repolr: An R Package for Fitting Proportional-odds Models to Repeated Ordinal Scores

Nick Parsons
University of Warwick
March 6, 2017

Email: nick.parsons@warwick.ac.uk
R Package **repolr**: https://CRAN.R-project.org/package=repolr

Abstract

Modelling repeated ordinal score data is a common statistical problem, across many application areas. The proportional-odds model is widely applied to such repeated ordinal scores and can be fitted in the **repolr** package (**re**peated measures **p**roportional **o**dds **l**ogistic **r**egression) in R using the method of generalized estimating equations (GEE). The GEE approach specifies a model for the mean of the correlated observations within clusters of repeated scores for each individual without fully specifying the joint distribution of the observations. This paper describes the core features of package **repolr**, which has undergone extensive changes since the first release in 2008, for the first time. A number of example datasets and extensive R code are used to illustrate a range of data analysis tasks that users of **repolr** may typically wish to undertake.

Keywords: repeated ordinal scores, generalized estimating equations, **repolr**, R.

1. Introduction

Modelling repeated ordinal scores (i.e., a sequence of assessments made on an an ordered categorical scale by or on the same individual or experimental unit at a fixed number of time-points) is a common statistical problem that has been studied by many researchers; Liu and Agresti (2005) and Agresti and Natarajan (2001) provide comprehensive reviews of available models and methods. Without doubt, the most widely used approach to modelling repeated ordinal scores, for example see Lipsitz, Kim, and Zhao (1994) and Kenward, Lesaffre, and Molenberghs (1994), is the generalized estimating equation (GEE) method originally proposed by Liang and Zeger (1986) for the proportional-odds model (McCullagh 1980). Briefly, the proportional-odds model follows from considering ordinal scores to be realizations of continuous unobservable measures that are difficult or impossible to assess directly. For instance in medical applications this might be a patient's experience of pain after surgery or in behavioural sciences an individual's quality of life. In the absence of objective methods to quantify characteristics of this type directly, we typically make assessments using a small number of ordered categories, often identified with common descriptive words such as severe, moderate, good or poor, in an attempt to provide an objective evaluation. The proportional-odds model is characterized by cut-point parameters that define the divisions between such ordered score categories, and regression parameters that characterize the relationship between model covariates and the ordinal scores that are the *same* for each category defined by the cut-points (i.e., the regression parameters do not depend on the ordinal score category). A number of variations of the basic GEE methodology have been suggested for modelling repeated ordinal scores, some specifically in order to overcome problems in estimation of the correlation parameters that can occasionally occur during model fitting. Prominent amongst these methods, for the proportional-odds model, is that proposed by Parsons, Edmondson, and Gilmour (2006). This method is discussed in detail here, together with a description of implementation in package **repolr** (**re**peated measures **p**roportional **o**dds **l**ogistic **r**egression) (Parsons 2016) in R (R Core Team 2015).

Prior to describing the features of **repolr**, it is informative to review GEE software for repeated ordinal responses available elsewhere. GEE methods for fitting the proportional-odds model to repeated ordinal data are available in widely used commercial packages such as SAS (SAS Institute Inc. 2015) and SPSS (IBM Corp. 2013). For instance, the `GENLIN` command in the latter software provides five options for the working correlation model (independence, exchangeability, AR1, M-independence and unstructured), and uses some (unspecified) functions of the residuals of the regression model for the cumulative probabilities to estimate the association parameters. However, the most informative comparison for **repolr** is with functions within the R packages **geepack** (Halekoh, Høsjsgaard, and Yan 2006) and **multgee** (Touloumis 2015), as these are equally general and the implementations of the methods used to estimate the association parameters, which is where all the methods described here differ from one another, are much more explicit. The function `ordgee` in R package **geepack** (Højsgaard, Halekoh, and Yan 2016) implements the GEE approach of Heagerty and Zeger (1996) who model pairwise associations between binary scores, determined from the ordinal scores, using global odds ratios; independence, exchangeability, unstructured and user-defined working correlation models are available for logit, probit and complementary log-log link functions for the regression model for the mean. The package **multgee** (Touloumis 2016) allows models to be fitted to repeated ordinal scores using function `ordLORgee` that implements a local odds ratio approach to estimating the association parameters, or rather nuisance parameters using the preferred terminology of Touloumis, Agresti, and Kateri (2013). A number of options are available for the local odds ratio structure, with the recommended option for ordinal scores being the uniform structure which assumes all ratios are identical. Package **multgee** offers the cauchit link function, in addition to those available in **geepack**, and also adjacent category logit models.

A recent review of available software and performance for implementation of GEE methodology for longitudinal ordinal data (Nooraee, Molenberghs, and van den Heuvel 2014) reviewed, described in detail and compared all of the most widely used options for model fitting in this setting; the authors concluded that **multgee** and **repolr** provided very similar results over the settings tested, and that these were consistent with results from SAS and SPSS. Nooraee *et al.* (2014) did not include **geepack** in their simulation study as it gave results that were significantly different from the other packages. The software implementations described here use a variety of methods for model fitting, but the main distinction is in the formulation of the association parameter model, and the importance given to parameter estimates from such models. **repolr** is much more specific than the other R packages described here in that it is restricted to marginal logistic models for ordinal responses and formulates the working correlation model for the GEE methodology, that expresses and separates the correlations between derived binary variables, in a natural manner. It provides consistent estimates of the correlation parameter between time-points (Parsons *et al.* 2006), and as such this parameter is regarded as having some inferential value and is reported routinely, rather than being regarded simply as a nuisance parameter. Package **repolr** also allows for flexible modelling of the so-called cut-point parameters, that define the divisions between ordinal score categories, using polynomial contrasts, that is often of particular interest for long or composite ordinal scores (Parsons 2013).

The first version of **repolr**, which was available in late 2008, required package **gee** for regression parameter estimation, with association parameter estimation and updating implemented within **repolr**. The latest versions of **repolr** use code written in C++, with the aid of R packages **Rcpp** (Eddelbuettel and Francois 2011; Eddelbuettel 2013) and **RcppArmadillo** (Eddelbuettel and Sanderson 2014) with the latter providing access to the widely used Armadillo templated C++ linear algebra library (Sanderson 2010). A brief description of the underlying model fitting algorithm for **repolr** was published in 2009 (Parsons, Costa, Achten, and Stallard 2009), but this contained no advice on practical application or implementation in R as it was written prior to the publication of **repolr**, despite the publication date of the article being being after the **repolr** was available. The syntax and available options in **repolr** have changed considerably from earlier versions, however, basic model fitting implementation is unlikely to change in any updates ($\geq$ v3.2) of **repolr**, so given the stability of the current release, now is an opportune occasion to provide a comprehensive description of the package and usage for analysis of repeated ordinal score data.

The paper illustrates the usage of the **repolr** package by application to three example datasets. Section 2 describes the GEE methodology of Parsons *et al.* (2006) and implementation in R, Section 3 presents three distinct datasets, available within the package, and describes typical analysis issues for repeated ordinal scores with code for implementation in **repolr** and Section 4 closes the paper with some concluding remarks.

2. Model

2.1. Proportional-odds

Let N experimental units be scored on an ordinal scale with K categories, which can take integer values from 1 to K (the optimum score), at each of T time points. The ordinal score for experimental unit i at time-point t is O_{it} and the vector of scores for the this experimental unit over the set of T time points is $O_i = (O_{i1}, O_{i2}, \ldots, O_{iT})$. The probability $\mu_{itk} = P(O_{it} \leq k)$ for ordinal score O_{it}, and inverse logit function $f(\mathrm{y}) = \exp(\mathrm{y})/(1 + \exp(\mathrm{y}))$, can be related to a vector of S explanatory variables x_{it} by the proportional-odds model based on cumulative logits (McCullagh and Nelder 1989).

$$\mu_{itk} = f\left(\beta_{0k} + \beta \mathrm{x}_{it}\right). \tag{1}$$

The cut-points $\beta_0 = (\beta_{01}, \beta_{02}, \ldots, \beta_{0(K-1)})$, where $-\infty < \beta_{01} < \beta_{02} < \cdots < \beta_{0K-1} < \infty$, define the divisions between the ordinal score categories on the cumulative logit scale and effectively transform the ordinal scale to a continuous scale based on the linear predictor $\beta \mathrm{x}_{it}$. Regression parameters β_0 and $\beta = (\beta_1, \beta_2, \ldots, \beta_S)$ can be estimated using the method of generalized estimating equations (GEE), originally proposed by Liang and Zeger (1986).

2.2. Parameter estimation

Regression parameters

At each time-point the ordinal score for each experimental unit O_{it} can be transformed into a set of $K-1$ binary variables such that $Z_{itk} = 1$ if $O_{it} \leq k$ or $Z_{itk} = 0$ if $O_{it} > k$, to give a data vector $\mathrm{Z}_{it} = (Z_{it1}, Z_{it2}, \ldots, Z_{it(K-1)})$ and a complete data vector $\mathrm{Z}_i = (\mathrm{Z}_{i1}, \mathrm{Z}_{i2}, \ldots, \mathrm{Z}_{iT})^\top$ for each experimental unit. The $T \times S$ data matrix of explanatory variables for each experimental unit is $\mathbf{X}_{0i} = [\mathrm{x}_{i1}, \mathrm{x}_{i2}, \ldots, \mathrm{x}_{iT}]^\top$, and the complete $((K-1)+S) \times T(K-1)$ data matrix including the cut-points is $\mathbf{X}_i = [1_T \otimes \mathbf{I}_{K-1}, \mathbf{X}_{0i} \otimes 1_{K-1}]^\top$, where 1_T and 1_{K-1} are T and $(K-1)$-dimensional vectors of unit elements, and $\mathbf{I}_{K-1}$ is a $(K-1)$-dimensional unit matrix. If $\mathrm{E}(\mathrm{Z}_i) = \mu_i$, where μ_i is defined in an analogous manner to the complete binary data vector Z_i, and the complete $(K-1)+S$ vector of regression coefficients is $\beta_c = (\beta_0, \beta)^\top$ then the proportional-odds model for each experimental unit can be expressed as

$$\mu_i = f\left(\mathbf{X}_i^{\mathrm{T}} \beta_c\right). \tag{2}$$

The GEE for this model can be expressed in matrix form by first writing the $T(K-1)$ diagonal matrix of partial derivatives $\mathbf{F}_i = \mathrm{diag}\{f'(\beta_c^{\mathrm{T}} \mathbf{X}_i)\}$, where $f'(\mathrm{y}) = \exp(\mathrm{y})/(1+\exp(\mathrm{y}))^2$, $\mathbf{D}_i = \mathbf{X}_i \mathbf{F}_i$ and and the $((K-1)+S) \times TN(K-1)$ matrix $\mathbf{D} = [\mathbf{D}_1, \mathbf{D}_2 \ldots, \mathbf{D}_N]$. The variances of the probabilities μ_{itk} are given by $\mu_{itk}(1-\mu_{itk})$, with the full model $TN(K-1) \times TN(K-1)$ diagonal matrix $\mathbf{U}^{-1/2}$ containing the square roots of the reciprocals of the variances on the leading diagonal. The $TN(K-1) \times TN(K-1)$ block diagonal matrix, of correlations between derived binary variables Z_{itk} at the same time and between different times, is denoted by $\mathbf{R}$. In **repolr** the between time correlation is characterized by a single (ancillary) association parameter α that, when estimating the regression coefficients, is assumed to be known. Expressing the GEE in matrix form as $\mathbf{D}\mathbf{W}^{-1}(\mathrm{Z} - \mu)$ where $\mathbf{W} = \mathbf{U}^{-1/2} \mathbf{R}^{-1} \mathbf{U}^{-1/2}$, $\mathrm{Z} = (\mathrm{Z}_1^\top, \ldots, \mathrm{Z}_N^\top)^\top$ and $\mu = (\mu_1^\top, \ldots, \mu_N^\top)^\top$, and equating to zero leads to the following updating equation for β_c,

$$\beta_c^{(k)} = \beta_c^{(k-1)} + (\mathbf{D}\mathbf{W}^{-1}\mathbf{D})^{-1} \mathbf{D}\mathbf{W}^{-1}(\mathrm{Z} - \mu). \tag{3}$$

The robust covariance matrix of the regression parameters is given by

$$\mathbf{V}_{\beta_\mathbf{c}} = (\mathbf{D}\mathbf{W}^{-\mathbf{1}}\mathbf{D})^{-\mathbf{1}} (\mathbf{D}\mathbf{W}^{-\mathbf{1}} \mathrm{cov}(\mathrm{Z}) \mathbf{W}^{-\mathbf{1}}\mathbf{D}) (\mathbf{D}\mathbf{W}^{-\mathbf{1}}\mathbf{D})^{-\mathbf{1}}, \tag{4}$$

where the $TN(K-1) \times TN(K-1)$ covariance matrix $\mathrm{cov}(\mathrm{Z})$ is constructed from the residual products $(\mathrm{Z}_i - \mu_i)(\mathrm{Z}_i - \mu_i)^\top$ to estimate the between time covariances and model based estimates of covariances between binary variables at the same time-points, using current estimates of β_c. Equations 3 and 4 are consistent estimators, as $N \to \infty$, even if the model covariance matrix $\mathbf{W}$ is misspecified. In order to improve efficiency for typical N, we attempt to choose the working correlation matrix $\mathbf{R}$ to be as close as possible to the true correlation matrix of Z.

Association parameter

The specification of the working correlation matrix $\mathbf{R}$ in **repolr** is described in detail by Parsons *et al.* (2006) and Parsons *et al.* (2009). In short the correlation structure is assumed to be the same for all N experimental units, so suppressing indexing on i the correlation between binary variables $r_{tk,sj}$ can be considered to be the product of two components; (i) the correlation ρ_{kj} between binary variables at each time point and (ii) the correlation between binary variables at differing time points $\alpha(t,s)$. The former correlation is given by $\rho_{k,j} = \rho_{j,k} = \exp(\beta_{0j} - \beta_{ok})^{1/2}$ (Kenward *et al.* 1994) and the latter can be modelled in **repolr** using either the first order autoregressive correlation model (AR1) described by Parsons *et al.* (2006) where $\alpha(t,s) = \alpha^{|t-s|}$ or the uniform correlation model of Parsons *et al.* (2009) where $\alpha(t,s) = \alpha$ for all $t \neq s$. Given estimates of β_c, the working correlation matrix is a function of a single association parameter α. In **repolr** α is estimated by minimizing $\log|\mathbf{V}_{\beta_\mathbf{c}}|$. Given a current estimate of $\mathbf{V}_{\beta_\mathbf{c}}$, an updated estimate of α is given after k iterations of Newton's method by the following expression,

$$\alpha^{(k)} = \alpha^{(k-1)} - \frac{\partial \log|\mathbf{V}_{\beta_\mathbf{c}}|}{\partial \alpha} \left\{ \frac{\partial^2 \log|\mathbf{V}_{\beta_\mathbf{c}}|}{\partial \alpha^2} \right\}^{-1}. \tag{5}$$

Analytic expressions for $\partial \log|\mathbf{V}_{\beta_\mathbf{c}}|/\partial\alpha$ and $\partial^2 \log|\mathbf{V}_{\beta_\mathbf{c}}|/\partial\alpha^2$ are available for the AR1 (Parsons *et al.* 2006) and uniform correlation (Parsons *et al.* 2009) models. Although analytic expressions are available, for more complex models where either K or S or both are large, it is often quicker to use finite differencing to estimate the partial derivatives; i.e., $\partial \log|\mathbf{V}_{\beta_\mathbf{c}}(\alpha)|/\partial\alpha = \frac{1}{2}h^{-1}\{\log|\mathbf{V}_{\beta_\mathbf{c}}(\alpha+h)| - \log|\mathbf{V}_{\beta_\mathbf{c}}(\alpha-h)|\}$ and $\partial^2 \log|\mathbf{V}_{\beta_\mathbf{c}}(\alpha)|/\partial^2\alpha = h^{-2}\{\log|\mathbf{V}_{\beta_\mathbf{c}}(\alpha+h)| - 2\log|\mathbf{V}_{\beta_\mathbf{c}}(\alpha)| + \log|\mathbf{V}_{\beta_\mathbf{c}}(\alpha-h)|\}$. In practice, a transformation of α is used, $\phi = \log(\alpha) - \log(1-\alpha)$, in order to ensure that α is constrained to lie in the interval $(0,1)$ during model fitting.

Implementation

Model fitting initiates from starting estimates for β_c and α, to give reasonably stable updated estimates for β_c after a number of iterations of equation (3). The derivatives of the current estimates of $\mathbf{V}_{\beta_\mathbf{c}}$ are then used to provide updated estimates of α after a number of iterations of equation (5) . Optimization proceeds by alternating between updating estimates using equations (3) and (5) until convergence is achieved. Convergence in **repolr** can be controlled using the argument `fit.opt` (see Section 3.1 for more details), with defaults `c(cmaxit = 10, omaxit = 5, ctol = 0.001, otol = 0.00001, h = 0.01)`, that control the maximum number of iterations for updating α, the maximum number of iterations for updating β_c within each of the updating steps for α, the convergence tolerances for α and β_c, and the interval h for finite differencing, if this option is selected. Convergence is generally achieved, for default settings of `fit.opt`, within 5 to 10 iterations. However, if issues do arise, and convergence cannot be achieve before `cmaxit` is reached, then this is usually due to either an inappropriate working correlation structure (i) or regression model (ii), leading to problematic estimates of α in the former or β_c in the latter case. It is often not possible to determine whether lack of convergence is due to (i) or (ii) or both, so we would advise changing each in turn to see if this solves the problem, if not then consider increasing `cmaxit`.

3. Examples

3.1. Specifying and fitting the model

Table 1 shows hip pain scores for the first five individual patients from the `HHSpain` dataset, available in in package **repolr**, at one, two and five years post-operatively (full population is

58 patients) after hip resurfacing surgery. Pain in the hip joint is assessed by the patient as either none (1), slight (2), mild (3) or moderate or marked (4). The primary interest in this study is to understand the effects of time and patient gender on post-operative pain scores. Using the conventional R syntax for linear modelling, for ordinal response variable of pain scores (variable `HHSpain` in dataset `HHSpain`) and explanatory variables (`Time` and `Sex`) this is expressed as `HHSpain ~ Time * Sex` with model fitting in `repolr` proceeding as follows:

```
R> library("repolr")
R> data("HHSpain", package = "repolr")
R> mod.0 <- repolr(HHSpain ~ Time * Sex, data = HHSpain, categories = 4,
+    subjects = "Patient", times = c(1, 2, 5),
+    corr.mod = "uniform", alpha = 0.5)
```

It is important that data are structured in an analogous manner to that shown in Table 1 for model fitting to be implemented correctly. Data should be ordered by a *subjects* variable, which takes integer values from 1 to N with no missing entries allowed, indicating the data clusters (e.g., patients or experimental units). This must be specified in the call to function `repolr` with the name of the subjects field in the index dataset (e.g., `data = HHSpain`) given explicitly (e.g., `subjects = "patient"`). These subject clusters must all be of the same size, that is they should contain the same number of repeated ordinal scores, if necessary by inserting missing values in other data fields. Ordinal scores, observed within each of the subject clusters, should be ordered temporally; i.e., from first to last observation time. Strictly speaking it is not necessary to include a variable indicating the temporal spacing of observations (Table 1; Time) in the dataset if this term is not included in the model formula, as this information must always be provided explicitly by the `times` argument. Times should be ordered integers starting from one and spaced to indicate the relative distance between the successive times at which all observations were made. For instance, four observations at equally spaced times would be entered as 1, 2, 3 and 4, whereas for the `HHSpain` data the observations were at one, two and five years, so this is provided as a vector as follows `c(1, 2, 5)`. In addition to the model formula, the only other required argument for `repolr` is the number of ordinal score categories (e.g., `categories = 4`). The default setting for the working correlation model (`corr.mod`) is taken to be `"independence"` and the initial (starting) value for the correlation parameter (`alpha`), if required, is set to 0.5. For our example, we have selected the `"uniform"` working correlation model with starting value for iteration of 0.5, for exemplary purposes. The model is summarised, in conventional R syntax, as follows:

```
R> summary(mod.0)

repolr: 2016-02-26 version 3.4

Call:
repolr(formula = HHSpain ~ Time * Sex, subjects = "Patient",
data = HHSpain, times = c(1, 2, 5), categories = 4, corr.mod = "uniform",
alpha = 0.5)

Coefficients:
coeff    se.robust  z.robust  p.value
```

Patient	Sex	Time	HHSpain
1	M	1	3
1	M	2	1
1	M	5	4
2	F	1	1
2	F	2	1
2	F	5	1
3	M	1	1
3	M	2	3
3	M	5	3
4	F	1	1
4	F	2	2
4	F	5	1
5	M	1	1
5	M	2	1
5	M	5	1

Table 1: Hip pain scores for first five patients from `HHSpain` dataset.

```
cuts1|2      0.6284  0.4334      1.4499    0.1471
cuts2|3      1.7038  0.4538      3.7545    0.0002
cuts3|4      3.0118  0.5039      5.9770    0.0000
Time        -0.2140  0.1039     -2.0597    0.0394
SexM         0.5230  0.5292      0.9883    0.3230
Time:SexM    0.0332  0.1152      0.2882    0.7732

Correlation Structure:  uniform
Estimated Correlation:  0.1043
```

The model coefficients, where `cuts1|2`, `cuts2|3` and `3|4` are estimates of β_0, are accessed using conventional R model syntax for linear modelling as `coef(mod.0)` and the variance covariance matrix of the regression parameters as `vcov(mod.0)`. By default this reports the robust or so-called sandwich estimator $\mathbf{V}_{\beta_c}$; the naive or model based estimator $(\mathbf{DW}^{-1}\mathbf{D})^{-1}$ is available as `vcov(mod.0, robust.var = FALSE)`. Confidence intervals for regression parameters are also available at the desired level as `confint(mod.0, robust.var = TRUE, level = 0.95)`. In general, the robust variance estimator is the preferred option, and default setting, in `repolr`. However summaries based on the naive estimator can be obtained, for model `mod.0` as follows `summary(mod.0, robust.var = FALSE)`. Regression parameter β_s describes the effect of explanatory variable s on the ordinal score such that β_s is the increase in log-odds of falling into or below each score category associated with a unit increase in s. Therefore, a negative regression coefficient indicates a tendency for the ordinal score to *increase* as the variable increases; i.e., for the `HHSpain` dataset, the negative `Time` coefficient and p-value ≤ 0.05 suggests that there is evidence for a statistically significant increase in pain scores as time proceeds from years one to five. Similarly, a positive regression coefficient indicates a tendency for the ordinal score to decrease as the variable increases; i.e., in this example, scores were overall lower for males (`M`) than for females, but there was no evidence that this difference was statistically significant (p-value > 0.05). It should be noted that this parametrization is different from that sometimes used elsewhere, where a negative sign in Equation 1 is used to ensure that a positive value for β_s leads to an increase in the probability of the higher numbered ordinal score categories. Methods to extract model residuals and fitted values are also available for fitted models; `methods(class = "repolr")`. Predicted values can also be obtained for new data, provided this is formatted in the same manner as for model fitting. For example, predicted values and standard errors, on the scale of the linear predictor, for a new male subject who is given an arbitrary subject number (in this case 100) can be obtained as follows:

```
R> predict(mod.0, newdata = data.frame(Patient = rep(100, 3),
+         Time = c(1, 2, 5), Sex = factor(rep(1, 3), levels=1:2,
+         labels=c("F", "M"))), type = "link", se.fit = TRUE)

$fit
100.1  100.2  100.3  100.4  100.5  100.6  100.7  100.8  100.9
0.414  1.490  2.798  0.200  1.276  2.584 -0.442  0.634  1.942

$se.fit
100.1 100.2 100.3 100.4 100.5 100.6 100.7 100.8 100.9
0.378 0.395 0.451 0.345 0.357 0.418 0.422 0.414 0.468
```

The naming conventions in reporting such output in **repolr** are to show the subject number and derived binary variable number; i.e., in this particular example, `100.1` is the predicted value for the first of the derived binary variables for subject 100, where we have $(K - 1) \times T$ derived binary variables for each subject and in this example $K = 4$ and $T = 3$.

The correlation parameter estimate can be accessed as `mod.0$alpha` and the full working correlation matrix $\mathbf{R}$ can also be examined:

```
R> work.corr(mod.0)

      1     2     3     4     5     6     7     8     9
1 1.000 0.584 0.304 0.104 0.061 0.032 0.104 0.061 0.032
2 0.584 1.000 0.520 0.061 0.104 0.054 0.061 0.104 0.054
3 0.304 0.520 1.000 0.032 0.054 0.104 0.032 0.054 0.104
4 0.104 0.061 0.032 1.000 0.584 0.304 0.104 0.061 0.032
5 0.061 0.104 0.054 0.584 1.000 0.520 0.061 0.104 0.054
6 0.032 0.054 0.104 0.304 0.520 1.000 0.032 0.054 0.104
7 0.104 0.061 0.032 0.104 0.061 0.032 1.000 0.584 0.304
8 0.061 0.104 0.054 0.061 0.104 0.054 0.584 1.000 0.520
9 0.032 0.054 0.104 0.032 0.054 0.104 0.304 0.520 1.000
```

The 3×3 diagonal blocks are the correlations between the binary scores at the same time-points and the 3×3 off-diagonal blocks are, for the uniform correlation model used here, given by α times the diagonal blocks. For the first order autoregressive working correlation model, the off-diagonal blocks would differ depending on the separation between time points, as described in Section 2. Updating model `mod.0` by selecting a first autoregressive working correlation model, rather than the uniform model gives the following outcome:

```
R> mod.1 <- update(mod.0, corr.mod = "ar1")
```

```
Warning message:
In repolr(formula = HHSpain ~ Time * Sex, subjects = "Patient",  :
Model did not converge: iter 11 and crit 0.00599302442179872
```

```
R> mod.1$convergence
```

```
[1] FALSE
```

The R model updating function update has been used here for brevity of presentation. Alternatively, the above model could, of course, have been obtained more explicitly, by simply changing the option for argument corr.mod to "ar1", leaving all other arguments unchanged and using the same syntax as for model mod.0, but assigning to object mod.1. In general, parameter estimation is not sensitive to the choice of working correlation model, however occasionally a pathologically bad choice will cause non-convergence of the algorithm. This is the case here. After eleven iterations the fit criterion, ctol, is still above the default threshold setting of 0.001 at iteration cmaxit+1. Experience suggests that convergence is rarely achieved in such settings simply by increasing cmaxit, but one may be willing to accept a larger, but still relatively small, value for ctol. For instance the following model does achieve convergence after eight iterations:

```
R> mod.1 <- update(mod.0, corr.mod = "ar1",
+    fit.opt = c(NA, NA, ctol=0.01, NA, NA))
R> mod.1$convergence
```

```
[1] TRUE
```

```
R> mod.1$iter
```

```
[1] 8
```

It can occasionally be useful to fix the correlation parameter (alpha), rather than estimate it, during model fitting. For instance, if analysis from a previous large study had provided strong evidence in favour of a particular correlation structure and value for α, one might wish to fix this in a subsequent analyses of any smaller subsidiary studies. This can be achieved by setting the fixed argument to TRUE and alpha explicitly. For example we could update the hip pain score model (mod.0), by fixing alpha to be 0.3 as follows.

```
R> mod.2 <- update(mod.0, fixed = TRUE, alpha = 0.3)
R> mod.2$alpha
```

```
[1] 0.3
```

The method of finite differencing to estimate the partial derivatives, $\partial \log|\mathbf{V}_{\beta_\mathbf{c}}|/\partial\alpha$ and $\partial^2 \log|\mathbf{V}_{\beta_\mathbf{c}}|/\partial\alpha^2$, can be implemented by setting the argument `diffmeth = "numeric"`, in contrast to the default setting of `"analytic"`. The former method is generally always faster than the latter, with little loss in precision of parameter estimates provided h is small enough. Formatting of outputs and summaries will look identical for both methods, the only differences are likely to be small changes to parameter estimates and standard errors, dependent on the setting used for `h`.

3.2. Measures of fit

Two of the most common tasks during model fitting in **repolr** are (i) the selection of the most appropriate working correlation model and (ii) the selection of the best subset of variables. Both of these tasks can be informed by the `QIC` (`Q`uasilikelihood model `I`nformation `C`riterion) function that allows variable selection to be undertaken in an analogous manner to the Akaike information criterion, with the model likelihood replaced by the model quasilikelihood under the independence model and the penalty term given by $2\times((K-1)+S)$ (Pan 2001). Using the notation of Pan (2001), this model fit criterion is called `QICu`, where when choosing between two or more models the one with the smallest `QICu` measure is preferred. Function `QIC` also allows correlation structures to be compared between competing models, that are otherwise identical, using a penalty term given by $2 \times \text{trace}(\mathbf{H}_{\beta_\mathbf{c}}^{-1}\mathbf{V}_{\beta_\mathbf{c}})$, where $\mathbf{H}_{\beta_\mathbf{c}} = (\mathbf{D}\mathbf{W}^{-1}\mathbf{D})^{-1}$ is the naive (model-based) covariance matrix for an independence model (Pan 2001). This latter criterion is labelled simply as `QIC` in output to calls to function `QIC`, and is discussed in relation to the data described here. Whereas use of `QICu` is described in Section 3.3. Dataset `mobility` provides ordinal scores (using a scale with three categories) to assess movement and function for patients after surgery, in a clinical trial comparing two methods of bone fixation (A and B), at four equally spaced time points. After adjusting for age and gender, it is of interest to asses the size of the treatment effect and its interaction with time, and also to use `QIC` to choose among the competing correlation structures.

```
R> data("mobility", package = "repolr")
R> mod.ar1 <- repolr(mobility ~ age + gender + time * treat, data = mobility,
+    categories = 3, subjects = "subject", times=c(1, 2, 3, 4),
+    corr.mod = "ar1", po.test = TRUE)
R> mod.unif <- update(mod.ar1, corr.mod = "uniform")
R> mod.ind <- update(mod.ar1, corr.mod = "independence")
```

As a preliminary assessment of the appropriateness of the overall model it is often useful to test the assumption of proportionality implicit in Equation 1, which can be done by selecting `po.test = TRUE` which implements the procedure described by Stiger, Barnhart, and Williamson (1999). Results for the independence model, which are output routinely in a call to `summary` if `po.test = TRUE`, suggest that the assumption of proportional-odds cannot be sustained for this working correlation model. The test-statistic, degrees of freedom and p-value are extracted from the fitted model as follows:

```
R> mod.ind$po.test

$po.stat
[1] 13.42185
```

```
$po.df
[1] 5
$po.chi
[1] 0.01973056
```

Similar outputs for `mod.ar1` and `mod.unif` suggest that proportionality cannot be rejected; `mod.ar1$po.test$po.chi = 0.5496093` and `mod.unif$po.test$po.chi = 0.2603108`. Although, the usefulness of such tests have been questioned by Nooraee *et al.* (2014) who reported, based on a small simulation study, that the type I error rate is likely to be inflated. Comparison of `QIC` values indicates a preference for the uniform working correlation model.

```
R> QIC(mod.ar1)$QIC

[1] 1336.297

R> QIC(mod.unif)$QIC

[1] 1335.669
```

Values for the partial derivatives $\partial \log|\mathbf{V}_{\beta_\mathbf{c}}|/\partial\phi$ and $\partial^2 \log|\mathbf{V}_{\beta_\mathbf{c}}|/\partial\phi^2$ at model convergence are given by `mod.unif$grad1 = 0.001001424` and `mod.unif$grad2 = 0.4310958`, indicating that a minimum has been obtained; that is $\partial \log|\mathbf{V}_{\beta_\mathbf{c}}|/\partial\phi \approx 0$ and $\partial^2 \log|\mathbf{V}_{\beta_\mathbf{c}}|/\partial\phi^2 > 0$. Parsons *et al.* (2006) suggested that `mod.unif$grad2` could be used to construct a standard error for α, and illustrated this graphically by plotting estimates of $\log|\mathbf{V}_{\beta_\mathbf{c}}|$ across a range of values for α (or ϕ), including the model estimate `mod.unif$alpha`, for a fixed correlation model. Resultant plots can be informative in showing that a true minimum has indeed been achieved and assessing the sensitivity of the model fit to parametrization of the selected working correlation model. Although we would not recommend routinely undertaking such an analysis, it can occasionally be informative in assessing the evidence available in support of the association parameter α. Code to implement this for the uniform correlation model `mod.unif` is as follows, and the result is shown in Figure 1.

```
R> update.ldetVb <- function(x){log(det(vcov(update(mod.unif,
+    fixed = TRUE, alpha = x))))}
R> phi <- function(x){log(x) - log(1 - x)}
R> range.alpha <- seq(0.25, 0.5, 0.01)
R> data <- lapply(range.alpha, update.ldetVb)
R> plot(x = phi(range.alpha), y = data, type = "b", las = 1, xaxt = "n",
+    xlab = expression(alpha), cex.lab = 1.3, pch = 19,
+    ylab = expression(paste("log|", V[beta][c], "|")))
R> axis(1, at = phi(range.alpha[seq(1, 26, 5)]),
+    labels = range.alpha[seq(1, 26, 5)])
R> opt.approx <- log(det(vcov(mod.unif))) +
+    0.5 * ((phi(mod.unif$alpha) - phi(range.alpha))^2)*mod.unif$grad2
R> lines(x = phi(range.alpha), y = opt.approx)
```

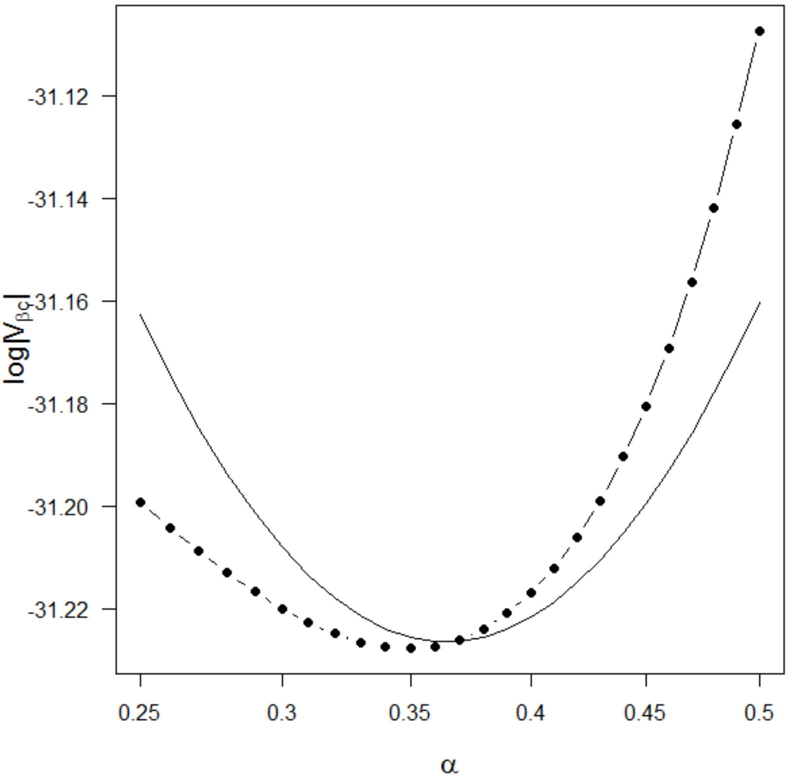

Figure 1: $\log|\mathbf{V}_{\beta_\mathbf{c}}|$ for model `mod.unif` based on (i) fitting a range of models with fixed values for α between 0.25 and 0.5 (●) and (ii) a second order approximation to α around the estimate of 0.365 at convergence (-).

3.3. Polynomial models for cut-point parameters

One of the key features of **repolr** that distinguishes it from other available software tools for model fitting in this setting is the option to model the cut-point parameters using orthogonal polynomial contrasts in the manner originally proposed by Parsons (2013). This option is fully integrated into the main model fitting functions for versions 3.0 and newer of **repolr**. The `begonia` dataset in **repolr** provides an ideal setting in which to explore some of the available options. It is composed of quality scores, for two varieties of begonia potted house plants transported from the growers using three contrasting methods (transport *chains*), collected weekly during 5 weeks in simulated shelf-life conditions. Plants were subjected to a range of conditions in controlled environment rooms, in order to assess how long they would retain key quality attributes. Quality scores were originally made on an ordinal scale from 1 to 10 (highest quality). However, only categories 2 to 9 were used, so these have been re-coded to scale from 1 to 8. In addition to overall quality scores, a range of plant physiological characteristics were also observed. Assuming a working independence correlation model, and modelling the expected strong interactions between temperature, lighting levels and irrigation (`Temp`, `Light` and `Irrig`) and the main effects of weeks in shelf-life (`Week`), transportation (`Chain`) and plant variety (`Variety`) gives the following model for quality score (`Qual`) made at five equally spaced time-points:

```
R> data("begonia", package = "repolr")
R> mod.design <- repolr(Qual ~ Week + Temp * Light * Irrig + Chain + Variety,
+    data = begonia, categories = 8, subjects = "Pot", times = 1:5)
```

Here we are using the conventional default (full) parametrization for the cut-points; i.e., fitting $K-1$ unstructured (but ordered) parameters. It is usually the case that a more parsimonious model than this may be found. Parsons (2013) showed that orthogonal polynomials, of degree $K-2$ or less, provide a simple and easily implemented alternative parametrization for the cut-points. In addition to the argument `poly` that specifies the degree of polynomial required, the only additional argument required for `repolr` is the cut-point spacing (`space`). The spacing vector defines the hypothesised distances between the ordinal score categories, which for the `begonia` data we assume to be equally spaced; i.e., `space = 1:8` . The spacing is the important thing, so the following argument `space = seq(1, 15, 2)` would work equally well and give identical results. Typical circumstances where one might want to deviate from the default settings for the spacing is where certain score categories were not reported in the data. For instance for the `begonia` plant quality scores, imagine that rather than observing ordinal score categories 2 to 9, on original 1 to 10 scale, we had actually observed score categories 1, 3, 5, 6, 7, 8, 9 and 10, then we might want to reflect our expectations of the likely spacings to `space = c(1, 3, 5:10)`. Fitting a third order polynomial model to the `begonia` data is implemented as follows:

```
R> mod.poly3 <- update(mod.design, poly = 3)
```

Models of other orders can be fitted simply by changing the argument `poly` to the appropriate integer value, and overall model fits compared for these models, using `QICu`, to those from other models. Table 2 shows these and parameter estimates for polynomials of orders 1 to 6. The polynomial model of order 6 is exactly equivalent to model `mod.design`; that is the

	Polynomial order					
	1	2	3	4	5	6
QICu	1633.88	1575.69	1567.11	1568.88	1569.07	1571.11
Intercept	-13.19	-14.10	-14.23	-14.27	-14.26	-14.28
poly(cuts, 7)1	365.23	372.96	375.35	376.06	375.89	376.25
poly(cuts, 7)2	-	43.76	45.50	46.03	46.00	46.08
poly(cuts, 7)3	-	-	-9.04	-8.95	-9.04	-9.14
poly(cuts, 7)4	-	-	-	-3.91	-3.88	-4.00
poly(cuts, 7)5	-	-	-	-	3.50	3.68
poly(cuts, 7)6	-	-	-	-	-	2.22

Table 2: Estimated polynomial cut-points and QICu values for begonia dataset.

regression parameters and standard errors will be equal. The only difference between them being the parametrization of the cut-points.

From the values of QICu in Table 2, the third order polynomial appears to provide the best model; i.e., QICu is smallest for this model. It is difficult to compare how the changing parametrization affects the cut-point model, by simply inspecting the parameter estimates in Table 2. The polycuts function converts polynomial coefficients to more conventional, and directly comparable, cut-point estimates. For the third order polynomial model, this is given as follows:

```
R> polycuts(mod.poly3)


repolr: 2016-02-26 version 3.4

Call:
repolr(formula = Qual ~ Week + Temp * Light * Irrig + Chain +
Variety, subjects = "Pot", data = begonia, times = 1:5, categories = 8,
poly = 3)

Coefficients:
coeff    se.robust
Intercept       -14.227    0.651
poly(cuts, 7)1  375.350   15.633
poly(cuts, 7)2   45.505    5.223
poly(cuts, 7)3   -9.041    4.285

Cut-point Estimates:
coeff    se.robust
cuts1|2  -21.095    0.903
cuts2|3  -19.652    0.854
cuts3|4  -17.563    0.783
cuts4|5  -14.967    0.689
cuts5|6  -12.001    0.589
```

```
cuts6|7   -8.802    0.486
cuts7|8   -5.508    0.392

Polynomial Order: 3
```

Model cut-points and standard errors can be extracted directly from fitted models; e.g., for the third order polynomial model these are given by the following:

```
R> cuts.poly <- mod.poly3$poly$polycuts$coeff
R> secuts.poly <- mod.poly3$poly$polycuts$robust.se
```

Figure 2 shows cut-points for polynomial models of order 1 through to 6, together with estimated cut-points and standard errors from model `mod.design` as a comparator. Each of the individual plots in Figure 2 can be constructed using the following code, which for illustration purposes uses the fit from the cubic polynomial model `mod.poly3`.

```
R> plot(x=1:7 + 0.5, y = coef(mod.design)[1:7], type = "p", pch = 19,
+    ylim = c(-23, -5), xlim = c(1, 8), las = 1, ylab = "Cut-points",
+    xlab = "Quality score", xaxt = "n", cex.lab = 1.3, cex.axis = 1.3)
R> axis(1, at = 1:8, labels = 1:8)
R> arrows(x0 = 1:7 + 0.5, x1 = 1:7 + 0.5, y0 = confint(mod.design)[1:7, 1],
+    y1 = confint(mod.design)[1:7, 2], length = 0, angle = 0)
R> lines(x = 1:7 + 0.5, y = cuts.poly, type = "l" , lwd = 1, lty = 1)
R> polygon(y = c(cuts.poly + qnorm(0.975) * secuts.poly,
+    rev(cuts.poly - qnorm(0.975) * secuts.poly)), border = NA,
+    x = c(1:7 + 0.5, 7:1 + 0.5), col = grey(0.7, alpha = 0.5))
```

The other plots in Figure 2 can be obtained simply by replacing `cuts.poly` and `secuts.poly` with output from the appropriate polynomial model. Plots of this type are a useful way to explore the various polynomial models, and for the example `begonia` data show that the third order polynomial provides the most parsimonious model. It is clear from Figure 2 and Table 2 that there is no appreciable improvement in fit for polynomial models with order greater than three, and that the increasing values of `QICu` for the more complex models are due to the penalty term only. The focus here has been the selection of the best model for the cut-point parameters, however, the procedure for finding the best subset of covariates would be undertaken in an analogous manner by comparing `QICu` for competing models and choosing the one with the lowest value. For instance, the effects of plant variety on model fit can be assessed by fitting the model

```
R> mod.newdesign <- update(mod.design,
+    formula = Qual ~ Week + Temp * Light * Irrig + Chain)
```

and comparing `QIC(mod.newdesign)$QICu` to `QIC(mod.design)$QICu`, with the design with the lower value being preferred. As with all such regression fitting tasks, this criterion should be used as a guide when no other scientific knowledge is available for selection.

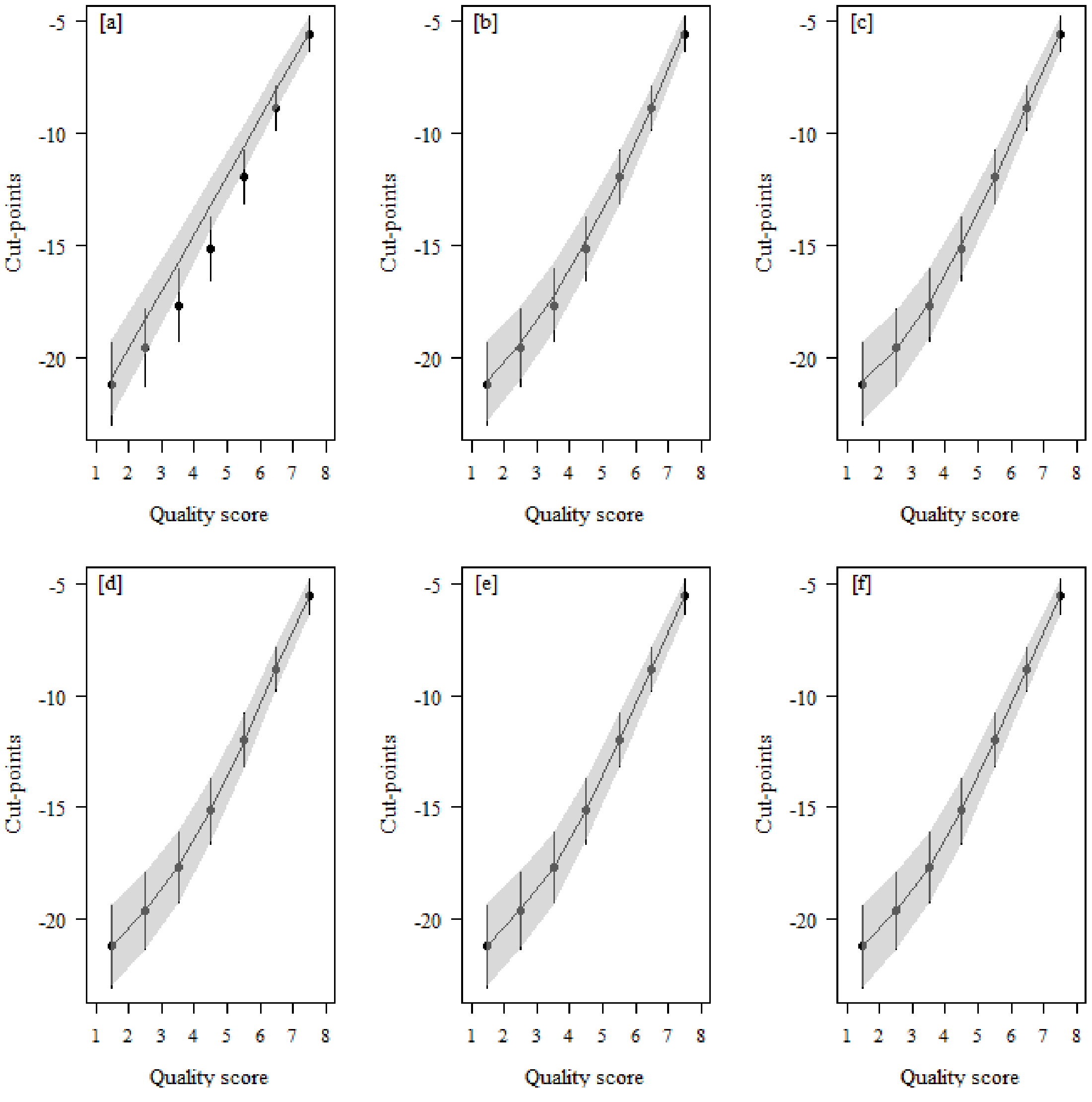

Figure 2: Cut-point models for polynomials of orders [a] 1, [b] 2, [c] 3, [d] 4, [e] 5 and [f] 6. Each plot shows cut-point estimates from the fully parametrized model `mod.design` (•), with bars showing 95% confidence intervals, and the polynomial model (-) with shaded region showing 95% confidence intervals.

4. Conclusions

This paper describes the R package **repolr** that implements a flexible set of functions that allow proportional-odds models to be fitted to repeated ordinal score data using the methods original proposed by Parsons *et al.* (2006) and subsequently extended by Parsons *et al.* (2009) and Parsons (2013). It also provides a detailed description of the syntax required to undertake a range of typical data analysis tasks within **repolr**. Although limited to fitting repeated measures proportional odds logistic regression models, **repolr** has proved to be an efficient and widely used package (Nooraee *et al.* 2014). The current version of **repolr** (v3.4; February 2016) provides the most comprehensive range of model fitting options and due to the fact that core functions are now written in C++, using the packages **Rcpp** and **RcppArmadillo**, it is considerably faster than previous versions for the range of typical models (e.g., $T \leq 5$, $K \leq 7$ and $N \leq 1000$) for which it is currently used. One of the primary appeals of **repolr**, not offered by any other software, is the ability to seamlessly include polynomial models for the cut-point parameters into the conventional modelling framework; see Section 3.3. This feature is particularly useful for long ordinal score scales ($K \geq 10$) which often occur in medical and social sciences, where attempting to estimate many cut-point parameters can be problematic, or where one does not simply wish to dismiss cut-points as nuisance parameters.

References

Agresti A, Natarajan R (2001). "Modeling clustered ordered categorical data: a survey." *International Statistical Review*, **69**(3), 345–371. doi:10.2307/1403450.

Eddelbuettel D (2013). *Seamless R and C++ Integration with **Rcpp***. Springer. ISBN 978-1-4614-6867-7.

Eddelbuettel D, Francois R (2011). "**Rcpp**: Seamless R and C++ Integration." *Journal of Statistical Software*, **40**(8), 1–18. doi:10.18637/jss.v040.i08.

Eddelbuettel D, Sanderson C (2014). "**RcppArmadillo**: Accelerating R with high-performance C++ linear algebra." *Computational Statistics & Data Analysis*, **71**, 1054–1063. doi:10.1016/j.csda.2013.02.0055.

Halekoh U, Høsjsgaard S, Yan J (2006). "The R Package **geepack** for Generalized Estimating Equations." *Journal of Statistical Software*, **15**(2), 1–11. doi:10.18637/jss.v015.i02.

Heagerty P, Zeger S (1996). "Marginal regression models for clustered ordinal measurements." *Journal of the American Statistical Association*, **91**(435), 1024–1036. doi:10.2307/2291722.

Højsgaard S, Halekoh U, Yan J (2016). ***geepack**: Generalized Estimating Equation Package.* R package version 1.2-0.1, URL http://CRAN.R-project.org/package=geepack.

IBM Corp (2013). *IBM SPSS Statistics for Windows, Version 22.0.* Armonk, NY. URL http://www-01.ibm.com/software/analytics/spss/.

Kenward M, Lesaffre E, Molenberghs G (1994). "An application of maximum likelihood and generalized estimating equations to the analysis of ordinal data from a longitudinal study with cases missing at random." *Biometrics*, **50**(4), 945–954. doi:10.2307/2533434.

Liang K, Zeger S (1986). "Longitudinal data analysis using generalized linear models." *Biometrika*, **73**(1), 13–22. doi:10.1093/biomet/73.1.13.

Lipsitz S, Kim K, Zhao L (1994). "Analysis of repeated categorical data using generalized estimating equations." *Statistics in Medicine*, **13**(11), 1149–1163. doi:10.1002/sim.4780131106.

Liu I, Agresti A (2005). "The analysis of ordered categorical data: an overview and survey of recent developments." *Test*, **14**(1), 1–73. doi:10.1007/BF02595397.

McCullagh P (1980). "Regression models for ordinal data." *Journal of the Royal Statistical Society Series B*, **42**(2), 109–142.

McCullagh P, Nelder J (1989). *Generalized Linear Models*. Chapman & Hall.

Nooraee N, Molenberghs G, van den Heuvel E (2014). "GEE for longitudinal ordinal data: Comparing R-geepack, R-multgee, R-repolr, SAS-GENMOD, SPSS-GENLIN." *Computational Statistics & Data Analysis*, **77**, 70–83. doi:10.1016/j.csda.2014.03.009.

Pan W (2001). "Akaike's information criterion in generalized estimating equations." *Biometrics*, **57**(1), 120–125. doi:10.1111/j.0006-341X.2001.00120.x.

Parsons N (2013). "Proportional-odds models for repeated composite and long ordinal outcome scales." *Statistics in Medicine*, **32**(18), 3181–3191. doi:10.1002/sim.5756.

Parsons N (2016). ***repolr**: Repeated Measures Proportional Odds Logistic Regression*. R package version 3.4, URL http://CRAN.R-project.org/package=repolr.

Parsons N, Costa M, Achten J, Stallard N (2009). "Repeated measures proportional odds logistic regression analysis of ordinal score data in the statistical software package R." *Computational Statistics & Data Analysis*, **53**(3), 632–641. doi:10.1016/j.csda.2008.08.004.

Parsons N, Edmondson R, Gilmour S (2006). "A generalized estimating equation method for fitting autocorrelated ordinal score data with an application in horticultural research." *Journal of the Royal Statistical Society C*, **55**(4), 507–524. doi:10.1111/j.1467-9876.2006.00550.x.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Sanderson C (2010). *Armadillo: an open source C++ library for fast prototyping and computationally intensive experiments*. Technical Report, NICTA. URL http://arma.sourceforge.net.

SAS Institute Inc (2015). *SAS/STAT Software, Version 9.3*. Cary, NC. URL http://www.sas.com/.

Stiger T, Barnhart H, Williamson J (1999). "Testing proportionality in the proprtional odds model fitted with GEE." *Statistics in Medicine*, **18**(11), 1419–1433. doi:10.1002/(SICI)1097-0258(19990615)18:11<1419::AID-SIM127>3.0.CO;2-Q.

Touloumis A (2015). "R Package **multgee**: A Generalized Estimating Equations Solver for Multinomial Responses." *Journal of Statistical Software*, **64**(8), 1–14. doi:10.18637/jss.v064.i08.

Touloumis A (2016). ***multgee**: GEE Solver for Correlated Nominal or Ordinal Multinomial Responses*. R package version 1.5.3, URL http://CRAN.R-project.org/package=multgee.

Touloumis A, Agresti A, Kateri M (2013). "GEE for Multinomial Responses Using a Local Odds Ratios Parametrization." *Biometrics*, **69**(3), 633–640. doi:10.1111/biom.12054.

Affiliation:

Nick Parsons
Epidemiology and Statistics Unit
Warwick Medical School
University of Warwick
CV4 7AL, Coventry, UK
E-mail: nick.parsons@warwick.ac.uk
URL: http://www2.warwick.ac.uk/fac/med/staff/nparsons/