carx: an R Package to Estimate Censored Autoregressive Time Series with Exogenous Covariates

by Chao Wang and Kung-Sik Chan

Abstract We implement in the R package **carx** a novel and computationally efficient quasi-likelihood method for estimating a censored autoregressive model with exogenous covariates. The proposed quasi-likelihood method reduces to maximum likelihood estimation in absence of censoring. The **carx** package contains many useful functions for practical data analysis with censored stochastic regression, including functions for outlier detection, model diagnostics, and prediction with censored time series data. We illustrate the capabilities of the **carx** package with simulations and an elaborate real data analysis.

Introduction

Censored data are frequently encountered in diverse fields including environmental monitoring, medicine, economics, and social sciences. Censoring may arise, for example, when a measuring device is subject to some detection limits beyond which the device cannot yield a reliable measurement. Censoring can also occur due to regulations on price change, e.g., limits on maximal intra-daily price change in a stock market.

There exists an extensive literature on regression analysis with censored responses since the pioneering work of Buckley and James (1979). Considerable efforts have also been spent implementing existing methods for estimating various models with censored observations, many of which have been implemented in R. For instance, Henningsen (2010) introduced the **censReg** package (Henningsen, 2013), which covers standard regression models with censored responses including the standard Tobit model (Tobin, 1958), maximum likelihood estimation with cross-sectional data, and random-effects maximum likelihood procedure for panel-data using Gauss-Hermite quadrature. The Tobit model is also implemented in other packages with possibly different estimation methods, including `tobit()` in **AER** (Kleiber and Zeileis, 2008), `cenmle()` in **NADA** (Lee, 2013), `tobit()` in **VGAM** (Yee, 2007), `MCMCtobit()` in **MCMCpack** (Martin et al., 2011), etc.

While there exists an extensive literature on estimating regression models with censored responses and associated software, there are few studies with censored time series response data. More generally, the problem of stochastic regression with both the response and covariates being possibly censored is relatively under-explored. Zeger and Brookmeyer (1986) studied maximum likelihood estimation of a regression model with the errors driven by an autoregressive model of known order $p \geq 0$ ($AR(p)$). Owing to censoring, the "state" vector is generally of variable dimension which can increase rapidly with increasing censoring and AR order. Thus, the maximum likelihood estimation becomes quickly numerically intractable with increasing censoring even for moderately high AR order (Wang and Chan, 2017). Zeger and Brookmeyer (1986) also briefly discussed a pseudo-likelihood approach but did not further develop it. Park et al. (2007) proposed an imputation method to estimate a censored autoregressive moving average (ARMA) process. Their method imputes each censored value by some random value simulated from their conditional distribution given the observed data and the censoring information, and treats the imputed time series as the complete data with which estimation can be done by any standard method. However, they focused on the AR(1) model and relied on simulation studies to demonstrate their method, with no derivation of theoretical properties.

In term of publicly available R packages facilitating estimation with censored time series data, we are aware of only three such packages to date, namely, **cents** (McLeod et al., 2014), **ARCensReg** (Schumacher et al., 2016), and our **carx** (Wang and Chan, 2016). The **cents** package includes the `fitcar1()` function, for fitting an AR(1) model in the presence of censored and/or missing data, and the `cenarma()` function which, according to the authors, implements a quasi-EM algorithm whose M-step is carried out by the `arima()` function and the E-step via the Durbin-Levinson recursions. However, there is little documentation about these functions, rendering it hard to understand and use the **cents** package. The **ARCensReg** package offers similar functionality as our **carx** package. But their estimation is implemented via a stochastic approximation version of the EM (SAEM), which is different from our approach. In addition, it seems to be developed after the **carx**, as a dataset in **carx** is included in **ARCensReg**.

Motivated by the need for developing a computationally efficient method for estimating censored stochastic regression models, Wang and Chan (2017) have recently introduced such a method for

censored autoregressions with exogenous covariates (CARX). The basic idea of our new approach assumes that the score of the complete-data conditional log-likelihood of Y_t^* (the uncensored counterpart of Y_t) given $Y_{t-j}^*, j = 1, \ldots, p$ (and the covariates) has a closed-form expression and so does its expectation given the possibly censored time series $Y_{t-j}, j = 0, \ldots, p$, evaluated at the same set of model parameters. Setting the preceding conditional mean score to zero then provides an unbiased estimating equation for estimating the model. The proposed method reduces to maximum likelihood estimation in the absence of censoring, hence it is referred to as quasi-likelihood estimation. Furthermore, the consistency and asymptotic normality of the quasi-likelihood estimator have been established under some mild regularity conditions (Wang and Chan, 2017).

In this paper we aim to introduce the R package **carx**, in which quasi-likelihood estimation of a CARX model is implemented for the important special case of normal innovations. The main functionality of the package is to provide an intuitive interface with comprehensive documentation to enable the user to estimate the parameters of a CARX model. In addition, some utility functions for model summary, model diagnostics, outlier detection, and prediction with censored time series data are also included in the package.

In addition, we have also implemented a new object class for censored time series, i.e., `cenTS()`. The `cenTS()` class inherits the extensible time series class `xts()` in the R package **xts** (Ryan and Ulrich, 2014). Some functionalities, including plotting and summary, for the `cenTS` class have been implemented. The `cenTS` class is expected to be extended in future and is hoped to be used as a standard data structure for censored time series data.

In the following sections we first elaborate the CARX model and review the quasi-likelihood estimation method, then present the functionality and main functions of the R package **carx** and illustrate the package with data analyses using both simulated and real data examples. Some simulation studies assessing the empirical performance of model selection by minimizing the AIC and the accuracy of the proposed forecasting method and real data example are also reported.

The CARX model

In this section we briefly review the quasi-likelihood method for estimating a CARX model, and refer the reader to Wang and Chan (2017) for details and some theoretical properties of the estimator. We first formulate the problem by specifying the model, then outline the estimation method and discuss some specific topics including model prediction, model diagnostics, and outlier detection.

Model specification

Let $\{Y_t^*\}_{t=0}^{\infty}$ denote a real-valued time series of interest with Y_t^* being not observable if it falls inside a censoring region $C_t \subset R$ which may be time-varying. The censoring region C_t is generally an interval of the form $(-\infty, l_t)$, (u_t, ∞), or (l_t, u_t) corresponding to left, right, and interval censoring, respectively (Huang and Rossini, 1997; Park et al., 2007). (Left and/or right censoring is allowed by **carx** but interval censoring is not yet implemented in **carx**.) In practice, when an observation is censored, it is often recorded as the nearest censoring limit, as it is typically known whether it is left or right censored. The **carx** package assumes the censoring limits to be independent of the underlying process, and automatically treats any missing data as resulting from left censoring with their corresponding censoring limit $l = \infty$.

In practice, Y_t^* is often found to be correlated with some vector covariate, say, X_t. We assume a linear regression relationship between Y_t^* and X_t, with the regression errors following an $AR(p)$ model, where p is the AR order.

The Censored Auto-Regressive model with eXgenous variables (CARX) specifies that the uncensored response $\{Y_t^*\}$ is an autoregressive (AR) process given by

$$\left(Y_t^* - X_t^\intercal \beta\right) - \sum_{j=1}^{p} \psi_j \left(Y_{t-j}^* - X_{t-j}^\intercal \beta\right) = \varepsilon_t, \tag{1}$$

and Y_t^*'s are linked to the observations as follows

$$Y_t = \begin{cases} l_t, & \text{if } Y_t^* \text{ is left censored,} \\ u_t, & \text{if } Y_t^* \text{ is right censored,} \\ Y_t^*, & \text{otherwise,} \end{cases} \tag{2}$$

where β is the vector of regression coefficients, $\psi_i, i = 1, 2, \ldots, p$, are the AR parameters, $\{\varepsilon_t\}$ is an independent and identically distributed (iid) process with mean 0, variance σ_ε^2, and independent of

$\{X_t\}$.

The ε_t's are also known as the innovations in the time series literature. Eqn. (1) is equivalent to the regression model $Y_t^* = X_t^\intercal \beta + \eta_t$, where the regression errors η_t are correlated over time and follow an AR(p) process with the ψ's being the AR coefficients. In the package, the innovations are assumed to be normal although it is shown by Wang and Chan (2017) that the proposed estimation method is robust to mild departure from the normality assumption.

Parameter estimation

Let $\psi = (\psi_1, \cdots, \psi_p)^\intercal$. Throughout, $\theta = (\beta^\intercal, \psi^\intercal, \sigma_\varepsilon)^\intercal$ denotes a generic parameter vector, while θ_0 denotes the true parameter vector. Let $\{(Y_t, X_t)\}_{t=1}^n$ be data generated from the CARX model with parameter θ_0. The quasi-likelihood estimation procedure is motivated by maximum likelihood estimation and leverages on (i) the availability of the closed-form expression of $\ell_t(\theta) = \ell(Y_t^* | Y_{t-j}^*, j = 1, \ldots, p, X_{t-k}, k = 0, \ldots, p; \theta)$ (which holds, for instance, for the case of normal errors as implemented in **carx**) and (ii) $\sum_{t=p+1}^n S_t(\theta) = 0$ is an unbiased estimating equation, where $S_t(\theta)$ is the first derivative of $\ell_t(\theta)$ with respect to θ. Since Y_t^* are unobservable, we replace $S_t(\theta)$ by $E_\theta(S_t(\theta)|Y_{t-k}, X_{t-k}, k = 0, \ldots, p)$ resulting in the following estimating equation:

$$\sum_{t=p+1}^{n} E_\theta(S_t(\theta)|Y_{t-k}, X_{t-k}, k = 0, \ldots, p) = 0. \tag{3}$$

The quasi-likelihood method estimates θ by solving Eq (3). Note that, in the absence of censoring, solving the preceding estimating equation reduces to maximum likelihood estimation, asymptotically.

The following iterative scheme for solving Eq (3) was proposed by Wang and Chan (2017).

Step(1) Initialize the parameter estimate by some consistent estimate, denoted by $\theta^{(0)}$.

Step(2) For each $k = 1, \ldots,$ obtain an update of estimate $\theta^{(k)}$ by

$$\theta^{(k)} = \text{argmax}_\theta \, Q(\theta|\theta^{(k-1)}), \tag{4}$$

where

$$Q(\theta|\theta^{(k-1)}) = \sum_{t=p+1}^{n} Q_t(\theta|\theta^{(k-1)}), \tag{5}$$

$$Q_t(\theta|\theta^{(k-1)}) = \mathrm{E}_{\theta^{(k-1)}} \left[\ell_t(\theta)|Y_{t-k}, X_{t-k}, k = 0, \ldots, p\right]. \tag{6}$$

Step(3) Iterate Step (2) until $\|\theta^{(k)} - \theta^{(k-1)}\|_2 / \|\theta^{(k-1)}\|_2 < \epsilon$ for some positive tolerance $\epsilon \approx 0$. Let $\hat{\theta}$ be the estimate obtained from the last iteration.

The optimization in Step (2) for the case of normal innovations is elaborated in Section 2.4 of Wang and Chan (2017). The value $Q(\hat{\theta}|\hat{\theta})$ evaluated at the convergence of the algorithm will be referred to as the maximum (quasi-)log-likelihood. In the absence of censoring, it reduces to the maximum log-likelihood, hence it will be used to replace the latter in evaluating information criteria such as the Akaike information criterion (AIC) (Konishi and Kitagawa, 2008).

In the **carx** package, the initial value for the preceding iterative algorithm is set to the conditional least squares estimate obtained with the censored data replaced by the corresponding censoring limit, which appears to work well in simulation examples reported in Wang and Chan (2017).

Wang and Chan (2017) proved the consistency and asymptotic normality of the quasi-likelihood estimator under mild regularity conditions. But the asymptotic covariance matrix of the estimator involves two intractable matrices. Consequently, Wang and Chan (2017) proposed to use parametric bootstrap for drawing inference, including estimating the asymptotic covariance matrix and constructing confidence intervals of the unknown parameters.

Model prediction

It is of practical interest to predict the future values Y_{n+h}^* given the observations $\{(Y_t, X_t)\}_{t=1}^n$, where $h = 1, 2, \ldots, H$ and H is some fixed upper bound, for instance, $H = 14$ for bi-weekly forecast, assuming the data are sampled daily. This is generally a non-trivial problem in the presence of censoring, and can be handled by Monte Carlo simulation for its solution. Since X is an exogenous process, we consider the simple case of the prediction problem conditioned on the given future covariate values $\{X_{t+h}\}_{h=1}^H$. We also assume normality of ε_t and known parameter θ_0, although the following discussion can be

readily extended to non-normal innovations. Relaxation of these assumptions will be discussed at the end of this subsection. The prediction problem is equivalent to finding the conditional distribution

$$\begin{aligned}\mathcal{D}_{n,h} &= \mathcal{D}(Y^*_{n+h} | \{X_{n+i}\}_{i=1}^h, \{(Y_t, X_t)\}_{t=1}^n) \\ &= \mathcal{D}(Y^*_{n+h} | \{X_{n+i}\}_{i=1}^h, \{(Y_t, X_t)\}_{t=\tau}^n),\end{aligned}$$

where $\tau = \max\left(\{1\} \cup \left\{u : 1 \le u \le n-p+1, \text{ and none of } \{Y_t\}_{t=u}^{u+p-1} \text{ is censored}\right\}\right)$, due to the autoregressive nature of the regression errors $\eta_t = Y_t^* - X_t^\intercal \beta$ (Zeger and Brookmeyer, 1986).

There are two cases. Case 1: $\tau = n-p+1$, i.e., the most recent p Y_t's are uncensored so that the prediction problem admits a closed-form solution which is well-known; see, e.g., Cryer and Chan (2008, Chapter 9). Specifically, for any $h = 1, \ldots, H$, $\mathcal{D}_{n,h}$ is a normal distribution whose mean serves as the point predictor denoted by $\hat{Y}^*_{n+h}$ that can be recursively computed as follows: $\hat{Y}^*_{n+h} = X_{n+h}^\intercal \beta + \hat{\eta}_{n+h}$, with $\hat{\eta}_t = \sum_{l=1}^p \psi_l \hat{\eta}_{t-l}$ for $t > n$, and $\hat{\eta}_t = Y_t - X_t^\intercal \beta$ if $t \le n$. The prediction error, denoted by $\epsilon_{n+h} = Y_{n+h} - \hat{Y}^*_{n+h}$, can be written as $\epsilon_{n+h} = \varepsilon_{n+h} + \sum_{l=1}^p \psi_l \epsilon_{n+h-l} = \sum_{i=0}^h \omega_{h,i} \varepsilon_{n+h-i}$, where the coefficients $\omega_{h,i}$ can be recursively calculated by making use of the preceding identity and the initial condition $\omega_{h,0} = 1$. The prediction variance is given by $(\epsilon_{n+h}) = {\sigma_\varepsilon}^2 \sum_{i=0}^h \omega_{h,i}^2$.

We now consider Case 2: $\tau < n-p+1$. Then $\mathcal{D}_{n,h}$ is a truncated multivariate normal distribution. Although the first and second moments of $\mathcal{D}_{n,h}$ admit closed-form solutions (Tallis, 1961; Genz et al., 2014), they are not useful for constructing predictive intervals as the predictive distributions are non-normal. Thus, we propose to use a sampling approach to estimate any interesting characteristic of the predictive distribution of Y^*_{n+h}. First, note that the regression errors $\{\eta_t = Y_t^* - X_t^\intercal \beta\}_{t=\tau}^n$ are jointly normal. Let η_c and η_o be the sub-vectors of $\eta_{\tau:n}$ such that the corresponding elements of $Y_{\tau:n}$ are censored and observed, respectively. Then given $\{(Y_t, X_t)\}_{t=\tau}^n$, η_c follows a truncated multivariate normal distribution, whose realizations can be readily simulated, and hence we can simulate $Y_t^* = X_t^\intercal \beta + \eta_t, \tau \le t \le n$. Then the realizations of $Y^*_{n+h}, h = 1, \ldots, H$ can be drawn from the multivariate normal predictive distribution stated in Case 1. Predictive intervals of Y^*_{n+h} can then be approximately constructed from a random sample from the predictive distribution of Y^*_{n+h}, using the percentile method.

Note that the proposed predictive scheme is conditional on the future covariate values $\{X_t\}_{h=1}^H$, which, in general, are non-deterministic. Extension to the case of stochastic $\{X_{t+h}\}_{h=1}^H$ is straightforward, provided that its stochastic generating mechanism is known, as drawing a realization from the predictive distribution of Y^*_{n+h} can be done in two steps. Step 1 consists of drawing a realization $\{x_{t+h}\}_{h=1}^H$, followed by drawing a future realization for Y^*_{n+h} given the data and $\{x_{t+h}\}_{h=1}^H$. In practice, θ_0 is unknown and it can be replaced by the quasi-likelihood estimator or a parametric bootstrap approach which can be readily implemented to incorporate parametric uncertainty in the prediction.

Model diagnostics

A main task in model diagnostics consists of checking whether or not the data are consistent with the model assumption that the innovations are independent and identically normally distributed of zero mean and constant variance. In the presence of censoring, how to define the residuals is unclear. For the simple case when Y_t^* is observed, the corresponding residual is universally defined as $Y_t^* - \hat{Y}^*_{t|t-1}$, where $\hat{Y}^*_{t|t-1}$ is the mean of $\mathcal{D}_{t-1,1}$, evaluated at the parameter estimate. In the case of censoring so that some Y_t^*s are unobserved, Wang and Chan (2017) advocated the use of the simulated residuals (Gourieroux et al., 1987) for model diagnostics. The simulated residuals are constructed as follows. First, impute each unobserved Y_t^* by a realization from the conditional distribution $\mathcal{D}(Y_t^* | \{(Y_s, X_s)\}_{s=1}^t)$, evaluated at the parameter estimate. Then, refit the model with $\{(Y_t^*, X_t)\}$ so obtained, via conditional maximum likelihood; the residuals from the latter model are the simulated residuals $\hat{\varepsilon}_t$. Let the corresponding parameter estimate of θ be $\tilde{\theta}$. The corresponding (simulated) partial residuals for the X's, i.e., $X_t^\intercal \tilde{\beta} + \hat{\varepsilon}_t$, can be used to assess the relationship between Y and X, after adjusting for the autoregressive errors.

A simulation study reported in Wang and Chan (2017) suggests that the asymptotic null distribution of the Ljung-Box test statistic, for checking residual autocorrelations, based on the simulated residuals is the same as that based on the uncensored data. Thus, standard diagnostic tools for residual analysis may be applicable with the simulated residuals.

Outlier detection

Real data are often marred by outliers. An outlier in a time series may result from a perturbation inducing an unknown shift in an observation or an innovation, resulting in the so-called additive or

innovative outlier, respectively (Cryer and Chan, 2008). An innovative outlier (IO) may mask as a contiguous block of additive outliers (AO). Since it is harder to detect IOs in censored time series, we focus on detecting AOs with a new method for doing so in censored time series.

As the number of outliers and their locations are generally unknown, outlier detection is carried out one by one and iteratively. The procedure begins with an outlier-free CARX model. Then we check for the presence of additive outliers by a method to be described below. If an outlier is detected at time t_o, the covariate X will be augmented with the indicator variable I_{t_o} which equals 1 if $t = t_o$, and 0 otherwise. The augmented CARX model is then fitted, with which outlier detection is repeated until no more outliers are found.

More specifically, we describe a method to detect any remaining additive outliers given the data and a CARX model. For the sake of fast computation, we consider the predictive distribution of Y_t given the information from $t-p$ to t, i.e., $\tilde{\mathcal{D}}_t := \mathcal{D}\left(Y_t^*|X_t, \left\{(Y_{t-j}, X_{t-j}), j = 1, \ldots, p\right\}\right)$. Let $P_{\tilde{\mathcal{D}}_t}(E)$ be the probability of the event E evaluated with distribution $\tilde{\mathcal{D}}_t$ and n the sample size, for each $t = p+1, \cdots, n$, we calculate the following probability p_t.

$$p_t = \begin{cases} P_{\tilde{\mathcal{D}}_t}(Y_t > u_t), & \text{if } Y_t^* \text{ is right censored,} \\ P_{\tilde{\mathcal{D}}_t}(Y_t < l_t), & \text{if } Y_t^* \text{ is left censored,} \\ \min\{P_{\tilde{\mathcal{D}}_t}(Y_t > y_t), P_{\tilde{\mathcal{D}}_t}(Y_t < y_t)\}, & \text{otherwise.} \end{cases}$$

Let $t_o = \text{argmin}_{t=1,\cdots,n} p_t$, and the response at t_o is declared as an AO if $p_{t_o} < 0.025/n$, where the Bonferroni inequality is used to limit the family error rate to not exceed 5% (Cryer and Chan, 2008); otherwise, it is deemed that there are no remaining outliers.

The carx package

In this section we present the R package **carx** in which the estimation, prediction, and diagnostics procedures discussed in previous section are implemented, assuming the normality of ε_t. For more detail, see the documentation of the package. Examples will be given in the next section.

A class for censored time series

First, let us introduce a class `cenTS` designed to encapsulate a censored time series with its observed values as well as the left (lower) and right (upper) censoring limits. The `cenTS` inherits the extensible time series class `xts` in the R package **xts**. A `cenTS` object can be constructed by the following function call.

```
cenTS(value, order.by, lcl = NULL, ucl= NULL, value.name = "value", ...)
```

Note that the `value` (whose name can be set in `value.name`) and `order.by` denote the observed values and their corresponding indices respectively, and `lcl` and `ucl` denote the left (lower) and right (upper) censoring limits respectively. Other time series variables to be included as covariates in the regression can be supplied via additional arguments.

A `cenTS` object can be inspected by the `print()` and `plot()` methods. Any covariate time series can be retrieved by the `xreg()` method.

The default estimation method

The foremost function is the method for the `S3` class `carx`, `carx()`, whose signature is the following.

```
    carx(formula, data = list(),  p = 1,
prmtrX = NULL, prmtrAR = NULL, sigma = NULL,
y.na.action = c("skip","as.censored"), addMu = TRUE,
tol = 1e-4, max.iter = 500,
CI.compute = FALSE, CI.level = 0.95,
b = 1000,b.robust = FALSE,b.show.progress = FALSE,
init.method = c("biased","consistent"),
cenTS = NULL, verbose = FALSE,...)
```

The `carx()` method provides a simple-to-use interface for the user to input a formula, a data set, and other arguments to estimate a CARX model.

The `carx()` method returns a `carx` object which stores the supplied data, the quasi-likelihood coefficient estimates, as well as other information. It allows many optional arguments to control the function behavior. The main arguments are listed below:

- `formula` is a formula representing the regression part of the model, such as `y ~ x1 + x2`.
- `data` denotes a `data.frame` which includes the following:
  - The response variable with variable name identified by the supplied formula.
  - Any covariate(s) with variable name(s) identified by the supplied formula.
  - A vector with name `ci` whose components take values from $\{-1, 0, 1\}$, where -1 (0,1) indicates that the corresponding element in the response variable is left-censored (not censored, right censored).
  - `lcl` representing the vector of left (lower) censoring limits. If not present, indicating no lower limit.
  - `ucl` representing the vector of right (upper) censoring limits. If not present, indicating no upper limit.
- `p` denotes the autoregressive order of the regression errors, default = 1.

The above arguments supply the data structure including the censoring information, and specify the CARX model to be estimated. Although the function contains many optional arguments for fine-tuning the fitting algorithm and obtaining more information about the fitted model such as confidence intervals, we merely discuss the following two arguments:

- `prmtrX`, `prmtrAR`, and `sigma` are used to specify the initial values of the regression coefficients β, the autoregressive parameters Ψ, and the innovation standard deviation σ_ε, respectively.
- `y.na.action` is a string indicating how to handle missing (`NA`) values in `y`. If it is set to `"skip"` (default), cases containing a missing value will be skipped, so that the estimating equation of future cases will be conditional on the most recent `p` complete cases after the skipped case. For `"as.censored"`, the `y` value will be treated as left-censored with the left (lower) censoring limit replaced by positive infinity. The user may choose to use `skip` if there exist few long gaps in the response. Use `"as.censored"` in the presence of numerous, non-contiguous missing values in `y`. Note that the presence of any missing values in `x` will automatically hard-code `y.na.action` to be `"skip"`.

Other methods

As `carx` is an `S3` class, some generic methods have been implemented so that the estimation function can be easily called for practical use and more information about the model fitting can be easily extracted.

The function `print()` simply returns a plain output of the fitted model, while the `summary()` function provides a more elaborate summary of the fitted model including the estimates, their standard errors, 95% confidence limits and p-values, based on parametric bootstrap, for each model parameter, if `CI.compute = TRUE`. The model parameters can be conveniently extracted by the function `coef()`, which returns all coefficient estimates except that the error (innovation) standard deviation is returned as the `sigma` component of the list returned by `carx()`. `logLik()` returns the maximum (quasi-)log-likelihood $\sum_{t=p+1}^{n} \mathrm{E}_{\hat{\theta}} \left[\ell(Y_t^* | \mathcal{F}_t^*; \hat{\theta}) | \mathcal{G}_t\right]$, which can be used in lieu of the intractable maximum log-likelihood. For instance, the function `AIC()` computes the AIC of the model with the maximum log-likelihood replaced by maximum (quasi-)log-likelihood.

There are some other useful functions in the package. The method `plot()` draws the time plot of the censored response time series, superimposed with the fitted values (1-step-ahead predictors) from the supplied CARX model. The function `predict()` computes the multi-step-ahead point predictors and their associated prediction limits, based on a given model and future values of the covariates supplied by the user. The function `fitted()` returns the fitted values by calling the `predict` method. The function `residuals()` returns the simulated residuals of the fitted model. The outlier detection method discussed in Section 2.2.5 is implemented by the method `outlier()`. Model diagnostics based on the simulated residuals are visualized by the method `tsdiag()`, which consists of four subplots: the time plot of standardized simulated residuals, the residuals versus fitted values plot, the residual autocorrelation function plot, and the plot of the p-values of the Ljung-Box test statistics, for testing no residual autocorrelations.

Using the package

In this section we illustrate the various functions of the package through two examples, the first one is a simulated data set and the second a real data set. Note that an extensive simulation study about the performance of the proposed estimation method and some model diagnostics can be found in Wang

and Chan (2017) which shows the robustness of the proposed estimation method to slight departure from the normality assumption of the innovations. We first load the carx package by the following command.

```
> library(carx)
```

A function to simulate data

To begin, we introduce the function carxSimCenTS() for simulating data from a CARX model, whose signature and default values of arguments are shown below.

```
carxSimCenTS(nObs = 200, prmtrAR = c(-0.28,0.25),
  prmtrX = c(0.2,0.4), sigma = 0.60, lcl = -1, ucl = 1, x = NULL,
  seed = NULL, value.name = 'y', end.date = Sys.date())
```

The carxSimCenTS() function generates a simulated cenTS time series of length nObs, with the AR parameters $(\psi_i, i = 1, \ldots, p)$ supplied through the argument prmtrAR, the regression coefficients through prmtrX, and innovation standard deviation through sigma, the lower and upper censoring limits through lcl and ucl respectively. The regressors can be supplied via x, which, if is NULL, will be generated as independent standard normal variables. The user can also specify the seed of the random number generator by seed for ensuring repeatability. As carxSimCenTS() encapsulates the simulated data into a cenTS object, the construction of which need a time/date-based index. The default treats the data as daily observations, with the end date specified by end.date. The user can set the name of the censored time series via value.name but the names of the regressors are hard-coded as X1, X2, etc. There is another function carxSim(), which returns a data.frame consisting of y, x, lcl, ucl and ci. We will mainly use the carxSimCenTS() function for simulation as it encapsulates the data as a cenTS object.

A step-by-step illustration with a simulated series

We first simulate a cenTS series, using the carxSimCenTS() function with essentially the default setting, i.e., simulate interval-censored data from a regression model with a 2-dimensional covariate comprising independent standard normal components whose regression coefficients are 0.2 and 0.4, and AR(2) regression noise terms with the AR coefficients being $\psi_1 = -0.28, \psi_2 = 0.25$; the data are then censored unless they fall inside the interval $(-1, 1)$.

```
> datSim <- carxSimCenTS(seed = 0,end.date = as.Date('2015-08-01'))
```

A glimpse of the last few data cases of the series is instructive.

```
> tail(datSim)

                y lcl ucl ci      X1     X2
2015-07-26  1.000  -1   1  1 -0.5466  0.288
2015-07-27 -0.321  -1   1  0 -1.6887 -1.505
2015-07-28 -0.259  -1   1  0 -1.5724  1.519
2015-07-29  0.386  -1   1  0 -0.4050  0.367
2015-07-30  0.282  -1   1  0  0.3193  1.700
2015-07-31  0.181  -1   1  0  0.0404  0.644

Censoring rate: 0.205
```

The simulated series can be readily visualized using the plot function (see Fig. 1).

```
> plot(datSim)
```

Then the parameters can be estimated by the carx() method, with the fitted model saved in the object named modelSim.

```
> modelSim <- carx(y ~ X1 + X2 - 1,data = datSim, p = 2, CI.compute = TRUE)
```

Note that -1 in the formula specifies no intercept in the regression. Information about the fitted model can be obtained directly by typing the variable name modelSim.

```
> modelSim
```

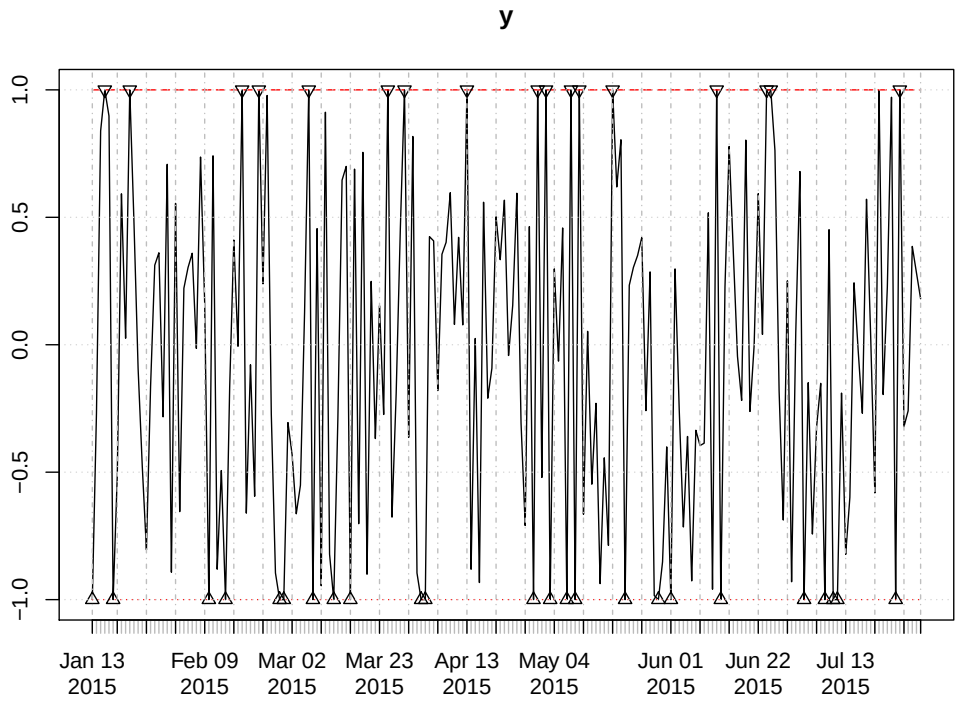

Figure 1: Time plot of the simulated `cenTS` series. The observed responses are connected as a solid black line, and the lower and upper censoring limits drawn as red dotted and dashed line with censored observations marked by triangles pointing up and down, respectively.

```
Call:
carx.formula(formula = y ~ X1 + X2 - 1, data = datSim, p = 2,
    CI.compute = T)

Coefficients:
    X1     X2    AR1    AR2
 0.203  0.460 -0.234  0.279

Residual (innovation) standard deviation:
[1] 0.548

Censoring rate:
[1] 0.205

Sample size:
[1] 200

Number of parameters:
[1] 5

Quasi-log-likelihood:
[1] 20.1

AIC:
[1] -30.1

Confidence interval:
       2.50%  97.50%
X1     0.131  0.2766
X2     0.383  0.5446
AR1   -0.390 -0.0919
```

```
AR2    0.123  0.4066
sigma  0.483  0.6122

Variance-covariance matrix:
             X1       X2      AR1       AR2    sigma
X1     1.41e-03 1.98e-05 5.86e-05 -1.49e-04 9.95e-05
X2     1.98e-05 1.73e-03 1.28e-04  9.08e-05 2.45e-04
AR1    5.86e-05 1.28e-04 5.76e-03  2.08e-03 1.32e-04
AR2   -1.49e-04 9.08e-05 2.08e-03  5.15e-03 5.88e-05
sigma  9.95e-05 2.45e-04 1.32e-04  5.88e-05 1.05e-03
N.B.: Confidence intervals and variance-covariance matrix
are based on 1000 bootstrap samples.
```

A summary of the fitted model can be obtained by running the `summary()` function.

```
> summary(modelSim)

Call:
carx.formula(formula = y ~ X1 + X2 - 1, data = datSim, p = 2,
    CI.compute = T)


Coefficients:
      Estimate  StdErr lowerCI upperCI p.value
X1      0.2025  0.0375  0.1314    0.28  <2e-16 ***
X2      0.4602  0.0416  0.3826    0.54  <2e-16 ***
AR1    -0.2336  0.0759 -0.3903   -0.09   0.002 **
AR2     0.2792  0.0717  0.1232    0.41  <2e-16 ***
sigma   0.2025  0.0324  0.4834    0.61  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

AIC:
[1] -30.1
```

Although it can be shown that the quasi-likelihood estimator is asymptotically normal under some regularity conditions (Wang and Chan, 2017), the asymptotic variance-covariance matrix is intractable so it is computed via parametric bootstrap. The summary function prints out the coefficient estimates and innovation standard deviation estimate, together with their estimated (bootstrap) standard errors, and lower and upper 95% confidence limits. Note that the bootstrap computation time increases almost linearly with the bootstrap replicate size; the default is 1000. More specific information can be easily obtained by invoking various methods. For instance, `logLik` returns the quasi-log-likelihood of the data, `coef` returns the coefficients of the model, and the standard deviation of ε can be obtained by `modelSim$sigma`.

```
> logLik(modelSim)

[1] 20.1
attr(,"class")
[1] "logLik.carx"

> coef(modelSim)

    X1     X2    AR1    AR2
 0.203  0.460 -0.234  0.279

> modelSim$sigma

[1] 0.548
```

The `plot()` function provides a visual check of how the fitted values track the data, with the censoring limits superimposed on the diagram, see Fig. 2.

```
> plot(modelSim)
```

Model diagnostics are facilitated by the `tsdiag()` function which is similar to the `tsdiag()` function in **TSA** package (Chan and Ripley, 2012). The `tsdiag()` function generates a plot of 4 sub-figures,

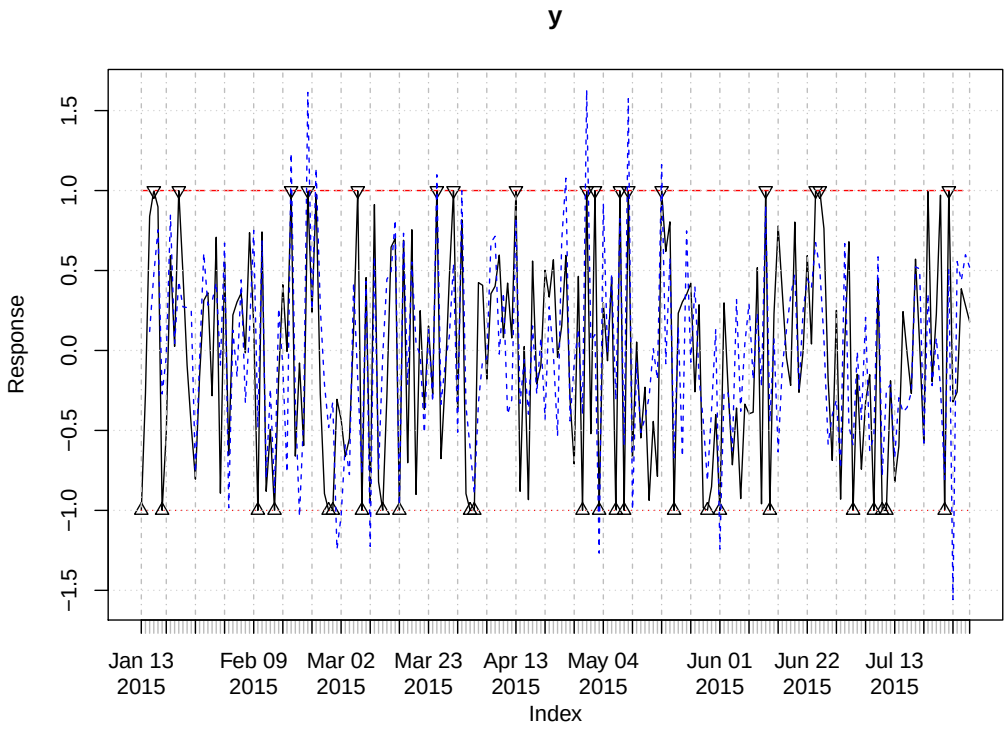

Figure 2: Time plot of the raw data and fitted values from the CARX model. The observed responses are connected as a solid black line, and the lower and upper censoring limits drawn as red dotted and dashed line with censored observations marked by triangles pointing up and down, respectively. The fitted values are connected as a blue dashed line.

namely, the time plot of the simulated residuals which is useful for visually checking the presence of residual temporal patterns and/or outliers, the simulated residuals versus fitted values plot which is useful for checking the adequacy of the linear regression model assumptiom, the residual autocorrelation function (ACF) plot that quantifies the residual correlations, and the plot of the p-values of the Ljung-Box tests for the presence of residual autocorrelation. The following command generates the diagnostics plot for the model fitted to the simulated data.

```
> tsdiag(modelSim)
```

The uppermost diagram in Fig. 3 shows no apparent residual temporal patterns, which is also confirmed by the fact that none of the examined residual autocorrelations in the second sub-figure from the bottom are significant and that the bottom sub-figure shows that all p-values of the Ljung-Box test statistics based on the first k lags of residual autocorrelations are larger than 5% for all allowable $k \leq 23$. Moreover, the second sub-figure from the top shows that the linear regression assumption is justifiable and so is the constant innovation variance assumption. Hence, we can conclude that the model is correctly specified, as it should be, and it provides a good fit to the data.

A real data application

In this example we utilize the package to analyze the change of total phosphorus (P) concentrations in river water. Phosphorus is a major nutrient in river water, of which an excessive amount can result in environmental problem such as eutrophication. Phosphorus concentration in many rivers in Iowa has been monitored under the ambient water quality program conducted by the Iowa Department of Natural Resources (Libra et al., 2004). An analysis of the change of P concentration has been reported by Wang et al. (2016). Here we illustrate the analysis for a particular data set from an ambient site located in the West Fork Cedar River at Finchford, with the data available in a `cenTS` object named `pts` in the **carx** package.

The P concentrations (in mg/l) were left-censored whenever they fell below certain time-varying detection limits, resulting in a censoring rate of 12.6%. The data were collected monthly from October 1998 to October 2013, but data collection was suspended between September 2008 to March 2009,

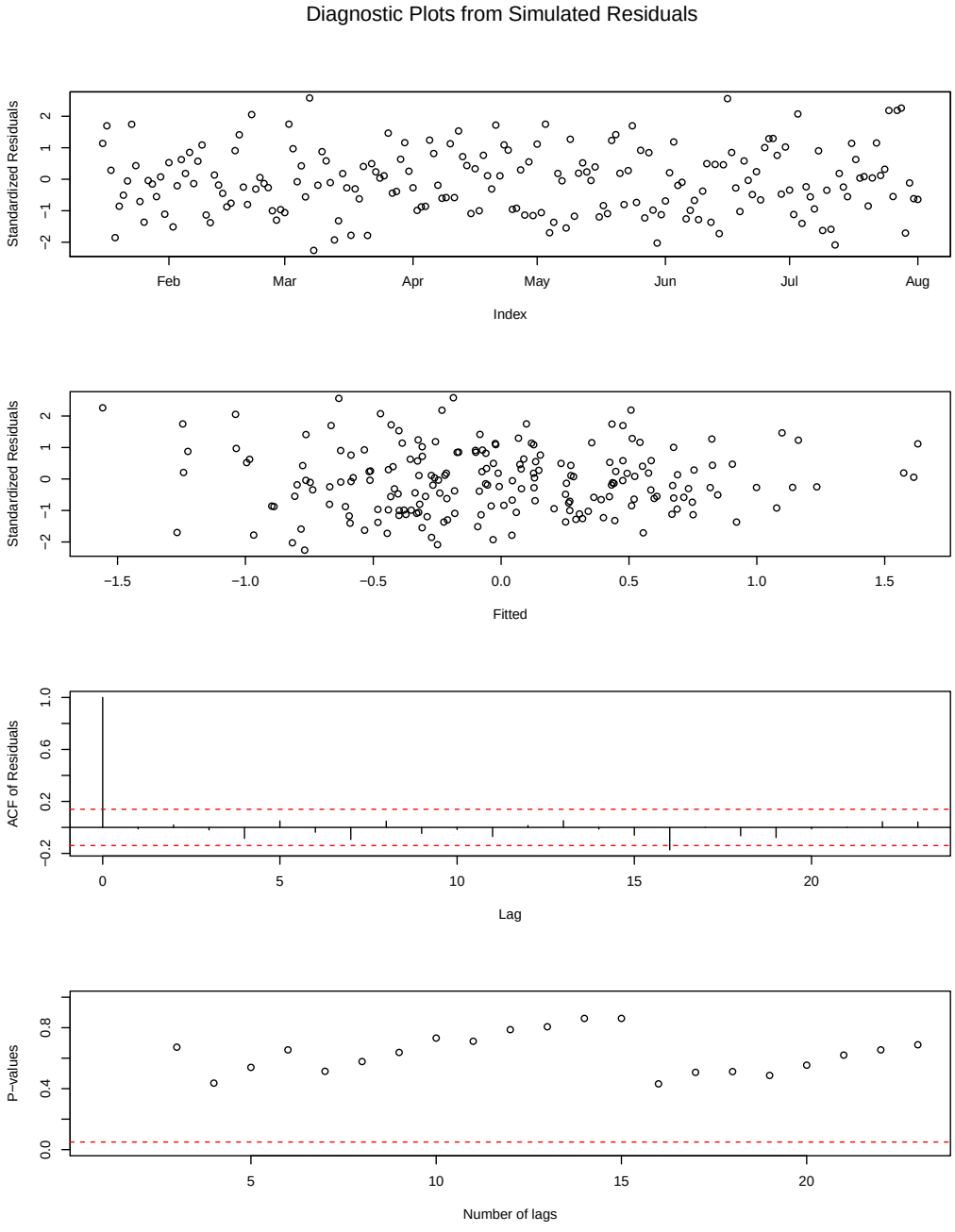

Figure 3: Diagnostic plots based on simulated residuals.

owing to lack of funding. In the data set, there are serveral variables.

```
> names(pts)

[1] "logP"     "lcl"      "ci"       "tInMonth" "logQ"     "season"
```

The variable `logP` consists of the logarithmic P, `lcl` the corresponding censoring limits, `ci` the indicator variable of censoring, `tInMonth` is the time index (in month), `logQ` is the corresponding logarithmic water discharge data (Q) measured in cf/s, obtained from the U.S. Geological Survey (USGS), and season indicates to which `season` the month index belongs (see below for further details). P is generally correlated with the water discharge (Schilling et al., 2010). We will explore the relationship between P and Q. See Fig. 4 for the time plots of P, Q, and the historical censoring limits.

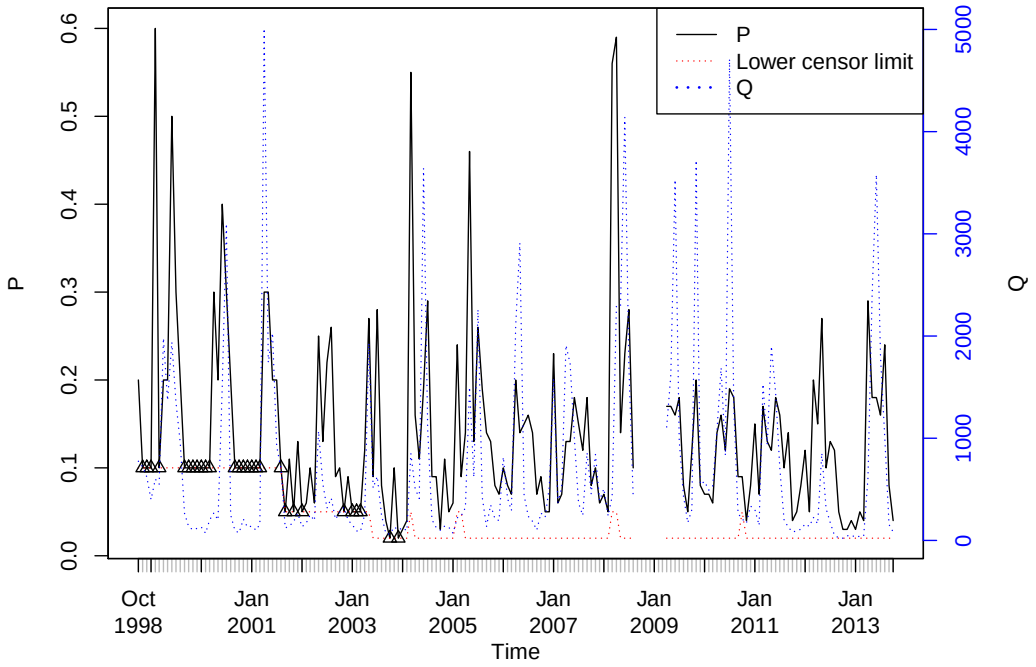

Figure 4: Time series plots of P (black line, scale shown on the left vertical axis), Q (blue line, scale shown on the right vertical axis) and the censor limits l_t (red line, in the same scale as that of P). Censored observations are marked by triangles.

It is also conjectured that the association between the logP and logQ may be seasonal. The variable `season` in `pts` is constructed to denote the quarter of the month with Quarter 1 consists of the first three months, namely, January, February, and March; Quarter 2 comprises the next three months, and so on. Figure 5 illustrates the seasonal relationship between `logP` and `logQ`. It is of interest whether there exists a linear trend in the logarithmic P. Preliminary analysis (not reported here) suggests the presence of significant autocorrelation in the regression errors. Thus, the general model takes the following form

$$\log(P_t) = \beta_1 t + f(\log(Q_t)) + \eta_t,$$

where f is some linear function that may be seasonal in the intercept and/or seasonal in the coefficient of `logQ`, and η_t follows an AR process.

Note that we need to determine whether the intercept and/or the regression coefficient are seasonal, and whether to include in the model a time trend, resulting in 8 combinations. Moreover, the AR order for the regression errors has to be specified. Assuming the maximal AR order to be m, we have to select among $8 \times m$ models, which can be done by selecting the best model that achieves the smallest AIC. Model selection by AIC is automated by the function `carxSelect()`. Here, the maximal AR order is 3.

```
> arOrder <- 3
```

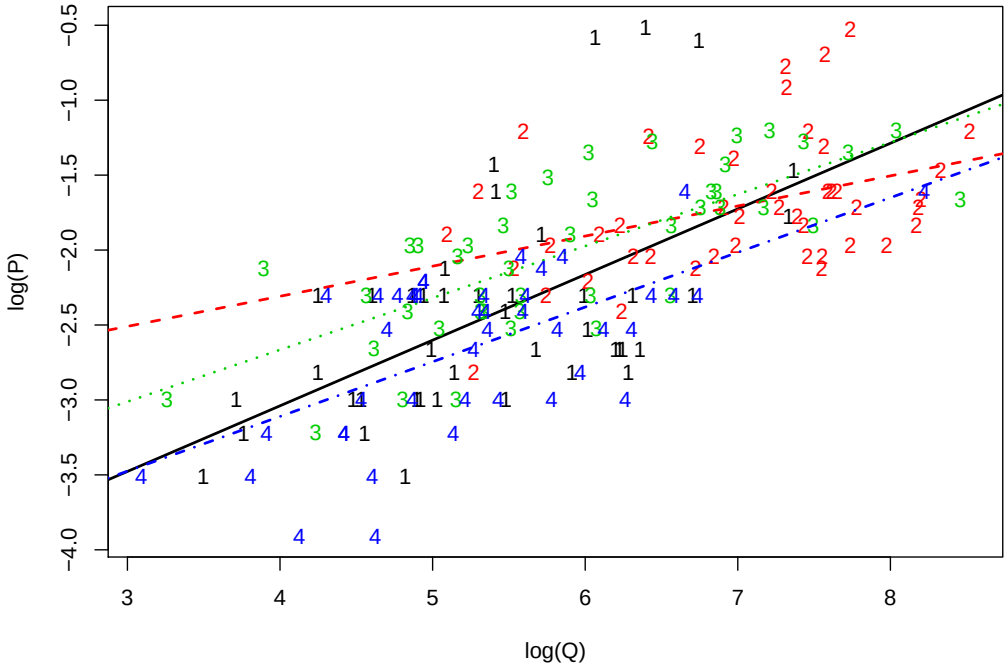

Figure 5: Scatter plot of `logP` versus `logQ`. The data are labeled by different numbers (1 for Quarter 1 and so on) and colors (black, red, green, blue for Quarter 1, 2, 3 and 4, respectively). Least squares quaterly regression lines (solid, dashed, dotted, and dash dotted) are superimposed with the same color as the data points.

The list of models, named M1 to M8, is specified in the following code.

```
> s1 <- logP ~ logQ
> s2 <- logP ~ tInMonth + logQ
> s3 <- logP ~ logQ:as.factor(season)
> s4 <- logP ~ tInMonth + logQ:as.factor(season)
> s5 <- logP ~ as.factor(season) + logQ - 1
> s6 <- logP ~ tInMonth + as.factor(season) + logQ - 1
> s7 <- logP ~ as.factor(season) + logQ:as.factor(season) - 1
> s8 <- logP ~ tInMonth + as.factor(season) +  logQ:as.factor(season) - 1
> fmls <- c(s1,s2,s3,s4,s5,s6,s7,s8)
> names(fmls) <- paste0('M',seq(1,8))
```

The model selection is performed by invoking the function `carxSelect()` which has two required arguments: a list of formulas and the maximal AR orders, plus an optional argument `detect.outlier`, which by default is `TRUE`, indicates whether outliers should be detected in the model. The function `carxSelect()` returns a `carx` object comprising an additional element `selectionInfo` which is a list containing more information about the selection result, including `aicMat` which is a matrix whose (i, j)th element is the AIC of the model represented by the i^{th} formula in the list and AR order j.

For the purpose of illustrating the prediction by the `predict()` method, we use all data up to the end of 2012 for model selection and fitting, and then check the model prediction against the observed data in 2013.

```
> cs <- carxSelect(fmls, arOrder, data = pts['1998/2012'],
+                  detect.outlier = TRUE, CI.compute = TRUE)

> print(round(cs$selectionInfo$aicMat,1))

     AR1   AR2   AR3
M1 -41.8 -39.6 -38.2
M2 -39.7 -38.1 -36.8
M3 -51.2 -47.5 -45.1
```

```
M4 -49.4 -45.9 -43.6
M5 -53.2 -49.6 -47.1
M6 -51.5 -48.2 -45.8
M7 -53.9 -50.8 -47.6
M8 -52.8 -49.8 -46.8
```

A summary of the model fit is shown below.

```
> summary(cs)

Call:
carx.formula(formula = formula(m0), data = m0$data, p = m0$p,
    CI.compute = ..1)


Coefficients:
                          Estimate    StdErr   lowerCI upperCI p.value
as.factor(season)1      -6.053239  0.633190 -7.325537 -4.9191  <2e-16 ***
as.factor(season)2      -3.455734  0.612764 -4.696565 -2.2607  <2e-16 ***
as.factor(season)3      -4.235837  0.414149 -5.028568 -3.4297  <2e-16 ***
as.factor(season)4      -4.854407  0.476015 -5.764154 -3.9324  <2e-16 ***
as.factor(season)1:logQ  0.633582  0.111002  0.436431  0.8566  <2e-16 ***
as.factor(season)2:logQ  0.248797  0.086893  0.073705  0.4247   0.006 **
as.factor(season)3:logQ  0.373538  0.068555  0.235532  0.5053  <2e-16 ***
as.factor(season)4:logQ  0.389721  0.087467  0.217672  0.5584  <2e-16 ***
AR1                      0.075072  0.090671 -0.137839  0.2227   0.596
sigma                    0.482897  0.028770  0.412907  0.5265  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

AIC:
[1] -53.85639
```

The fitted model can be visualized by calling the `plot()` function, which is shown in Figure 6. The fitted values appear to track the data well.

```
> plot(cs)
```

We examine the goodness-of-fit of the fitted model via the `tsdiag()` function which generates 4 diagnostic plots in Fig. 7. These plots indicate that the fitted model provides good fit to the data.

```
> tsdiag(cs)
```

The selected model can be interpreted as follows. The linear trend was not selected, suggesting no significant long-term change in the P concentrations. The intercept and the regression coefficient of logQ were seasonal. The regression errors appeared to be mildly auto-correlated and can be modeled as an order 1 AR process, although the AR coefficient was not significant.

Finally, we compute the prediction of `logP` via the `predict()` method and compare the prediction with the actual data from January to October 2013. Note the prediction makes use of observed discharge data in 2013. Fig. 8 shows the point predictors (blue dashed line) against the actual values (black solid line) and the 95% prediction bands (red lines), which indicates that the prediction tracks the actual data well.

A simulation study on model selection

In this section, we report a simulation study on the effectiveness of model selection by minimizing the AIC. Recall this functionality is implemented by the `carxSelect()` function which outputs the model with the smallest AIC from a set of models of various AR orders up to some pre-specified maximum order.

We restrict the simulation study to the problem of selecting the AR order with the same model specification, i.e, the same set of regressors, which is conducted as follows. We simulated 1000 series by calling `carxSim()` with the default setting, hence the true AR order is equal to 2, and for each simulated series we selected the best model among the models with the AR order from 1 to 6. Since the uncensored data were available in the simulation, we repeated the model selection with the uncensored observed data, for comparing with the results using the censored data. This simulation study can be reproduced by the following code.

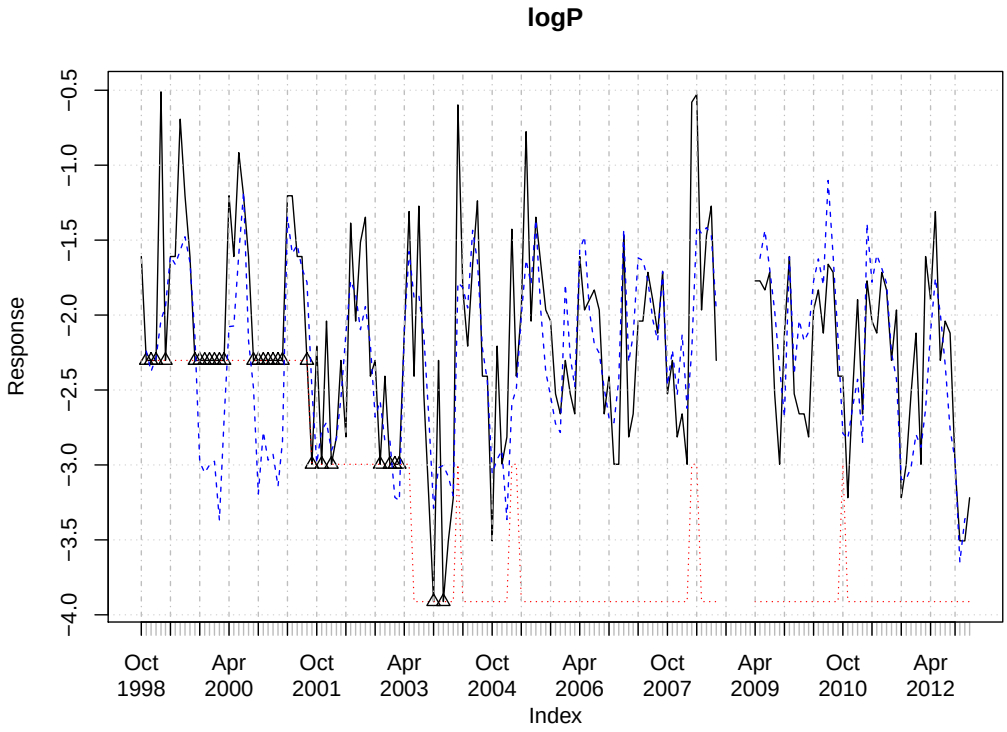

Figure 6: Time plot of the raw data and fitted values from the CARX model. The observed responses are connected as a solid black line, and the lower censoring limits drawn as a red dotted line with censored observations marked by triangles pointing up. The fitted values are connected as a blue dashed line. Outliers (if detected) are marked with a dashed red vertical line.

```
> singleTestSelectAROrder <- function(iter)
+ {
+   seed <- 1375911
+   cts <- carxSim(seed = iter*seed)
+   m0 <- carxSelect(list(f1 = as.formula(y~X1+X2-1)), max.ar = 6,
+                    data=cts[,c("y","X1","X2")],detect.outlier = FALSE)
+   m1 <- carxSelect(list(f1 = as.formula(y~X1+X2-1)),max.ar = 6,
+                    data = cts, detect.outlier = FALSE)
+   c(m0$fitted$p, m1$fitted$p)
+ }
> nRep <- 1000
> orders <- parallel::mclapply( 1:nRep, singleTestSelectAROrder,
+           mc.cores = parallel::detectCores() - 1)
> orders <- do.call(rbind, lapply(orders, matrix, ncol = 2, byrow = TRUE))
> freqComDat = count(orders[1,])
> freqCenDat = count(orders[2,])
```

The selected orders are reported in Table 1, which shows that the true order can be recovered with an empirical probability of 52.7%, and the results using censored and complete data are comparable.

AR order	1	2	3	4	5	6
Frequency (complete data)	17	626	139	97	69	52
Frequency (censored data)	37	527	160	99	101	76

Table 1: Summary of selected orders. The frequency of selected orders with complete data and censored data are reported.

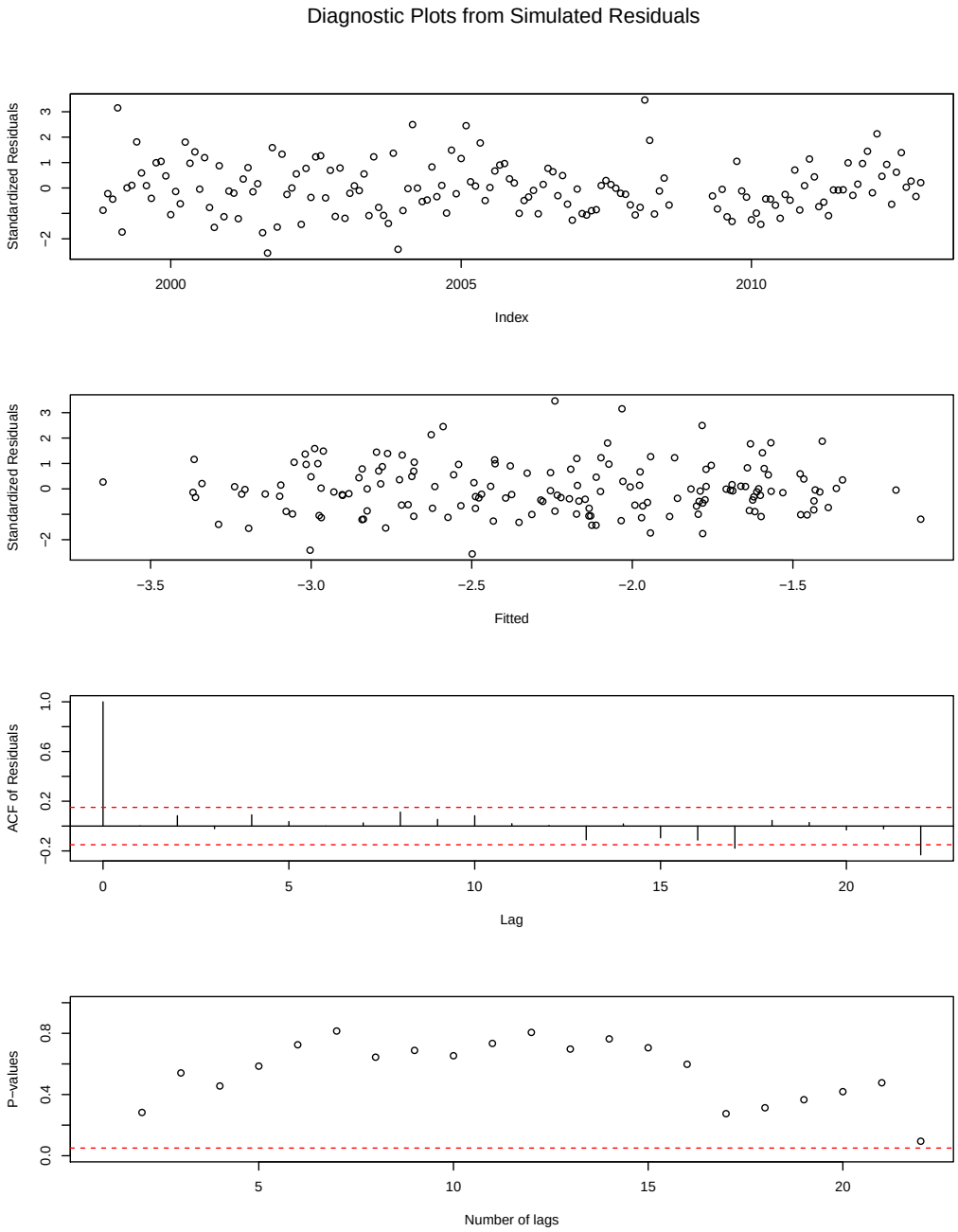

Figure 7: Model diagnostic plots. The plots from top to bottom correspond respectively to the time series plots of standadized simulated residuals, residuals versus fitted values, the residual autocorrelation plots, and the Ljung-Box test statistics of the residuals.

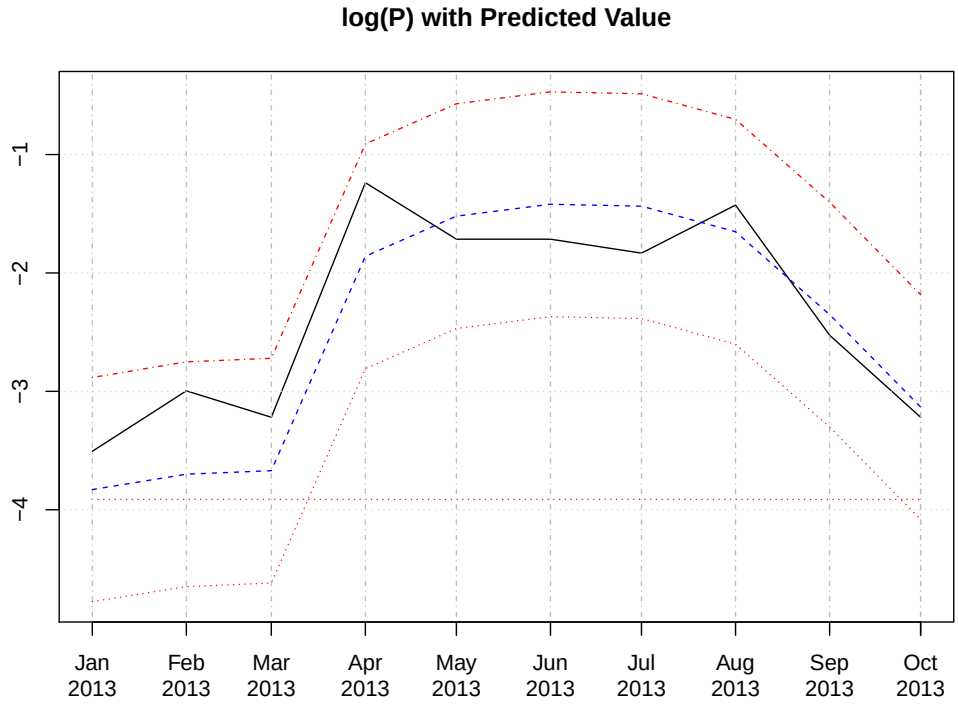

Figure 8: Plot of predictions and observed values. The observed values are drawn by black solid line. The predicted values, lower and upper bound of confidence intervals are drawn by blue dashed, red dotted, and red dash dotted lines respectively.

Performance of model prediction

In this section we report a simulation study about the empirical performance of the model prediction procedure. A series of 210 data was simulated using the default parameters of `carxSim()`, with the first 200 data used to estimate the model, and the last 10 observations used to compare with the predicted values based on the fitted model and the simulated future covariate values. The above procedure was repeated 500 times. The empirical coverage rates of the 95% ℓ-step ahead prediction intervals, $\ell = 1, 2, \ldots, 10$, are summarized in Table 2, which indicates a close match between the empirical and nominal coverage rates. The simulation exercise can be reproduced by the following code.

```
> nRep = 500; nObs = 200; n.ahead=10
> runSimPredCR <- function()
+ {
+   set.seed(0)
+   crMat = matrix(nrow = n.ahead, ncol = nRep)
+   for(iRep in 1:nRep)
+   {
+     sdata = carxSim(nObs = nObs + n.ahead)
+     trainingData = sdata[1:nObs,]
+     testData = sdata[-(1:nObs),]
+     mdl = carx(y ~ X1 + X2 - 1, data = trainingData, p = 2)
+     newxreg = testData[,c('X1','X2')]
+     predVal = predict(mdl, newxreg = newxreg, n.ahead = n.ahead)
+     crInd = (predVal$ci[,1] <= testData$y) & (predVal$ci[,2] >= testData$y)
+     crMat[,iRep] = crInd
+   }
+   crPred = apply(crMat,1,mean)
+ }
> runSimPredCR()
```

n.ahead	1	2	3	4	5	6	7	8	9	10
Empirical Coverage	0.954	0.932	0.946	0.952	0.946	0.946	0.938	0.946	0.948	0.952

Table 2: Empirical coverage rates of nominally 95% predictive confidence intervals for ℓ-step-ahead prediction, for $\ell = 1, 2, \ldots, 10$.

Conclusion

In summary, we have reviewed the quasi-likelihood method to estimate a censored time series regression model and introduced the **carx** package in which quasi-likelihood estimation is implemented, together with other useful functions for model selection, prediction, diagnostics and outlier detections. We illustrated the **carx** package with two major examples, and shed light on the effectiveness of model selection via minimizing AIC and the prediction accuracy.

Future work includes extending the method for more complex regression noise structure than the AR model, for instance, the more general ARIMA model, and updating the package according to the feedback from the public.

Bibliography

J. D. Cryer and K. S. Chan. *Time Series Analysis: With Applications in R*. Springer-Verlag, New York, 2008. [p4, 5]

J. Buckley and I. James. Linear regression with censored data. *Biometrika*, 66(3):429–436, 1979. [p1]

K.-S. Chan and B. Ripley. *TSA: Time Series Analysis*, 2012. R package version 1.01. [p9]

A. Genz, F. Bretz, T. Miwa, X. Mi, F. Leisch, F. Scheipl, and T. Hothorn. *mvtnorm: Multivariate Normal and t Distributions*, 2014. R package version 1.0-0. [p4]

C. Gourieroux, A. Monfort, E. Renault, and A. Trognon. Simulated residuals. *Journal of Econometrics*, 34(1):201–252, 1987. [p4]

A. Henningsen. Estimating censored regression models in r using the censreg package. *University of Copenhagen*, 2010. [p1]

A. Henningsen. *censReg: Censored Regression (Tobit) Models*, 2013. URL http://CRAN.R-project.org/package=censReg. R package version 0.5-20. [p1]

J. Huang and A. J. Rossini. Sieve estimation for the proportional-odds failure-time regression model with interval censoring. *Journal of the American Statistical Association*, 92(439):960–967, 1997. ISSN 01621459. URL http://www.jstor.org/stable/2965559. [p2]

C. Kleiber and A. Zeileis. *Applied Econometrics with R*. Springer-Verlag, New York, 2008. URL http://CRAN.R-project.org/package=AER. ISBN 978-0-387-77316-2. [p1]

S. Konishi and G. Kitagawa. *Information Criteria and Statistical Modeling*. Springer-Verlag, 2008. [p3]

L. Lee. *NADA: Nondetects And Data Analysis for Environmental Data*, 2013. URL http://CRAN.R-project.org/package=NADA. R package version 1.5-6. [p1]

R. D. Libra, C. F. Wolter, and R. J. Langel. *Nitrogen and Phosphorus Budgets for Iowa and Iowa Watersheds*. Iowa Department of Natural Resources, Geological Survey, 2004. [p10]

A. D. Martin, K. M. Quinn, and J. H. Park. MCMCpack: Markov chain monte carlo in R. *Journal of Statistical Software*, 42(9):22, 2011. URL http://www.jstatsoft.org/v42/i09/. [p1]

A. I. McLeod, N. M. Mohammad, J. Veenstra, and A. El-Shaarawi. *Cents: Censored Time Series*, 2014. URL http://CRAN.R-project.org/package=cents. R package version 0.1-41. [p1]

J. W. Park, M. G. Genton, and S. K. Ghosh. Censored time series analysis with autoregressive moving average models. *Canadian Journal of Statistics*, 35(1):151–168, 2007. ISSN 1708-945X. URL https://doi.org/10.1002/cjs.5550350113. [p1, 2]

J. A. Ryan and J. M. Ulrich. *Xts: eXtensible Time Series*, 2014. R package version 0.9-7. [p2]

K. E. Schilling, K. S. Chan, H. Liu, and Y. K. Zhang. Quantifying the effect of land use land cover change on increasing discharge in the upper mississippi river. *Journal of Hydrology*, 387(3):343–345, 2010. [p12]

F. L. Schumacher, V. H. Lachos, and C. E. Galarza. *ARCensReg: Fitting Univariate Censored Linear Regression Model with Autoregressive Errors*, 2016. URL http://CRAN.R-project.org/package=ARCensReg. R package version 2.1. [p1]

G. M. Tallis. The moment generating function of the truncated multi-normal distribution. *Journal of the Royal Statistical Society B*, 23(1):223–229, 1961. [p4]

J. Tobin. Estimation of relationships for limited dependent variables. *Econometrica: journal of the Econometric Society*, pages 24–36, 1958. [p1]

C. Wang and K.-S. Chan. *Carx: Censored Autoregressive Model with Exogenous Covariates*, 2016. URL http://CRAN.R-project.org/package=carx. R package version 0.6.2. [p1]

C. Wang and K.-S. Chan. Quasi-likelihood estimation of a censored autoregressive model with exogenous variables. *Journal of the American Statistical Association*, 0(ja):0–0, 2017. URL https://doi.org/10.1080/01621459.2017.1307115. [p1, 2, 3, 4, 6, 9]

C. Wang, K.-S. Chan, and K. E. Schilling. Total phosphorus concentration trends in 40 iowa rivers, 1999 to 2013. *Journal of Environmental Quality*, 2016. [p10]

T. W. Yee. *Vector Generalized Linear and Additive Models*. Springer-Verlag, 2007. [p1]

S. L. Zeger and R. Brookmeyer. Regression analsis with censored autocorrelated data. *Journal of the American Statistical Association*, 81(395):722–729, 1986. ISSN 01621459. [p1, 4]

Chao Wang
Department of Statistics and Actuarial Science
The University of Iowa
Iowa City, IA 52242
USA
chao-wang@uiowa.edu

Kung-Sik Chan
Department of Statistics and Actuarial Science
The University of Iowa
Iowa City, IA 52242
USA
kung-sik-chan@uiowa.edu