally insensitive to the number and location of the knots unless they are distributed in a very nonuniform way such as at the 25th, 75th, 90th, 95th, and 99th percentiles (Durrleman & Simon, 1989). The number of knots is not a driving factor in the amount of smoothing. Hence, software defaults to a large number of knots and uses other tools to control the amount of smoothing.

GAMs are estimated by iteratively reweighted least squares (Wood, 2006). To avoid overfitting curves to the data (fitting chance fluctuations as if they were real nonlinearities), the estimation routine we use in this article adds a penalty for too much wiggliness in the smoothing function.\(^2\) The optimal smoothing function is then chosen (internally by the computer program) using a generalized cross-validation (GCV) score or an Unbiased Risk Estimator (UBRE) score, both of which measure how well models minimize predictive error, where a smaller score is a better fit.

GAMs measure nonlinearity using the effective degrees of freedom (edf) of the smoothing term, roughly equivalent to the polynomial order of the smoother plus one (Hothorn & Everitt, 2010, Chapter 10; Ruppert et al., 2003). However, when edf is less than 3, this rule of thumb becomes very approximate. Fig. 2 illustrates edf with simulated data. An edf = 1 is a linear effect (Row one of Fig. 2). Row two of Fig. 2 shows a quadratic term; it has just over three edf. Row three of Fig. 2 shows very wiggly than 3, this rule of thumb becomes very approximate.

GAM results are generally insensitive to the number and location of the knots unless they are distributed in a very nonuniform way such as at the 25th, 75th, 90th, 95th, and 99th percentiles (Durrleman & Simon, 1989). The number of knots is not a driving factor in the amount of smoothing. Hence, software defaults to a large number of knots and uses other tools to control the amount of smoothing.

1.3. GAM equations

Statistically, we write a GAM by replacing at least some of the usual parametric regression terms with a smoothed term. For normally distributed data, an example is

\[
Y_j = x_j\theta + s_1(x_{j1}) + s_2(x_{j2}) + \ldots s_p(x_{jp}) + \varepsilon_i
\]

where \(Y_j\) is the outcome at time \(j\), \(x_j\) is a vector with multiple columns containing parametric covariates (for the present examples, typically the intercept and treatment variables), \(\theta\) is the corresponding parameter vector of regression coefficients, \(s_1(x_{j1}), \ldots s_p(x_{jp})\)

\(^2\) The amount of smoothing can be controlled many ways, by changing span size in moving average and loess smoothers, changing the amount of knots in linear/quadratic/cubic spine regression, putting conditions on the betas of the smoother (penalized splines), ensuring that first-order and second-order derivatives at the knots are the same (cubic regression spline), or penalizing wiggliness with an integral (smoothing) spline (Zuur, 2012).
are smoothing functions (the s’s) for each predictor (the x’s) to discover the functional form, and \( \epsilon_j \) is the random error, independently and identically distributed as \( N(0, \sigma^2) \). For SCDs, the \( x \) vector would typically include an intercept and the treatment variable, and the trend or interaction terms might be candidates for smoothing. Additional substantive covariates might or might not be smoothed (e.g., child’s age or IQ score).

The above equation assumes normality. In the social sciences, SCD data are rarely normally distributed (Shadish & Sullivan, 2011), so the model changes for other distributions. For Poisson distributed counts, the left side of the equation is replaced with a log link function where:

\[
Y_j \sim \text{Poisson}(\mu_j) \\
E(Y_j) = \mu_j \\
\log(\mu_j) = \mathbf{x}_j \beta + s_1(x_{1j}) + s_2(x_{2j}) + \ldots + s_p(x_{pj})
\]

These three equations are the distribution of the response variable, the link between the expectation (mean) of the response variable and the covariates, and the predictor function, respectively.

For the Lambert et al. (2006) data, we use a binomial distribution. Those authors gathered their data using partial interval recording (Ayres & Gast, 2010). A 10-second observation interval was followed by a 5-second recording interval. If the student engaged in disruptive behavior at any time during the 10-second interval, the interval had a disruptive (1) score. Otherwise, it was scored as not disruptive (0). These observations continued for total of 10 intervals per session. The binomial distribution is
appropriate when the outcome is a frequency of a binary (e.g., success or failure) event out of a known number of trials, sometimes reported as percentages or proportions \( \hat{P}_j \). The default link function for the binomial distribution is a logit:

\[
Y_j \sim \text{Bin}(\pi_j)
\]

\[
E(Y_j) = \pi_j \text{ and } \text{var}(Y_j) = (1-\pi_j) \times \pi_j
\]

\[
\logit(\pi_j) = \ln \left( \frac{\pi_j}{1-\pi_j} \right) = \mathbf{x}_j^T \mathbf{\beta} + s_1(x_{1j}) + s_2(x_{2j}) + \ldots s_p(x_{pj})
\]

where \( \pi_j \) is the probability of success.

As Shadish, Kyse, and Rindskopf (2013) said, other distributional assumptions for different kinds of SCD outcomes include Bernoulli for outcomes that are only binary (1, 0) within a session (e.g., only one trial per session) and multinomial for outcomes that are either ordered or nominal categories—though the kind of data requiring these models is rare in SCD research. The Bernoulli sampling model applies when each case has only a binary (1/0) response during each session, and the link function is a logit as in Eq. (1). We have not seen such an outcome used in SCD research in surveys of the literature (Smith, 2012). We also have not seen the use of a nominal category as an outcome in SCD research, for example, whether a case falls into one of several diagnostic categories over time. If such an outcome were to occur, it could be analyzed with a multinomial logistic regression. We have seen the use of ordered categories as outcomes. Jostad, Miltenberger, Kelso, and Knudson (2008), for example, trained children with an intervention designed to prevent them playing with firearms. The outcome was four ordered categories where 0 = touched the firearm; 1 = did not touch the firearm; 2 = did not touch and left the area within 10 s; and 3 = did not touch, left the area, and told an adult. The model is multinomial with a cumulative logit link function. Raudenbush and Bryk (2002, Chapter 10) provide details of the model, Raudenbush, Bryk, Cheong, Congdon, and du Toit (2004) show how the models can be estimated in the hierarchical linear model computer program HLM, and Raudenbush and Ferron (in press) show an application to SCDs.

Another set of distributions is a variation on the basic binomial distribution to accommodate overdispersion in the data. Overdispersion occurs when the variance of the observations exceeds what is predicted from statistical theory, and we discuss it in detail later. In this article, we show a quasi-binomial model; Shadish, Kyse, and Rindskopf (2013) also showed a quasi-Poisson model. Essentially, these models adjust standard errors to take overdispersion into account. A different approach to overdispersion is to find a distribution in which the data match the assumptions of the distribution. Examples are beta-binomial and negative binomial distributions. Similarly, when more zero observations are present than the distribution predicts, zero-inflated models are also an option. These are advanced topics, and as we will see later in this article, they may need to be estimated using more advanced statistical methods often used in the Bayesian traditions.

### 1.4. GAM computations

In this article, we employ the free computer program R (R Development Core Team, 2012). An excellent introduction to working in the R environment is Fox and Weisberg (2011; see also Zuur, Ieno, & Meesters, 2009). We do the GAM analyses mostly in the R mgcv package (Wood, 2006, 2010),\(^3\) using its \texttt{gam} command. Syntax for all analyses is in the Appendix, but we only report a subset of the best fitting results from the syntax in the article given space limitations. In our previous work (Sullivan et al., in press), we used cubic regression splines and separate smooth terms for trend and the interaction of trend and treatment, for example:

\[
Y_j = \beta_0 + s_1(x_{1j}) + \beta_2 x_{2j} + s_3(\text{Int}_{ij}) + \epsilon_j
\]

where \( Y_j \) is the outcome variable, \( \beta_0 \) is the intercept (the outcome level at the start of baseline), \( x_{1j} \) is the sequence number (usually time or session number), \( x_{2j} \) is the treatment dummy (0 for baseline, 1 for treatment; see Shadish, Kyse, & Rindskopf, 2013, for other ways to code treatment), \( \epsilon_j \) is a random error term, and \( \text{Int}_{ij} \) is an interaction term:

\[
\text{Int}_{ij} = \left[ x_j - t_j \right] \times x_{2j}
\]

where \( t_j \) is the time or session number for the first data point in the first treatment phase. However, this approach can lead to collinearity between the trend and interaction terms, and part of the \( z = 0 \) data will determine the shape of the \( z = 1 \) data, which may not really reflect the interaction of interest.

In this article, we use thin plate regression splines and a different parameterization of the interaction. We also show a different way of comparing models, how to estimate GAMs for each case simultaneously rather than one at a time, how to estimate the collinearity between the trend and interaction terms, and part of the \( z = 0 \) data will determine the shape of the \( z = 1 \) data, which may not really reflect the interaction of interest.

\(^3\) Other R programs that can estimate GAMs are \texttt{gamlss} (Stasinopoulos & Rigby, 2007; Stasinopoulos, Rigby, & Akantziliotou, 2008). See Zuur (2012) for discussion of their relative strengths and weaknesses. GAMs can also be estimated in SAS and in Stata.
We read the data into R as a tab delimited text file, which can usually be obtained from most statistical packages or spreadsheet files using some version of a "Save As" option. Fig. 3 is a snapshot of the data set. In this case, the original raw data file contains the following variables: Case is the case identification number, x is the session number, y is the outcome variable, z indicates baseline (0) or treatment (1), trial is the number of intervals observed each session, and ... indicates data cut from the data for this figure but actually present in the full data set.

### 2. Application to Lambert et al. (2006)

We read the data into R as a tab delimited text file, which can usually be obtained from most statistical packages or spreadsheet files using some version of a "Save As" option. Fig. 3 is a snapshot of the data set. In this case, the original raw data file contains the following variables: Case is the case identification number, x is the session number, y is the outcome variable, z indicates baseline (0) or treatment (1), trial is the number of intervals observed each session, and ... indicates data cut from the data for this figure but actually present in the full data set.

---

4 See Fox and Weisberg (2011) for other kinds of files to read, such as csv files, that can be useful with large numbers of variables.
contains 264 rows (one for each time point over all cases), and columns labeled Case (a case identification number from 1 to 9), x (session), y (outcome), z (0 = baseline, 1 = treatment), and trial (the number of trials on which the percent was based for the outcome variable, in this example 10). We also create an index to indicate in which of the two classrooms the case was nested and a variable called successes computed as Trials minus y (i.e., the number of trials during a session in which the child was not disruptive), which will be needed in the binomial model (see Appendix). Unlike Sullivan et al. (in press), we treat z as a categorical variable (a factor in R terms; see appendix model M0), and we treat Case and Class as factors because they are categorical variables.

When we model trend in GAMs simultaneously for all cases at once, it can be useful (but is not necessary) for all of the data series included in the design to have the same overall length. The same is true in other linear models as well. The issue is one of interpretation. If you apply these methods with one trend on data where some series stop earlier, then the estimated trend is partly based on all the cases until some cases run out of observations. After that point, the estimated trend is only based on the fewer remaining cases. However, phases do not need to be of equal length. For instance, if all time series are 26 observations in length, those observations may be distributed unequally over phases, and those unequal distributions do not have to be the same across cases. Also, all cases do not have to be observed at each time point.

Lambert et al. (2006) does not meet the condition of having all cases be of equal overall length for two reasons. One is illustrated by Case A3 where data collection ends earlier for A3 than the other students in Class A. The second is that Class A ended data collection at time 31, but Class B ended at session 34. For simplicity, we simply delete time points after the 26th observation from all cases. The deleted data points are all from the second treatment phase. A binomial mixed model with Case as a random effect finds that the dropped data do not differ significantly from the data points that remained in the two treatment phases (z = −1.307, p = .191), so this deletion probably did not much change the general results. This deletion reduces the size of the dataset from 264 to 234 observations, or by about 11.4%.

The topic of missing data has not been much examined in SCD research, so giving well-grounded advice is difficult. An alternative to deleting time points after the 26th point would have been to delete observations randomly from longer datasets, probably preferable but a bit more tedious. If the data were planned to be the same length, but some data are missing for unplanned reasons, such as a child missing a session due to illness, an option is to impute missing data (Smith et al., 2012). If data were missing for planned reasons, as the apparently planned decision that Class A ended data collection at time 31 but Class B ended at session 34, an option is to analyze those classes separately or to do a sensitivity analysis to see if results combining A and B differ from results analyzing them separately. Dealing with missing data is a topic needing more research to identify best practice.

We use a binomial model, where $Y_{ij} =$ number of disruptive behaviors at observation $j$ ($j = 1,.., n_i$, where $n_i$ is the sometimes differing length of the time series for each case) for person $i$ ($i = 1,...,9$), and where the first line of (1) is

$$Y_{ij} \sim \text{Bin}(\pi_{ij}, N_{ij})$$

with $N_{ij} =$ number of trials per session for person $i$. $N_{ij}$ is always 10 in Lambert et al. (2006) but may vary in other studies. The next question is what covariates to put into the third line of (1), that is, which of our variables predicts expected disruptive behavior and its changes. In this article, we approach this as a model building question in which we explore which of many models is best fitting using Akaike’s Information Criterion (AIC)\(^5\):

$$\text{AIC} = -2LL + (2 \times K)$$

where $LL$ is the model Log Likelihood, and $K$ is the number of parameters estimated in the model. In general, model fit indices are useful indicators of how well a model predicts the data. However, because one can increase fit to the data by adding more and more predictors—and get a perfect fit with as many predictors as observations—penalizing the fit index for an increasing number of parameters is useful. That is what AIC does. Everitt (1998, p. 7) defines AIC as follows: “The AIC takes into account both the statistical goodness of fit and the number of parameters that have to be estimated to achieve this particular degree of fit, by imposing a penalty for increasing the number of parameters. Lower values of the index indicate the preferred model, that is, the one with the fewest parameters that still provides an adequate fit to the data.” Some capitalization on chance may occur in model building, but it is appropriate for the early stages of research when we do not have strong theory about the best predictors. Also, model building will help illustrate how to use R and mgcv in more detail.

2.1. Model building

Table 1 summarizes the models we test in this article. Models are labeled M0 to M11 in order in the left column of the table, and those labels match the appendix labels. In the right column are the equations for each model. First consider four models that predict the outcome from (1) session (x), treatment (z) and Case (called M0 in the Appendix), (2) changing M0 by adding a thin plate spline

\(^5\) One could also use UBRE or GCV. GCV is the Gaussian version of the UBRE. AIC, UBRE, and GCV tend to be very similar and are closely linked mathematically.
the actual collinear. If so, M1 is best fitting among these four models. Rather than table the results, we present them in text so the reader can see the AICs for all models in this article.) However, M2 fails to estimate some coefficients in the model; it may be that Class and Case are predictor from M2 (M3). AIC for models M0 to M3 are 1076.95, 1041.52, 1041.52, and 1075.28, respectively. (Table 2 summarizes smooth to x in order to see if trend requires smoothing (M1), (3) adding a predictor for Class to M1 (M2), and (4) deleting the Case predictor from M2 (M3). AIC for models M0 to M3 are 1076.95, 1041.52, 1041.52, and 1075.28, respectively. (Table 2 summarizes the AICs for all models in this article.) However, M2 fails to estimate some coefficients in the model; it may be that Class and Case are collinear. If so, M1 is best fitting among these four models. Rather than table the results, we present them in text so the reader can see the actual mgcv output (in a distinct font to indicate this is a copy of the output):

<table>
<thead>
<tr>
<th>Model label</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>M0</td>
<td>logit((\tau_i)) = (\alpha + \beta_1 x_i + \beta_2 z_i + \beta_3 x_i z_i + \beta_4 x_i z_i^2 + \beta_5 x_i z_i^3 + \epsilon_i)</td>
</tr>
<tr>
<td>M1</td>
<td>logit((\tau_i)) = logit((\tau_i)) + (\beta_1 x_i + \beta_2 z_i + \beta_3 x_i z_i + \epsilon_i)</td>
</tr>
<tr>
<td>M2</td>
<td>logit((\tau_i)) = logit((\tau_i)) + (\beta_1 x_i + \beta_2 x_i z_i + \beta_3 x_i z_i^2 + \epsilon_i)</td>
</tr>
<tr>
<td>M3</td>
<td>logit((\tau_i)) = logit((\tau_i)) + (\beta_1 x_i + \beta_2 x_i z_i + \epsilon_i)</td>
</tr>
<tr>
<td>M4</td>
<td>logit((\tau_i)) = logit((\tau_i)) + (\beta_1 x_i + \beta_2 z_i + \epsilon_i)</td>
</tr>
<tr>
<td>M5</td>
<td>logit((\tau_i)) = logit((\tau_i)) + (\beta_1 x_i + \beta_2 x_i z_i + \beta_3 x_i z_i^2 + \epsilon_i)</td>
</tr>
<tr>
<td>M6</td>
<td>logit((\tau_i)) = logit((\tau_i)) + (\beta_1 x_i + \beta_2 x_i z_i + \beta_3 x_i z_i^2 + \beta_4 x_i z_i^3 + \epsilon_i)</td>
</tr>
<tr>
<td>M7</td>
<td>logit((\tau_i)) = logit((\tau_i)) + (\beta_1 x_i + \beta_2 x_i z_i + \beta_3 x_i z_i^2 + \epsilon_i)</td>
</tr>
<tr>
<td>M8</td>
<td>logit((\tau_i)) = logit((\tau_i)) + (\beta_1 x_i + \beta_2 x_i z_i + \beta_3 x_i z_i^2 + \beta_4 x_i z_i^3 + \epsilon_i)</td>
</tr>
</tbody>
</table>

Generalized additive mixed models

<table>
<thead>
<tr>
<th>Model label</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>M9</td>
<td>logit((\tau_i)) = (\alpha + f(x_i); \beta_2 x_i + \beta_2 x_i z_i + \beta_3 x_i z_i^2 + \epsilon_i), where (\epsilon_i = \phi \times e_i - z_i + \psi_i), (\psi_i = \text{N}(0, \sigma^2))</td>
</tr>
<tr>
<td>M10</td>
<td>logit((\tau_i)) = (\alpha + f(x_i); \beta_2 x_i + \beta_2 x_i z_i + \beta_3 x_i z_i^2 + \beta_4 x_i z_i^3 + \epsilon_i), where (\epsilon_i = \phi \times e_i - z_i + \psi_i), (\psi_i = \text{N}(0, \sigma^2))</td>
</tr>
<tr>
<td>M11</td>
<td>logit((\tau_i)) = (\alpha + f(x_i); \beta_2 x_i + \beta_2 x_i z_i + \beta_3 x_i z_i^2 + \epsilon_i), where (\epsilon_i = \phi \times e_i - z_i + \psi_i), (\psi_i = \text{N}(0, \sigma^2))</td>
</tr>
</tbody>
</table>

Note. The model label corresponds to the notation in the Appendix and text; \(x_i\) is the session; \(z_i\) is the treatment dummy variable; Case is a categorical variable assigning numbers 1 to 9 to each of the 9 cases respectively; Class is a categorical variable assigning numbers 1 or 2 to the two classes, respectively; Case.Z is a variable that assigns separate identification numbers to each phase within each case; \(\epsilon_i\) is random error; \(\phi\) is a first-order autoregressive parameter; \(\psi_i\) is the variance component (random effect) for Case; the colon (:) between two terms indicates an interaction between those terms.

smoothing to \(x\) in order to see if trend requires smoothing (M1), (3) adding a predictor for Class to M1 (M2), and (4) deleting the Case predictor from M2 (M3). AIC for models M0 to M3 are 1076.95, 1041.52, 1041.52, and 1075.28, respectively. (Table 2 summarizes the AICs for all models in this article.) However, M2 fails to estimate some coefficients in the model; it may be that Class and Case are collinear. If so, M1 is best fitting among these four models. Rather than table the results, we present them in text so the reader can see the actual mgcv output (in a distinct font to indicate this is a copy of the output):

| Parametric coefficients: | Estimate | Std. Error | z value | Pr(>|z|) |
|---------------------------|----------|------------|---------|----------|
| (Intercept)               | 0.936    | 0.166      | 5.643   | 0.000    |
| factor(2) 1               | -2.426   | 0.158      | -15.353 | 0.000    |
| factor(2) 2               | 0.277    | 0.222      | 1.248   | 0.212    |
| factor(2) 3               | 0.221    | 0.216      | 1.024   | 0.306    |
| factor(2) 4               | 0.233    | 0.219      | 1.065   | 0.287    |
| factor(2) 5               | 0.244    | 0.221      | 1.103   | 0.270    |
| factor(2) 6               | -0.869   | 0.219      | -3.970  | 0.000    |
| factor(2) 7               | -0.477   | 0.219      | -2.181  | 0.029    |
| factor(2) 8               | -0.959   | 0.216      | -4.438  | 0.000    |
| factor(2) 9               | -0.954   | 0.218      | -4.374  | 0.000    |

Approximate significance of smooth terms: | edf | Ref.edf | Chi.sq | p-value |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>s(x)</td>
<td>8.557</td>
<td>8.944</td>
<td>43.66</td>
<td>0.000</td>
</tr>
<tr>
<td>R-sq.(adj)</td>
<td>0.654</td>
<td>Deviance explained = 58.9%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UBRE score</td>
<td>1.759</td>
<td>Scale est. = 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note the several features of this output. First, compared to Sullivan et al. (in press), this model incorporates all cases in one model, allowing statistical tests about how cases differ. The absence of Case 1 is due to its use as a reference category to which other cases are compared. The logit of the outcomes for Cases 6 through 9 are significantly lower than Case 1, but the other cases do not differ significantly from Case 1. Second, the treatment significantly decreases the logit of disruptive behavior on average, and the logit of disruptive behavior differs significantly from zero at the start of the study. Third, a model with a smoothed session term fits significantly better than a model without it, with edf = 8.557 indicating a highly nonlinear trend. Finally, the adjusted R-square (higher is better), deviance explained (akin to variance explained where higher is better) and UBRE score (lower is better) are all fit statistics that could be used to compare models, though we use AIC, as noted previously. The scale estimate in the output is something we discuss in more detail later in the article when we discuss overdispersion. Also, we do not show how to interpret the individual coefficients in this or other models until after we have selected our preferred model in the Discussion section of this article.

---

6. Here and throughout, some extraneous mgcv output is deleted, and results are rounded to three decimals.
Now we model various interactions, all of which are extensions of M1 because it was best-fitting: (1) a model in which session interacts with treatment to understand if treatment changes the slope of observations (M4); (2) a model with a case by session interaction replacing the interaction in M4, so that each case can have its own trend (M5); and (3) a model with a classroom by session interaction replacing the interaction in M5 so each classroom can have its own trend (M6). The interaction is modeled by “s(x, by = factor(z))”, which estimates the smoothed trend separately for baseline and treatment, or separately for case (M5) or class (M6), as the case may be. The AICs for these models are 1028.50 (M4), 870.79 (M5), and 992.82 (M6). M4 fit relatively poorly. Results for the best-fitting M5 are as follows:

| Parametric coefficients: | Estimate | Std. Error | z value | Pr(>|z|) |
|--------------------------|----------|------------|---------|----------|
| (Intercept)              | 1.213    | 0.196      | 6.180   | 0.000    |
| factor(z)1               | −3.143   | 0.230      | −13.695 | 0.000    |
| factor(Case)2            | 0.311    | 0.237      | 1.309   | 0.191    |
| factor(Case)3            | 0.104    | 0.263      | 0.396   | 0.692    |
| factor(Case)4            | 0.212    | 0.270      | 0.785   | 0.432    |
| factor(Case)5            | 0.335    | 0.253      | 1.324   | 0.186    |
| factor(Case)6            | −1.039   | 0.255      | −4.078  | 0.000    |
| factor(Case)7            | −0.455   | 0.254      | −1.793  | 0.073    |
| factor(Case)8            | −0.865   | 0.242      | −3.573  | 0.000    |
| factor(Case)9            | −0.984   | 0.258      | −3.808  | 0.000    |

Approximate significance of smooth terms:

<table>
<thead>
<tr>
<th>edf</th>
<th>Ref.df</th>
<th>Chi.sq</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.630</td>
<td>8.512</td>
<td>26.309</td>
<td>0.001</td>
</tr>
<tr>
<td>1.000</td>
<td>1.000</td>
<td>4.677</td>
<td>0.031</td>
</tr>
<tr>
<td>7.037</td>
<td>8.103</td>
<td>38.957</td>
<td>0.000</td>
</tr>
<tr>
<td>8.486</td>
<td>8.921</td>
<td>41.657</td>
<td>0.000</td>
</tr>
<tr>
<td>8.506</td>
<td>8.927</td>
<td>43.738</td>
<td>0.000</td>
</tr>
<tr>
<td>1.000</td>
<td>1.000</td>
<td>1.728</td>
<td>0.189</td>
</tr>
<tr>
<td>5.285</td>
<td>6.405</td>
<td>18.386</td>
<td>0.007</td>
</tr>
<tr>
<td>8.462</td>
<td>8.922</td>
<td>47.272</td>
<td>0.000</td>
</tr>
<tr>
<td>8.574</td>
<td>8.941</td>
<td>42.320</td>
<td>0.000</td>
</tr>
</tbody>
</table>

UBRE score = 1.029

Note. AICs for gam models M0 to M8 are not comparable to AICs for gamm and gamm4 models M9 to M12 and so should not be compared.
1 Model did not converge.

The main difference between M5 and M1 is that M5 shows individual case smooth terms, nearly all of which are significant. Case 6 has no significant trend, and Case 1 has a significant linear trend. The remaining cases show significant nonlinear trends ranging from edf = 5.29 (Case 7) to edf = 8.57 (Case 9).

We now create a new variable (Case_Z in the Appendix) that combines information from Case and z to allow for a different trend during each phase for each case (M7). For instance, where M5 only allowed Case 1 to have a different overall trend from the other cases, M7 allows Case 1 to have a different trend in the first phase from the other cases, and the same for each of the other three phases. The AIC for that model is 819.67. However, M7 requires estimating trend within each phase within each case. Since those within-case phases are short, we are skeptical that they contain sufficient data to estimate nonlinearities within phase very well. Finally, we could change M5 by allowing Case and treatment (z) to interact...
(M8) instead of having separate main effects for Case and treatment (z). This models a different treatment effect for each case, and yields AIC = 844.71. So both M7 and M8 are candidates for best fitting models. Both are interesting and yield consistent results concerning the overall treatment effect. However, we will work with M8, both because of the skepticism above and because we are particularly interested in variability of treatment effects over cases. Here is output for M8 using \texttt{anova(M8)}:

\begin{table}
\centering
\begin{tabular}{lrrr}
\hline
& df & Chi.sq & p-value \\
\text{factor(z)} & 1 & 13.49 & 0.000 \\
\text{factor(Case)} & 8 & 57.67 & 0.000 \\
\text{factor(z):factor(Case)} & 8 & 39.63 & 0.000 \\
\hline
\end{tabular}
\end{table}

\textbf{Approximate significance of smooth terms:}

\begin{table}
\centering
\begin{tabular}{lrrrr}
\hline
s(x):factor(Case)1 & edf & Ref.df & Chi.sq & p-value \\
7.375 & 8.345 & 22.449 & 0.005 \\
1.000 & 1.000 & 5.959 & 0.015 \\
8.106 & 8.766 & 39.676 & 0.000 \\
8.514 & 8.934 & 35.251 & 0.000 \\
1.000 & 1.000 & 0.212 & 0.646 \\
5.062 & 6.168 & 16.871 & 0.011 \\
7.891 & 8.680 & 34.438 & 0.000 \\
8.488 & 8.924 & 36.840 & 0.000 \\
\hline
\end{tabular}
\end{table}

These results show a significant treatment effect, a significant effect for case, and a significant interaction suggesting that treatment is differentially effective over cases. Results also show that smoothing trend separately for each case yields very different trends for each case, with one case having significant linear trend (Case 2 edf = 1.00), one having nonsignificant linear trend (Case 6 edf = 100, p = .65), and most cases having significant nonlinear trend nearly the same as in M5.

Further insight into M8 comes from \texttt{summary(M8)}:

\begin{table}
\centering
\begin{tabular}{lrrrr}
\hline
& Estimate & Std. Error & z value & Pr(>|z|) \\
\text{(Intercept)} & 1.288 & 0.379 & 3.396 & 0.001 \\
\text{factor(z)}1 & -3.347 & 0.911 & -3.673 & 0.000 \\
\text{factor(Case)}2 & 0.420 & 0.448 & 0.937 & 0.349 \\
\text{factor(Case)}3 & 1.257 & 0.738 & 1.704 & 0.088 \\
\text{factor(Case)}4 & -1.031 & 0.589 & -1.752 & 0.080 \\
\text{factor(Case)}5 & -0.194 & 0.508 & -0.382 & 0.703 \\
\text{factor(Case)}6 & -0.939 & 0.434 & -2.163 & 0.031 \\
\text{factor(Case)}7 & -0.481 & 0.506 & -0.951 & 0.342 \\
\text{factor(Case)}8 & -2.207 & 0.505 & -4.367 & 0.000 \\
\text{factor(Case)}9 & -0.851 & 0.580 & -1.467 & 0.142 \\
\text{factor(z):factor(Case)2} & -0.233 & 0.997 & -0.234 & 0.815 \\
\text{factor(z):factor(Case)3} & -2.375 & 1.521 & -1.562 & 0.118 \\
\text{factor(z):factor(Case)4} & 2.871 & 1.299 & 2.210 & 0.0271 \\
\text{factor(z):factor(Case)5} & 1.631 & 1.281 & 1.273 & 0.2030 \\
\text{factor(z):factor(Case)6} & -1.017 & 1.197 & -0.850 & 0.3952 \\
\text{factor(z):factor(Case)7} & 0.082 & 1.148 & 0.071 & 0.9432 \\
\text{factor(z):factor(Case)8} & 3.970 & 1.228 & 3.233 & 0.0012 \\
\text{factor(z):factor(Case)9} & -0.510 & 1.640 & -0.311 & 0.7557 \\
\hline
\end{tabular}
\end{table}

\textbf{R-sq.(adj)} = 0.798

M8 shows a significant intercept and treatment effect, that Cases 6 and 8 differ significantly from the reference Case 1 in their mean logit disruptive behavior, and that Cases 4 and 8 differ significantly in their response to treatment compared to the other case. Just as AIC(M8) > AIC(M5), so too all the other fit statistics show M8 improves on M5.

\textbf{Fig. 4} is a multipanel scatterplot (called a lattice plot in R) that graphs the raw data (black dots), a smoothed line depicting the predicted values from M8 (the solid line), and 95% confidence intervals around that line (the dashed line). The graphs are very consistent with the results reported above for M8. For example, results were that Case 6 displayed no significant trend at all, and that is what we see in \textbf{Fig. 4}—horizontal lines within phase. Case 2 has a significant linear trend, and again we see straight lines with slight increase over time. The remaining cases all show significant nonlinearity over time. Case 9 had the highest nonlinearity, which is reflected in part in the larger number of inflections in the curve for that case.
2.2. Diagnostics

Just as in ordinary regression, diagnostics exist to help understand the adequacy of a model. Here we illustrate some of them using the results from M8. One is to plot the fitted values against the residuals (Fig. 5). The dots should be distributed randomly and evenly above and below the horizontal line, as they generally are here. Fig. 6 shows a multipanel scatterplot similar to Fig. 5 except done for each case individually. Note that Case 6 appears a bit unusual. Case 6 had no trend, so its predictions are all clustered around zero, consistent with Fig. 4. In general, however, no systematic relationship appears between errors and predicted values for any case. Fig. 7 is a histogram of residuals, which should be normally distributed, and in fact they are. One can construct similar plots of each covariate against residuals and interpret them the same way. To save space, we do not produce all such plots, but Fig. 8 is an example of plotting sessions (x) against residuals (EB) for each case. Again, it shows no systematic relationship of errors to covariates.

2.3. Autocorrelations

Even though these models do not incorporate autocorrelations into their error structure, we can still compute and examine the autocorrelations on the model residuals for each case at any lag. Fig. 9 is a lattice plot of the lag 0 to 5 autocorrelations separately for each case. The dashed lines are 95% confidence bands around zero, and vertical lines extending above or below them reflect statistically significant autocorrelations. Given the small sample, we estimated those bands using $\pm 1.96/\sqrt{n-a}$ rather than $\pm 1.96/\sqrt{n}$ (see Appendix), where n is the length of the time series (26) and a is the number of autocorrelations estimated ($a = 6$). We tried adjusting the autocorrelations for small sample bias (Huitema & McKean, 1991, 1994), but that adjustment made no difference to which
autocorrelations were significant. Lag 0 correlations are 1.0 by definition. Of the remaining 45 autocorrelations, four (9%) are significant, all are negative, and all are at lag 1. We might ignore the significant autocorrelations because we would expect 5% to be significant by chance, except that all the significant ones are at lag 1, not evenly distributed over all lags. An implication is that the standard errors in the analyses we reported may be underestimated. We return to this later.

Fig. 6. A multipanel scatterplot of fitted values (F8) against residuals (E8) for M8 for each individual case.

Fig. 7. A histogram of residuals.
2.4. Overdispersion

For Poisson, negative binomial, and binomial distributions, when the observed conditional variance of the outcome is greater than would be predicted by the reference distribution (in this case, binomial), the data are said to be overdispersed (normal distributions cannot be overdispersed). A scale parameter (\( \phi \); sometimes called a dispersion parameter) indexes dispersion and should be equal to one if the variance is as predicted. We show how to compute it in the Appendix. In the present case, \( \phi = 1.76 \) for M8, indicating overdispersion. Overdispersion causes standard errors to be too low compared to what would be expected under the binomial distribution. How much overdispersion is too much? McCullagh and Nelder (1989) say that anytime \( \phi \geq 1 \), the analyst should consider some adjustment or alternative analysis so as to prevent incorrect statistical inferences.

Overdispersion can result in biased parameter estimates depending on which of the many possible causes of the overdispersion is present (Zuur, Hilbe, & Ieno, 2013). One is zero inflation, that is, the presence of more zeros than the reference distribution would predict. To judge from the raw data and diagnostic plots, that does not seem to be the case here. Zero-inflated data can be analyzed with zero-inflated models (Zuur, Saveliev, & Ieno, 2012). Nonlinear patterns in the data can be a cause, but this is unlikely here given the use of GAMs to model nonlinearities. Those two problems can occur together and would be analyzed with zero inflated GAMs. Autocorrelation can be another cause—plausible in this case given Fig. 9 and possible to address in principle (but not necessarily in practice!) using a generalized additive mixed model that allows for autoregressive error structures. We discuss these in more detail shortly. Missing covariates or interactions can cause overdispersion, which we do not explore here. Outliers can be a cause, but the raw data show little sign this would be a problem. Finally, the variation may just really be bigger than predicted by a binomial model, in which case use of a beta-binomial model may be appropriate. We would pursue the latter in more detail with the Lambert et al. (2006) data, but this approach is more complex and detailed than we can demonstrate in the available space.

A more approximate approach that is simpler to implement than the beta-binomial model is a quasi-binomial model. This approach inflates standard errors by a factor of \( \sqrt{\phi} \). Model comparison uses a quasi-AIC (qAIC) that modifies AIC by dividing \(-2LL\) by the scale parameter \( \phi \):

\[ AIC = \left( -\frac{2LL}{\phi} \right) + (2 \times K). \]
All this can be done easily when working with GLMs. When working with GAMs, however, the mgcv package does not report a likelihood or AIC for the quasibinomial model. A solution is to fit a binomial model to get the likelihood, calculate the dispersion parameter, and compute the quasi-AIC using the qAIC function in the R bbmle program (Bolker, 2013). The Appendix includes syntax for M0 as an example.

However, if you use quasi-models and wish to compare models with different covariates, McCullagh and Nelder (1989) suggest fixing the dispersion to the dispersion of the model with the largest number of covariates (M7). The Appendix shows how to compute the dispersion for each model. The results are (M0) = 2.75, (M1) = 2.69, (M2) = 2.69, (M3) = 2.74, (M4) = 2.70, (M5) = 1.86, (M6) = 2.60, (M7) = 1.25, and (M8) = 1.76. M7 is the largest model, so we use its overdispersion in the qAIC function to obtain

```
<table>
<thead>
<tr>
<th></th>
<th>qAIC</th>
<th>df</th>
</tr>
</thead>
<tbody>
<tr>
<td>M7</td>
<td>0.0</td>
<td>93.866</td>
</tr>
<tr>
<td>M8</td>
<td>32.7</td>
<td>73.499</td>
</tr>
<tr>
<td>M6</td>
<td>50.6</td>
<td>66.000</td>
</tr>
<tr>
<td>M5</td>
<td>132.3</td>
<td>26.491</td>
</tr>
<tr>
<td>M4</td>
<td>160.4</td>
<td>25.323</td>
</tr>
<tr>
<td>M2</td>
<td>168.1</td>
<td>18.557</td>
</tr>
<tr>
<td>M1</td>
<td>168.1</td>
<td>18.557</td>
</tr>
<tr>
<td>M3</td>
<td>192.3</td>
<td>11.526</td>
</tr>
<tr>
<td>M0</td>
<td>193.4</td>
<td>11.000</td>
</tr>
</tbody>
</table>
```

The qAIC suggests the same best fitting models (M7, M8) as AIC, which should always be the case. Now we can run the mgcv GAM quasibinomial model (M8q) for M8 (the model we previously chose under strict binomial assumptions), but with two other

---

Fig. 9. A lattice plot of autocorrelations (acf) on Pearson residuals from lag 0 to 5 separately for each case. The dashed lines are 95% confidence bands around zero, with vertical lines extending above or below those lines being significantly different from zero.
changes to the model syntax. One is to use the dispersion parameter from M7 (McCullagh & Nelder, 1989). The other is to constrain the degrees of freedom in M8q equal to M8, for otherwise the degrees of freedom would change and so the comparison about which effects were significant might change as an artifact of that. The Appendix contains the code, and results are as follows:

### Parametric terms:

<table>
<thead>
<tr>
<th></th>
<th>df</th>
<th>Chi.sq</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>factor(z)</td>
<td>1</td>
<td>10.79</td>
<td>0.001</td>
</tr>
<tr>
<td>factor(Case)</td>
<td>8</td>
<td>46.11</td>
<td>0.000</td>
</tr>
<tr>
<td>factor(z):factor(Case)</td>
<td>8</td>
<td>31.69</td>
<td>0.000</td>
</tr>
</tbody>
</table>

### Parametric coefficients:

|                              | Estimate | Std. Error | z value | Pr(>|z|) |
|------------------------------|----------|------------|---------|---------|
| (Intercept)                  | 1.288    | 0.424      | 3.036   | 0.002   |
| factor(z)1                   | -3.347   | 1.019      | -3.284  | 0.001   |
| factor(Case)2                | 0.420    | 0.501      | 0.838   | 0.402   |
| factor(Case)3                | 1.257    | 0.825      | 1.524   | 0.128   |
| factor(Case)4                | -1.031   | 0.658      | -1.567  | 0.117   |
| factor(Case)5                | -0.194   | 0.568      | -0.341  | 0.733   |
| factor(Case)6                | -0.939   | 0.485      | -1.935  | 0.053   |
| factor(Case)7                | -0.481   | 0.566      | -0.850  | 0.395   |
| factor(Case)8                | -2.207   | 0.565      | -3.905  | 0.000   |
| factor(Case)9                | -0.851   | 0.649      | -1.311  | 0.190   |
| factor(z):factor(Case)2     | -0.233   | 1.115      | -0.209  | 0.834   |
| factor(z):factor(Case)3     | -2.375   | 1.700      | -1.397  | 0.163   |
| factor(z):factor(Case)4     | 2.871    | 1.453      | 1.976   | 0.048   |
| factor(z):factor(Case)5     | 1.631    | 1.433      | 1.138   | 0.255   |
| factor(z):factor(Case)6     | -1.017   | 1.338      | -0.760  | 0.447   |
| factor(z):factor(Case)7     | 0.082    | 1.284      | 0.064   | 0.949   |
| factor(z):factor(Case)8     | 3.970    | 1.374      | 2.891   | 0.004   |
| factor(z):factor(Case)9     | -0.510   | 1.834      | -0.278  | 0.781   |

### Approximate significance of smooth terms:

<table>
<thead>
<tr>
<th></th>
<th>edf</th>
<th>Ref.df</th>
<th>Chi.sq</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>s(x):factor(Case)1</td>
<td>7.375</td>
<td>8.345</td>
<td>18.645</td>
<td>0.020</td>
</tr>
<tr>
<td>s(x):factor(Case)2</td>
<td>1.000</td>
<td>1.000</td>
<td>4.765</td>
<td>0.029</td>
</tr>
<tr>
<td>s(x):factor(Case)3</td>
<td>8.106</td>
<td>8.780</td>
<td>29.731</td>
<td>0.000</td>
</tr>
<tr>
<td>s(x):factor(Case)4</td>
<td>8.063</td>
<td>8.766</td>
<td>31.727</td>
<td>0.000</td>
</tr>
<tr>
<td>s(x):factor(Case)5</td>
<td>8.514</td>
<td>8.934</td>
<td>28.264</td>
<td>0.001</td>
</tr>
<tr>
<td>s(x):factor(Case)6</td>
<td>1.000</td>
<td>1.000</td>
<td>0.169</td>
<td>0.681</td>
</tr>
<tr>
<td>s(x):factor(Case)7</td>
<td>5.062</td>
<td>6.168</td>
<td>13.491</td>
<td>0.039</td>
</tr>
<tr>
<td>s(x):factor(Case)8</td>
<td>7.891</td>
<td>8.680</td>
<td>27.538</td>
<td>0.001</td>
</tr>
<tr>
<td>s(x):factor(Case)9</td>
<td>8.488</td>
<td>8.924</td>
<td>29.459</td>
<td>0.001</td>
</tr>
<tr>
<td>R-sq.(adj)</td>
<td>0.798</td>
<td>Deviance explained = 79.6%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UBRE score</td>
<td>0.825</td>
<td>Scale est. = 1.251</td>
<td>n = 234</td>
<td></td>
</tr>
</tbody>
</table>

Comparing results from M8 and M8q, note that the parameter estimates remain the same, but now all standard errors are increased by the square root of the dispersion—exactly what a quasi-binomial GLM does automatically. Again, for GAM we have to make the same adjustments to the standard errors by computation.

We repeat that it would be worthwhile to use a betabinomial model to analyze these data. The reason is that the quasibinomial models are a very approximate (some would say ad hoc) approach that only attempts to adjust standard errors and AIC tests. It does not deal with the underlying problem that the data do not appear to be distributed according to the expectations of the binomial distribution. A better approach would be to find a distribution that is more appropriate for the data. However, the beta-binomial GAMM model requires Markov chain Monte Carlo (MCMC) programming that is beyond the scope of this article.

### 3. Generalized additive mixed models

In principle, generalized additive mixed models (GAMMs) have two advantages over GAMs. They allow estimating random effects for cases, treatment, trend, and other covariates; and they can include complex error structures. However, the `gamm` routine in mgcv does not allow use of quasi-binomial distributions. Moreover, using `gamm` successfully can require large sample sizes given the increase in parameters to be estimated. We tried several models (see M9 to M11 in Table 1). M9 estimates a first-order autoregressive model from the data. M9a fixes the autoregressive coefficient to .25 instead of estimating it, where .25 is an estimate from the autocorrelation plots. The packages mgcv and gamm4 (Wood, 2009) require that the autocorrelation to be the
same over all cases; MCMC estimation would allow different autocorrelations, but is beyond the scope of this article. Both models would have the advantage of adjusting standard errors to more realistically reflect the modest autocorrelations apparent in the data. However, both models gave error messages suggesting these data cannot support estimating all the parameters. So, instead we ran a GAMM version of a simpler model, M1 (see M10 in Appendix and Table 1) with a first-order autoregressive model:

\[
\text{Correlation Structure: AR(1)}
\]

\[
\text{Formula: ~ 1 | g.1/Case}
\]

Parameter estimate(s):

| Parameter | Estimate | Std. Error | t value | Pr(>|t|) |
|-----------|----------|------------|---------|--------|
| (Intercept) | 1.040 | 0.309 | 3.366 | 0.001 |
| factor(2)1 | -2.652 | 0.230 | 11.509 | 0.517 |
| factor(Case)2 | 0.244 | 0.431 | 0.571 | 0.568 |
| factor(Case)3 | 0.248 | 0.430 | 0.578 | 0.564 |
| factor(Case)4 | 0.222 | 0.427 | 0.520 | 0.604 |
| factor(Case)5 | -0.383 | 0.428 | 0.895 | 0.052 |
| factor(Case)6 | -0.464 | 0.426 | 1.047 | 0.296 |
| factor(Case)7 | -0.559 | 0.417 | 2.399 | 0.022 |
| factor(Case)8 | -0.878 | 0.420 | 2.091 | 0.038* |

Approximate significance of smooth terms:

<table>
<thead>
<tr>
<th>s(x)</th>
<th>edf</th>
<th>Ref.df</th>
<th>Chi.sq</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>s(x)</td>
<td>1.0</td>
<td>1.0</td>
<td>0.045</td>
<td>0.833</td>
</tr>
</tbody>
</table>

The autocorrelation in M10 (phi) is about .20, the coefficient estimates are about the same as M1, but the standard errors are substantially higher in M10 than M1, which is what should happen when the model includes a positive autocorrelation. However, the edf = 1 in M10, which is linear and very different from the significant and nonlinear edf = 8.775 in M1. Two possible explanations for this substantial change are (1) the autoregressive component has taken over all the information from the smoother because failure to model autocorrelation can induce trend, or (2) the time series are too short, so gamm encounters numerical trouble and produces edf = 1. To test the first explanation, we tried fixing the autocorrelation to .20 and then again to zero (M10a), but in both cases edf = 1. This result suggests the data may be too short to support an autoregressive GAMM (the second explanation) because forcing the autocorrelation to be zero in M10a should induce the original trend we saw in M1 if the first explanation is true.

An alternative is to model case as a random effect using the \texttt{gamm4} program (Wood, 2009) without an autoregressive structure. The user will need to download and install both the \texttt{gamm4} and the \texttt{lme4} packages (Bates & Maechler, 2009). Modeling the random effects imposes a compound symmetry correlation structure on the data whereby all observations from the same student are equally correlated with each other. This modeling technique will also adjust the standard errors to take within case dependencies into account. We ran the model two ways, once with the default thin plate spline and once with a cubic regression spline. The former yielded edf = 1. The result for the cubic regression spline (M11) is as follows:

Parametric coefficients:

| Parameter | Estimate | Std. Error | z value | Pr(>|z|) |
|-----------|----------|------------|---------|--------|
| (Intercept) | 0.692 | 0.288 | 2.400 | 0.000 |
| factor(2)1 | -2.442 | 0.472 | -5.174 | 0.000 |

Approximate significance of smooth terms:

<table>
<thead>
<tr>
<th>s(x)</th>
<th>Edf</th>
<th>Ref.df</th>
<th>Chi.sq</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>s(x)</td>
<td>5.584</td>
<td>5.584</td>
<td>2.869</td>
<td>0.078</td>
</tr>
</tbody>
</table>

AIC BIC deviance
684.6 701.9 -337.3 674.6
Random effects:

<table>
<thead>
<tr>
<th>Groups</th>
<th>Name</th>
<th>Variance</th>
<th>Std.Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>fCase</td>
<td>(Intercept)</td>
<td>0.252</td>
<td>0.502</td>
</tr>
<tr>
<td>Xr</td>
<td>s(x)</td>
<td>2.321</td>
<td>1.524</td>
</tr>
</tbody>
</table>

Number of obs: 234, groups: fCase, 9; Xr, 8
Note that the AIC for GAMMs cannot be directly compared to the AIC for GAMs. The edf = 5.584 for the smoothing function, but the χ² test suggests a model with that smoother does not fit better than a model without it. The treatment effect remains significant at about the same magnitude as before. The scale estimate = 1, but that result is misleading because the binomial model forced it to take on that value (recall the quasi-binomial model is not available in \texttt{gamm4}). Calculating dispersion with syntax (see Appendix) shows it is 2.52, clearly overdispersed, so the standard errors are likely to be too large. Ideally, we would address overdispersion with a beta-binomial model but do not do so here for reasons already stated.

So we see that the edf seems to differ between \texttt{gamm} and \texttt{gam}, particularly as a function of the number of observations in the time series and the particular smoothing function used. The \texttt{gamm} help file gives syntax (see Appendix) that helps demonstrate this point by simulating a normally distributed outcome with four smoothed predictors and with the default thin plate splines. If the simulation is done with 400 observations over time, the smoothed terms from \texttt{gamm} yielded edf = 3.01, 2.45, 8.11, and 1.00 for the four terms, while the \texttt{gam} yielded 5.17, 2.36, 8.527, and 1.00 for the same coefficients. These are reasonably consistent with each other. With only 50 observations, however, the smoothed terms from \texttt{gamm} yielded edf = 1.00, 1.00, 5.66, and 1.00 for the four terms, while the \texttt{gam} yielded 1.00, 6.55, 5.71, and 4.35, which are quite different.

With 26 observations as in Lambert et al. (2006), \texttt{gamm} yielded edf = 1.00, 1.00, 1.00, and 1.00 for the four terms. The \texttt{gam} would not run with the default number of knots (places where the curve inflects) \( k = 10 \) because it had more coefficients than data. Reducing knots for the four terms to \( k = 5, 9, 9, \) and 5, respectively, produced \texttt{gam} edfs of 2.09, 7.80, 7.74, and 1.00. So the small number of time points in SCDs may be an obstacle to the use of GAMMs—at least as we have implemented them in this article.

Both \texttt{gamm} and \texttt{gam4} are newer and not as well tested or robust as \texttt{gam} in \texttt{mgcv}. Further, as we saw in M11, changing the type of smoother can change the results in \texttt{gam4}. Preliminary results from analyses we do not present here show that estimating the parameters of the mixed model smoother with MCMC is more robust; it also allows for different distributions (e.g., beta-binomial distributions) A follow-up article that uses MCMC is in preparation.

4. Summary and interpretation

Among all the models, we would tend to choose M8 given its low AIC and qAIC, given the estimation problems in GAMM with small number of observations per case, and the fact that more complex GAMMs with random effects or autoregressive terms had little effect on the conclusions about treatment effectiveness. In this sense, a GA(M)M can be used not only as a primary analytic tool for SCDs but also as sensitivity analyses about the robustness of treatment effects to different assumptions about trend.

Up to this point, we did not show how to interpret the parametric coefficients in the models we tested for the Lambert et al. (2006) data because we first wanted to select a best model. Here we show how to interpret them for M8, but the procedures are general and could be applied to any of the model outputs. Consider first case (A1) in Lambert et al. (2006), which is the reference case in M8. Recall that the model for this analysis is as follows:

$$
\logit(\pi_j) = \alpha_j + \beta_{1j}x_{1ij} + \beta_{2j}x_{2ij} + \beta_{3j}x_{3ij} + s_1(x_{4ij})
$$

where the \( \beta \)'s are the fixed effects estimates from the M8 output, \( x_1 \) is the treatment dummy variable, \( x_2 \) is the case, \( x_3 \) is the interaction, and \( x_4 \) is session, the variable that is being smoothed. For Case 1, then, at the start of baseline this equation becomes

$$
\logit(\pi_j) = 1.288 + 0 + 0 + 0 + s_1(x_{4ij})
$$

where 1.288 is the intercept, the first zero is because the treatment dummy is zero during baseline so the fixed effects estimate \(-3.3467\) is multiplied by zero, and the second and third zeros are because Case 1 is the reference case and has no case or interaction effect. The smoothing term does not have a simple numerical representation, so we do not add it to this interpretation. However, we show shortly that it is simple to obtain from \texttt{gam} the exact predicted value that includes the smoothed term for any given session, which is exactly what is portrayed in the smoothed line in Fig. 4. With this in mind, then, the logit outcome for Case 1 at the start of baseline is 1.288, which converts to an odds of \(\exp(1.288) = 3.626\), meaning the odds
of observing disruptive behavior at the start of baseline for Case A1 was about 3.5 to 1. Further converting this value to a probability yields $P = \frac{\text{Odds}}{1 + \text{Odds}} = .784$. So the probability of observing disruptive behavior in Case 1 during baseline is high.

For the change from the baseline to treatment, the predicted logit is

$$\logit(\pi_j) = 1.288 - 3.3467 + 0 + 0 + s_1(x_{ij})$$

with the only change being the addition of the treatment effect. So the predicted logit is $-2.0587$, yielding an odds of $\exp(-2.0587) = .128$, and probability of observing disruptive behavior of .113, a drop from the previous level of 784. Translating back to the original metric by multiplying by 10, the drop is about 6.7 fewer intervals in which disruptive behavior was observed. Again, these results are consistent with the graph for A1.

A more complicated example would be Case 4. The equation for the predicted logit at the start of baseline is now

$$\logit(\pi_j) = 1.288 + 0 - 1.0312 + 0 + s_1(x_{ij})$$

where the new number in the prediction is the case effect, that is, how much Case 4 differs from the reference Case 1 at the start of the baseline. The prediction is a logit $= .2568$, converting to an odds of 1.293 and a probability of observing a disruptive behavior of .5638. When treatment begins, the predicted logit is

$$\logit(\pi_j) = 1.288 - 3.3467 - 1.0312 + 2.8712 + s_1(x_{ij})$$

where the two new terms are the treatment effect for the reference Case 1, and the interaction of the treatment effect with case (or how much the treatment effect for Case 4 differs from the reference case 1). The predicted logit $= -.2187$, converting to an odds of .8036 and a probability of observing a disruptive behavior of .4455, which is a drop from the original level of .5638. Translating back to the original metric by multiplying by 10, the drop of about 1.2 fewer intervals in which disruptive behavior was observed.

Recall that these predictions do not include the influence of the smoothed term. Case 4 has quite a bit of nonlinearity ($\text{edf} = 8.063$). Code to obtain predicted outcomes that do take the smoothed term into account is in the Appendix. For example, the line reading `DataC1 <- TAll2[ TAll2$Case == 1, ]` extracts Case 1 from the data set, and `Case1 <- DataC1[,"Pi"]` extracts the predicted probabilities from that. To view them, type `Case1`, which results in

<table>
<thead>
<tr>
<th>Case1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.77655903</td>
<td>0.79192523</td>
<td>0.76567938</td>
<td>0.67980029</td>
<td>0.57755571</td>
<td>0.55669106</td>
<td>0.64880831</td>
</tr>
<tr>
<td>8</td>
<td>0.76710750</td>
<td>0.13453385</td>
<td>0.10444676</td>
<td>0.05273234</td>
<td>0.02599244</td>
<td>0.01964756</td>
<td>0.43398291</td>
</tr>
<tr>
<td>15</td>
<td>0.60159735</td>
<td>0.75841192</td>
<td>0.84886452</td>
<td>0.89514004</td>
<td>0.92160814</td>
<td>0.93640256</td>
<td>0.93986607</td>
</tr>
<tr>
<td>22</td>
<td>0.32277545</td>
<td>0.26817584</td>
<td>0.22770613</td>
<td>0.21685158</td>
<td>0.22713804</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

So, the predicted probability for Case 1 at the start of baseline is .7766, a number that takes the smoother into account but is only a little different from the value of .784 calculated previously. At the start of treatment (observation 9 in the `Case1` output above), the predicted probability is .1345, again about the same as the value of .113 calculated above. The predicted probabilities for Case 4 are

<table>
<thead>
<tr>
<th>Case4</th>
<th>87</th>
<th>88</th>
<th>89</th>
<th>90</th>
<th>91</th>
<th>92</th>
<th>93</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.87336433</td>
<td>0.71805457</td>
<td>0.59379008</td>
<td>0.63014690</td>
<td>0.76316061</td>
<td>0.87644366</td>
<td>0.88450308</td>
</tr>
<tr>
<td>94</td>
<td>0.66191954</td>
<td>0.28199909</td>
<td>0.07448150</td>
<td>0.03737597</td>
<td>0.06463484</td>
<td>0.37281858</td>
<td>0.82001760</td>
</tr>
<tr>
<td>101</td>
<td>0.95936090</td>
<td>0.97737381</td>
<td>0.96057682</td>
<td>0.90219466</td>
<td>0.76283439</td>
<td>0.45289917</td>
<td>0.28957580</td>
</tr>
</tbody>
</table>
| 108 | 0.17768893 | 0.10624404 | 0.06701439 | 0.04802530 | 0.03812142

The predicted probabilities for Case 4 are
The first predicted baseline value is .8733, and the first treatment value is .6619, compared to the calculated values of .5638 and .4455, respectively. Here, the addition of the smoothing function did make a more substantial difference to the predicted probabilities. In any case, obtaining the predicted probabilities directly from the model seems preferable to the hand calculations both for ease and because it takes the smoothing function into account.

5. Discussion

We have primarily taken a didactic approach in this article, and we also estimated a wide variety of possible generalized additive (mixed) models in an exploratory mode. One reason is to provide sufficient detail about the many ways that GAMs and GAMMs can be done so that readers can explore them further with their own data. A second reason is that GA(M)Ms have almost never been used in the social sciences, and we hope the clear examples and syntax will help increase their use. Third, the model specifications we presented from M1 to M8 can also be implemented in generalized linear (mixed) models but, of course, without the smoothed terms. Using these model parameterizations would further expand the range of models that can be tested with those approaches compared to past implementations (e.g., Shadish et al., 2014–this issue; Van den Noortgate & Onghena, 2003a, b, 2007, 2008). Fourth, although it would be desirable if we had formal a priori hypotheses about functional form and optimal predictors, the state of the art in the analysis of SCDs is still sufficiently rudimentary that the field will benefit from exploratory analyses that can inform future tests.

At the start of this article, we noted that SCD researchers ideally want an analytic method to consider the full set of information from level, trend, variability, overlap, immediacy of effect, and phase consistency when interpreting a functional relation. In many respects, GAMs are ideally suited for to provide that information. Examples of how they provide that information are as follow:

- Good modeling of trend is clearly the strength of GAMs, as illustrated in Fig. 4.
- The treatment effect coefficient summarizes differences in outcome level and degree of outcome overlap between phases, adjusted for trend.
- The standard error contains information about variability; and in a GA(M)M, the variance component can do the same. Moreover, they can do so not just for the treatment effect but for any term in the model, such as trend.
- Immediacy of the effect is modeled by the nonlinear fit to the data, if needed. In Fig. 4, for example, Cases 2 and 6 have a very immediate effect. In contrast, Case 3 does not respond immediately to treatment, though by the end of treatment response is excellent.
- Phase consistency can be modeled many ways. We showed in this article only a simple way of modeling outcomes over phases. Shadish et al. (2014–this issue) showed how one might model ABAB designs (and also many other designs) in different ways to examine, for instance, the change from A to B, from B to the second A, and from the second A to the second B. Those models are directly adaptable to GAMs.

In addition to all of these benefits of GAMs, GAMs can allow the treatment effect and its standard error to vary over cases and can also allow trend to vary over cases or even within phases. Although no method is perfect, GAMs may, in fact, do better than any other statistical approach to SCDs to meeting the needs of SCD researchers. Currently, it lacks only a standardized effect size and a Bayesian implementation. We are currently working on the latter.

The results of the GAMs in this article are quite similar to the results from Sullivan et al. (in press) using different examples in two respects. First, both articles found large and significant nonlinearity in trend for most cases. Second, treatment effects were robust to nonlinearities in that they remained about the same order of magnitude and were still statistically significant. Although we clearly are in early stages of learning about the use, strengths, and weaknesses of GA(M)Ms for SCD research, this consistency over different examples is encouraging that the approach may prove useful in the analysis of SCDs, especially for diagnosing and assessing the impact of nonlinearities in SCD data. We hope this article encourages more researchers to explore the use of GA(M)Ms both in SCD data and longitudinal data in general.

Appendix A. R Syntax for Generalized Additive (Mixed) Models

First, download and install R (http://www.R-project.org/), and then install the mgcv package by using the Install Packages menu in R. This appendix contains the R commands that yielded the analyses and graphs in this article. They may need to be adapted to the particular circumstances of the user, for example, by writing the appropriate path to the directory in which the user's dataset exists, or by changing the names of variables. R is case sensitive, so be certain to follow upper and lower case syntax exactly. Lines preceded by # are comments and are ignored by R. In the dataset, Case identifies each of the nine cases, x is session number, y is the outcome, z is the treatment dummy, and trial is the count of the number of trials per session.
# Set working directory, load the data, and view names of variables
setwd("c:/R/J School Psychology/Lambert Data")
TAll <- read.table("Lambert Data All2.txt", header = TRUE)
names(TAll)

# Load necessary packages
library(mgcv)
library(lattice)

### Syntax to create Figure 1
### Extract first case from dataset
DataC1 <- TAll[TAll$Case == 1,]
### Set the organization (2 rows, 2 columns) and margins of the plot
par(mfrow = c(2,2), mar = c(5,5,2,2))
### Create upper left plot with raw data
plot(x = DataC1$x,
y = DataC1$y,
xlab = "N Intervals Disruptive Behavior",
ylab = "\"N Intervals Disruptive Behavior\",
pch = 16)
abline(v = 8.5, lty = 1)
abline(v = 13.5, lty = 1)
abline(v = 21.5, lty = 1)
### Create upper right plot with straight lines connecting each dot
plot(x = DataC1$x,
y = DataC1$y,
xlab = "N Intervals Disruptive Behavior",
ylab = "\"N Intervals Disruptive Behavior\",
pch = 16, type = "p")
lines(x = DataC1$x[1:8], y = DataC1$y[1:8])
lines(x = DataC1$x[9:13], y = DataC1$y[9:13])
lines(x = DataC1$x[14:21], y = DataC1$y[14:21])
lines(x = DataC1$x[22:30], y = DataC1$y[22:30])
abline(v = 8.5, lty = 1)
abline(v = 13.5, lty = 1)
abline(v = 21.5, lty = 1)
### Create lower left plot with linear regression line
plot(x = DataC1$x,
y = DataC1$y,
xlab = "Sessions",
ylab = "N Intervals Disruptive Behavior",
pch = 16)
M1 <- lm(y ~ x + factor(z), data = DataC1)
F1 <- fitted(M1)
abline(v = 8.5, lty = 1)
abline(v = 13.5, lty = 1)
abline(v = 21.5, lty = 1)
lines(x = DataC1$x[1:8], y = F1[1:8])
lines(x = DataC1$x[9:13], y = F1[9:13])
lines(x = DataC1$x[14:21], y = F1[14:21])
lines(x = DataC1$x[22:30], y = F1[22:30])
### Create lower right plot with loess smoother
plot(x = DataC1$x,
y = DataC1$y,
xlab = "Sessions",
ylab = "N Intervals Disruptive Behavior",
pch = 16)
M2 <- loess(y ~ x, data = DataC1)
F2 <- fitted(M2)
abline(v = 8.5, lty = 1)
abline(v = 13.5, lty = 1)
abline(v = 21.5, lty = 1)
lines(x = DataC1$x[1:8], y = F2[1:8], lwd = 2, lty = 1)
lines(x = DataC1$x[9:13], y = F2[9:13], lwd = 2, lty = 1)
lines(x = DataC1$x[14:21], y = F2[14:21], lwd = 2, lty = 1)
lines(x = DataC1$x[22:30], y = F2[22:30], lwd = 2, lty = 1)

(continued on next page)
Appendix A (continued)

# Define classroom:
TAll$Class <- rep(c(1, 1, 1, 1, 2, 2, 2, 2, 2), table(TAll$Case))
# Define success (opposite of y disruptive behavior, needed for binomial GLMs):
TAll$Succ <- TAll$trial - TAll$y
# Take the data from 1 to 26
TAll2 <- TAll[TAll$x < 27,]
# Test if the deleted data points differ from the remaining
TAll$drop = ifelse (TAll$x <= 26, 0, 1)
library(lme4)
TAlld <- TAll[TAll$x > 0,]
TAlld$Succ <- TAlld$trial - TAlld$y
fit <- glmer(cbind(y, Succ) ~ drop + (1|Case), data=TAlld, family=binomial)
summary(fit)

# Now try four initial models.
M0 <- gam(cbind(y, Succ) ~ x + factor(z) + factor(Case), data = TAll2, family = binomial)
M1 <- gam(cbind(y, Succ) ~ s(x) + factor(z) + factor(Case), data = TAll2, family = binomial)
M2 <- gam(cbind(y, Succ) ~ s(x) + factor(z) + factor(Class) + factor(Case), data = TAll2, family = binomial)
M3 <- gam(cbind(y, Succ) ~ s(x) + factor(z) + factor(Class), data = TAll2, family = binomial)
# and print a summary of the results for each model.
summary(M0)
summary(M1)
summary(M2)
summary(M3)
# Compare model fit using Akaike’s Information Criterion.
AIC(M0, M1, M2, M3)

# And then some models with interactions
# This model contains a session x treatment (x by z) interaction:
M4 <- gam(cbind(y, Succ) ~ s(x, by = factor(z)) + factor(z) + factor(Case), data = TAll2, family = binomial)
# This model contains a time x student interaction. So each Case is allowed to have its own trend
M5 <- gam(cbind(y, Succ) ~ s(x, by = factor(Case)) + factor(z) + factor(Case), data = TAll2, family = binomial)
# And here each classroom has its own trend
M6 <- gam(cbind(y, Succ) ~ s(x, by = factor(Class)) + factor(z) + factor(Case), data = TAll2, family = binomial)
AIC(M0, M1, M2, M3, M4, M5, M6)

# We make a new variable that will allow different trends during the different phases
TAll2$Case_Z <- factor(paste(TAll2$Case, TAll2$x, sep = "."))
# Now use Case_Z instead of Case in the interaction.
M7 <- gam(cbind(y, Succ) ~ s(x, by = factor(Case_Z)) + factor(z) + factor(Case), data = TAll2, family = binomial)
AIC(M0, M1, M2, M3, M4, M5, M6, M7)

# Now run a model that allows for a different z effect per student.
M8 <- gam(cbind(y, Succ) ~ s(x, by = factor(Case)) + factor(z) + factor(Case), data = TAll2, family = binomial)
AIC(M0, M1, M2, M3, M4, M5, M6, M7, M8)
# Use anova() to obtain test statistics on significance of effects:
anova(M8)
summary(M8)
# Graph predicted values from M8 along with trend line and CIs
P8 <- predict(M8, se = TRUE, type = "link")
TAll2$Pi.up <- exp(P8$fit + 2 * P8$se.fit)/(1+exp(P8$fit + 2 * P8$se.fit))
TAll2$Pi.lo <- exp(P8$fit - 2 * P8$se.fit)/(1+exp(P8$fit - 2 * P8$se.fit))

## this syntax is simple but graphs continuous lines over phases
xyplot(y/trial ~ x | factor(Case),
data = TAll2,
ylim = c(0,1),
panel = function(x,y,subscripts,...){
  panel.points(x,y, pch = 16, col = 1, cex = 1)
  panel.lines(TAll2$x[subscripts],TAll2$Pi[subscripts], col = 1, lty = 2)
  panel.lines(TAll2$x[subscripts],TAll2$Pi.up[subscripts], col = 1, lty = 2)
  panel.lines(TAll2$x[subscripts],TAll2$Pi.lo[subscripts], col = 1, lty = 2)
})

## this syntax is very long but graphs discontinuous lines by phase
xyplot(y/trial ~ x | factor(Case),
data = TAll2,
ylim = c(0,1),
xlab = list(label = "Sessions", cex = 1.5),
ylab = list(label = "N Intervals Disruptive Behavior", cex = 1.5),
panel = function(x,y,subscripts,...){
  panel.points(x,y, pch = 16, col = 1, cex = 1)
  x1 <- TAll2x[subscripts]
y1 <- TAll2P[subscripts]
y1.up <- TAll2P.up[subscripts]
y1.lo <- TAll2P.lo[subscripts]
ThisCase <- TAll2Case[subscripts][1]
if (ThisCase == 1) {
  s1 <- 1; s2 <- 8
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 9; s2 <- 13
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 14; s2 <- 21
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 22; s2 <- 26
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
}
if (ThisCase == 2) {
  s1 <- 1; s2 <- 7
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 8; s2 <- 13
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 14; s2 <- 21
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 22; s2 <- 26
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
}

(continued on next page)
if (ThisCase == 3) {
    s1 <- 1; s2 <- 6
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
    panel.abline(v = s2 + 0.5, lty = 1)
    s1 <- 7; s2 <- 11
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
    panel.abline(v = s2 + 0.5, lty = 1)
    s1 <- 12; s2 <- 18
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
    panel.abline(v = s2 + 0.5, lty = 1)
    s1 <- 19; s2 <- 26
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
    panel.abline(v = s2 + 0.5, lty = 1)
}

if (ThisCase == 4) {
    s1 <- 1; s2 <- 7
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
    panel.abline(v = s2 + 0.5, lty = 1)
    s1 <- 8; s2 <- 12
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
    panel.abline(v = s2 + 0.5, lty = 1)
    s1 <- 13; s2 <- 19
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
    panel.abline(v = s2 + 0.5, lty = 1)
    s1 <- 20; s2 <- 26
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
    panel.abline(v = s2 + 0.5, lty = 1)
}

if (ThisCase == 5) {
    s1 <- 1; s2 <- 10
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
    panel.abline(v = s2 + 0.5, lty = 1)
    s1 <- 11; s2 <- 16
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
    panel.abline(v = s2 + 0.5, lty = 1)
    s1 <- 17; s2 <- 23
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
    panel.abline(v = s2 + 0.5, lty = 1)
    s1 <- 24; s2 <- 26
    panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
    panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
    panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
if (ThisCase == 6) {
  s1 <- 1; s2 <- 8
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 9; s2 <- 12
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 13; s2 <- 18
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 19; s2 <- 26
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
}

if (ThisCase == 7) {
  s1 <- 1; s2 <- 7
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 8; s2 <- 13
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 14; s2 <- 20
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 21; s2 <- 26
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
}

if (ThisCase == 8) {
  s1 <- 1; s2 <- 10
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 11; s2 <- 15
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 16; s2 <- 22
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 23; s2 <- 26
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
}

(continued on next page)
if (ThisCase == 9) {
  s1 <- 1; s2 <- 10
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 11; s2 <- 16
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 17; s2 <- 22
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
  panel.abline(v = s2 + 0.5, lty = 1)
  s1 <- 23; s2 <- 26
  panel.lines(x1[s1:s2], y1[s1:s2], col = 1)
  panel.lines(x1[s1:s2], y1.up[s1:s2], col = 1, lty = 2)
  panel.lines(x1[s1:s2], y1.lo[s1:s2], col = 1, lty = 2)
}

# Various diagnostic plots. Dev.new() allows multiple graphs to stay open in R.
# at the same time, and is not necessary. Material in () is options, and in this
# case sizes the graph.
# First create errors.
E0 <- resid(M0, type = "pearson")
E1 <- resid(M1, type = "pearson")
E2 <- resid(M2, type = "pearson")
E3 <- resid(M3, type = "pearson")
E4 <- resid(M4, type = "pearson")
E5 <- resid(M5, type = "pearson")
E6 <- resid(M6, type = "pearson")
E7 <- resid(M7, type = "pearson")
E8 <- resid(M8, type = "pearson")
# Then create fitted values
F0 <- fitted(M0)
F1 <- fitted(M1)
F2 <- fitted(M2)
F3 <- fitted(M3)
F4 <- fitted(M4)
F5 <- fitted(M5)
F6 <- fitted(M6)
F7 <- fitted(M7)
F8 <- fitted(M8)
dev.new(width=5, height=4)
# Now create diagnostic plots. To save space, only M8 is illustrated.
# Plot fitted values against residuals for M8 over all cases
plot(x=F8, y=E8)
abline(h=0)
dev.new()
# Plot fitted values against residuals for M8 for each case individually
xyplot(E8 ~ F8 | factor(TAll2$Case))
# A histogram of residuals
hist(E8)
# Plot residuals vs each covariate in the model, here session (x)
# to plot another covariate, say treatment (z), use TAll2$z, and so forth.
dev.new()
plot(x=TAll2$X, y = E8)
abline(h=0)
# Same plot but done individually for each case
dev.new()
xyplot(E8 ~ TAll2$X | factor(TAll2$Case))
#Compute the autocorrelation for each case from lag 0 to lag 5
AC <- tapply(E8, FUN = acf, INDEX = TAll2$Case, plot = FALSE, lag.max = 5)

#creates a long vector of the autocorrelations
AC9 <- c(AC$`1`$acf,
         AC$`2`$acf,
         AC$`3`$acf,
         AC$`4`$acf,
         AC$`5`$acf,
         AC$`6`$acf,
         AC$`7`$acf,
         AC$`8`$acf,
         AC$`9`$acf)

#creates a vector of numbers to represent lags 0 to 5 repeated 9 times, one for each case
K <- rep(0:5, 9)

#creates a vector of case identification numbers from 1 to 9, with “each = 6” specifying
#to repeat each number 6 times before moving to the next identification number
ID <- rep(1:9, each = 6)

#binds the above three vectors together into a matrix.
cbind(AC9, K, ID)

#creates a lattice plot of the autocorrelations separately for each case
xyplot(AC9 ~ K | factor(ID),
       ylim = c(-1,1),
       xlab = "Time lag",
       ylab = "acf",
       panel = function(x,y){
         panel.lines(x,y, type = "h", col = 1)
         panel.abline(h = 0)
         panel.abline(h = 1.96 / sqrt(26), lty = 2)
         panel.abline(h = -1.96 / sqrt(26), lty = 2)
       })

#with a small sample, denominator "sqrt(26)" where 26 is the length of the
#series (n) could be changed to "sqrt(n-k)" where k is the number of
#autocorrelations estimated, in this case 6, hence "sqrt(20)

#to do all this with an autocorrelation corrected for small sample
#bias, add 1/n to each autocorrelation
BCacf <- AC9 + (1/26)

#and then substitute BCacf for AC9 as per below
cbind(BCacf, K, ID)

#creates a lattice plot of the autocorrelations separately for each case
xyplot(BCacf ~ K | factor(ID),
       ylim = c(-1,1),
       xlab = "Time lag",
       ylab = "acf",
       panel = function(x,y){
         panel.lines(x,y, type = "h", col = 1)
         panel.abline(h = 0)
         panel.abline(h = 1.96 / sqrt(20), lty = 2)
         panel.abline(h = -1.96 / sqrt(20), lty = 2)
       })

#Examine overdispersion
sum(E8^2) / (M8$df.res)

#compute a quasibinomial model
M8q <- gam(cbind(y,Succ) ~ s(x, by = factor(Case)) + factor(z) * factor(Case),
data = TAll2,
family = quasibinomial)

#syntax to obtain qAIC for quasibinomial model
#first call the bbmle package (which you will have to download
#from CRAN)
library(bbmle)

#then define a function called dfun to compute overdispersion as:
dfun <- function(object) {
  with(object,sum((weights * residuals^2)[weights > 0])/df.residual)
}

#then call the qAIC function from bbmle, replacing "object"
#with the name of your model such as M0. Do not replace "object"
#in dfun above, however; only replace it when you call dfun in qAIC.
(qAIC(object,dispersion=dfun(object)))

#for instance to get qAIC for M0 use
(qAIC(M0,dispersion=dfun(M0)))

#define a function to compute dispersion (scale parameter) for each
#quasibinomial model
MyDisp <- function(m){
  e <- resid(m, type = "pearson")
  Overd <- sum(e^2) / m$df.res
  Overd
}(continued on next page)
Now obtain dispersion for each model from the function

MyDisp(M0)
MyDisp(M1)
MyDisp(M2)
MyDisp(M3)
MyDisp(M4)
MyDisp(M5)
MyDisp(M6)
MyDisp(M7)
MyDisp(M8)

#M7 is the largest model, so use the overdispersion of M7 to compute qAIC for all nine models, and table using the ICtab function.

ICtab(M0, M1, M2, M3, M4, M5, M6, M7, M8, dispersion=dfun(M7), type="qAIC")

Now compute quasibinomial results using mgcv, using dispersion from M7 and degrees of freedom from M8

Dispersion7 <- MyDisp(M7)
M8q <- gamm(cbind(y,Fail) ~ s(x, by = factor(Case)) + factor(z) * factor(Case), data = TAll2, family = quasibinomial, scale = Dispersion7, sp = M8sp )
anova(M8q)

A generalized additive mixed model to estimate a first-order autoregressive model based on the model in M8

M9 <- gamm(cbind(y,Succ) ~ s(x, by = factor(Case)) + factor(z) * factor(Case), data = TAll2, correlation = corAR1(form =~ x | Case), family = binomial)

A GAMM that incorporates a fixed lag 1 autocorrelation = 25 in the model

M9a <- gamm(cbind(y,Succ) ~ s(x, by = factor(Case)) + factor(z) * factor(Case), data = TAll2, correlation = corAR1(value = 0.25, fixed = TRUE, form =~ x | Case), family = binomial)

Revert to one of the more easier models, and add a temporal correlation structure

This is the gamm equivalent of model M1 with an autocorrelation

M10 <- gamm(cbind(y,Succ) ~ s(x) + factor(z) + factor(Case), data = TAll2, correlation = corAR1(form =~ 1|Case), family = binomial)

summary(M10$mer)
summary(M10$gam)
anova(M10$gam)

Fix autocorrelation to .20 and see if edf > 1.

M10a <- gamm(cbind(y,Succ) ~ s(x) + factor(z) + factor(Case), data = TAll2, correlation = corAR1(value = 0.2, fixed = TRUE, form =~ 1|Case), family = binomial)

GAMM using student as random effect using gamm4

library(gamm4)

TAll2$Case <- factor(TAll2$Case)
M11 <- gamm4(cbind(y,Succ) ~ s(x, bs = "cr") + factor(z), data = TAll2, random = ~(1|Case), family = binomial)

summary(M11$gam)
summary(M11$mer)

Test M11 for overdispersion

E11 <- resid(M11$mer, type = "pearson")

sum(E11^2) / (nrow(TAll2) – 8.5)
References


