# An Introduction to Bayesian Quantile Regression for Binary Longutudinal Data with R Package `qbld`

Ayush Agarwal

August 27, 2020

## Contents

# 1 Introduction

The R package `qbld` follows **Rahman and Vossmeyer (2019)** as its motivating literature, and contributes by extending the various methodologies in quantile framework, to a hierarchical Bayesian quantile regression model for binary longitudinal data (QBLD) and proposing a Markov chain Monte Carlo (MCMC) algorithm to estimate the model. The model handles both common (fixed) and individual-specific (random) parameters (commonly referred to as mixed effects in statistics). The algorithm implements a blocking, and an unblocking procedure that is computationally efficient and the distributions involved allow for straightforward calculations of covariate effects.

# 2 Quantile Regression for Binary Longitudinal Data

## 2.1 The Model

The QBLD model can be conveniently expressed in the latent variable formulation (Albert & Chib, 1993) as follows:

$$\begin{aligned} z_{it} &= x_{it}^{'}\beta + s_{it}^{'}\alpha_i + \epsilon_{it}, \qquad \forall i = 1, ..., n; t = 1, ..., T_i \\ y_{it} &= \begin{cases} 1 & if\ z_{it} > 0 \\ 0 & \text{otherwise,} \end{cases} \end{aligned} \tag{1}$$

$y_{it}$ = response variable $y$ at $t^{th}$ time period for the $i^{th}$ individual,
$z_{it}$ = unobserved latent variable $z$ at $t^{th}$ time period for the $i^{th}$ individual,
$x_{it}^{'} = 1 * k$ vector of fixed-effects covariates,
$\beta = k * 1$ vector of fixed-effects parameters,
$s_{it}^{'} = 1 * l$ vector of covariates that have individual-specific effects,
$\alpha_i = l * 1$ vector of individual-specific parameters, and
$\epsilon_{it}$ = the error term $\overset{\text{iid}}{\sim} AL(0, 1, p)$. (AL is Asymmteric Laplace Distribution)

## 2.2 Asymmetric Laplace Distribution

While working directly with the AL density is an option, the resulting posterior will not yield the full set of tractable conditional distributions necessary for a Gibbs sampler. The mixture representation gives access to the appealing properties of the normal distribution. Thus, we utilize the normal-exponential mixture representation of the AL distribution, presented in Kozumi and Kobayashi (2011) :

$$\epsilon_{it} = w_{it}\theta + \tau\sqrt{w_{it}}u_{it} \qquad \forall i = 1, ..., n; t = 1, ..., T_i \tag{2}$$

$u_{it} \sim N(0,1)$, is mutually inependent of $w_{it} \sim \exp(1)$,
$\theta = \frac{1-2p}{p(1-p)}$, and $\tau = \sqrt{\frac{2}{p(1-p)}}$.

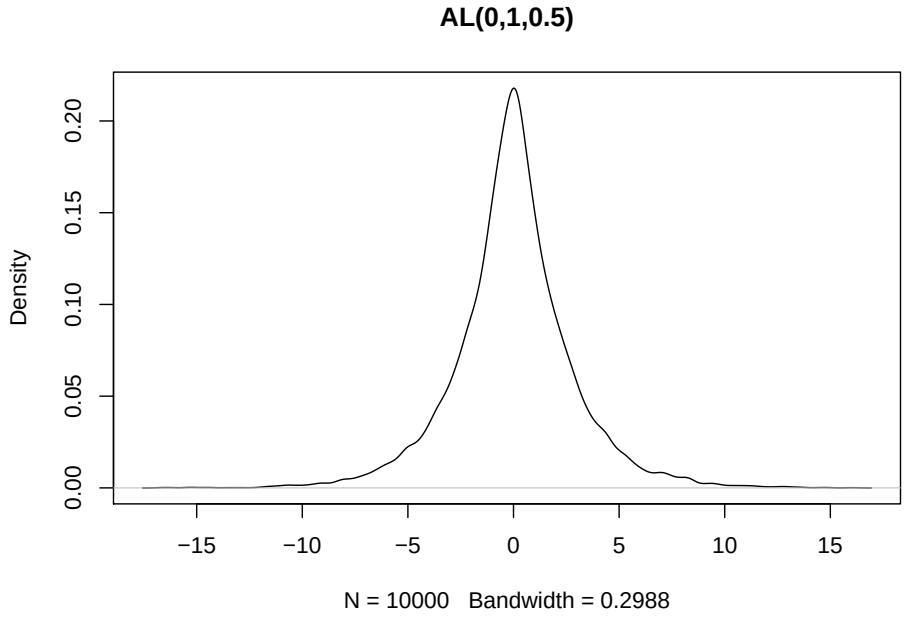
This mixture implementation of the AL distribution as in (2) is used to random sample, and I shall demonstarte the use of `raldmix` function to generate the said sample. For a sense of completeness, the package contains Cumulative density function, Probability distribution function, Quantile function for AL distribution as well. For help using the functions, use `?aldmix`.

```
library(qbld)

## qbld:  Quantile Regression for Binary Longitudinal Data
## Version 1.0 created on 2020-08-17.
##
## For citation information, type citation("qbld").
## Type help("qbld-package") or help("model.qbld") to get started.

set.seed(10)

#generate 1e4 samples
ald.sample <- raldmix(n = 1e4, mu = 0, sigma = 1, p = 0.5)
plot(density(ald.sample), main="AL(0,1,0.5)")
```




```
## additional functions
ald.density <- daldmix(c(4,5),mu = 0,sigma = 1,p = 0.5)
ald.cdf <- paldmix(c(1,4),mu = 0,sigma = 1,p = 0.5,lower.tail=TRUE)
ald.quantile <- qaldmix(0.5,mu = 0,sigma = 1,p = 0.5,lower.tail=TRUE)
```

### 2.3 Model with Priors

Longitudinal data models often involve a moderately large amount of data, so it is important to take advantage of any opportunity to reduce the computational burden. One such trick is to stack the model for each individual $i$ (Hendricks, Koenker, & Poirier, 1979).

We define, $z_i = (z_{i1}, ..., z_{iT_i})'$, $X_i = (x_{i1}', ..., x_{iT_i}')'$, $S_i = (s_{i1}', ..., s_{iT_i}')'$, $w_i = (w_{i1}, ..., w_{iT_i})'$, $D_{\tau\sqrt{w_i}} = diag(\tau\sqrt{w_{i1}}, ..., \tau\sqrt{w_{iT_i}})'$, and $u_i = (u_{i1}, ..., u_{iT_i})'$.

Building on Eqs. (1) and (2),

$$
\begin{aligned}
z_i &= X_i\beta + S_i\alpha_i + w_i\theta + D_{\tau\sqrt{w_i}}u_i, \\
y_{it} &= \begin{cases} 1 & if\ z_{it} > 0 \\ 0 & \text{otherwise,} \end{cases} \\
\alpha_i|\varphi^2 &\sim N_l(0, \varphi^2 I_l), w_{it} \sim \exp(1), u_{it} \sim N(0,1) \\
\beta &\sim N_k(\beta_0, B_0), \varphi^2 \sim IG(c1/2, d1/2)
\end{aligned} \tag{3}
$$

## 3 Blocked vs Unblocked Sampler

We can derive the conditional posteriors of the parameters and latent variables by a straightforward extension of the estimation technique for the linear mixed-effects model presented in Luo et al.(2012). This is presented as Algorithm 2 in Appendix, which shows the conditional posterior distributions for the parameters and latent variables necessary for a Gibbs sampler.

While this **Unblocked** Gibbs sampler is straightforward, there is potential for poor mixing properties due to correlation between $(\beta, \alpha_i)$ and $(z_i, \alpha_i)$. The correlation often arises because the variables corresponding to the parameters in $\alpha_i$ are often a subset of those in $x_{it}$. Thus, by conditioning these items on one another, the mixing of the Markov chain will be slow.

To avoid this issue, we present an alternative algorithm which jointly samples $(\beta, z_i)$ in one block within the Gibbs sampler. This is presented as Algorithm 1 in Appendix. This **blocked** approach significantly improves the mixing properties of the Markov chain.

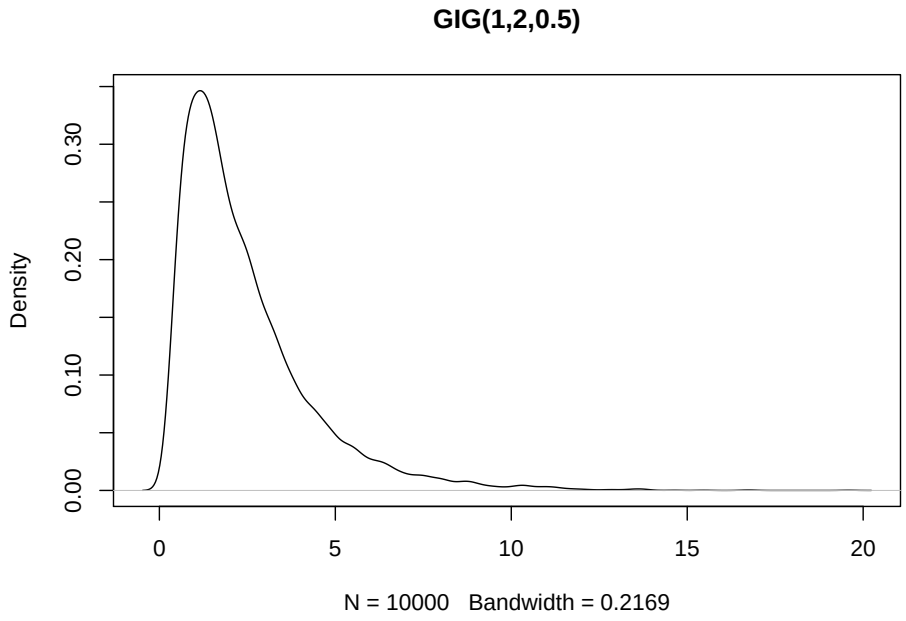
## 4 Generalized Inverse Gaussian Distribution

Gibbs sampler used in the model requires random sampling from Generalized Inverse Gaussian(GIG) distribution. For the sake of completion, the random generation function `rgig`, and the density function, `dgig` are made available to the user. For help using the functions, use `?gig`.
The Generalised Inverse Gaussian distrubtion(GIG), which has the following pdf:

$$
f(a, b, p) = \frac{(a/b)^{p/2}}{2K_p(\sqrt{ab})} \exp -\frac{ax + b/x}{2}, \quad x > 0 \tag{4}
$$

where $a, b > 0$ and $p \in \mathbb{R}$ are the parameters, and $K_p$ is a modified Bessel function of the second kind.

```r
# random generation
gig.sample <- rgig(n = 1e4, lambda = 0.5, a = 1, b = 2)
plot(density(gig.sample),main="GIG(1,2,0.5)")
```




```r
# density
gig.density <- dgig(x = 1, a = 1, b = 2, p = 0.5, log_density = FALSE)
```

## 5 Using `qbld`

Let us examine the dataset we will use to demonstrate the sample usage of the package.

### 5.1 Dataset:- Airpollution

This example datset is a subset of data from Six Cities study, a longitudinal study of the health effects of air pollution. The data set contains complete records on 537 children from Ohio, each woman was examined annually at ages 7 through 10. The repeated binary response is the wheezing status (1="yes", 0="no") of a child at each occasion. Although mother's smoking status could vary with time, it was determined in the first interview and was

treated as a time-independent covariate. Maternal smoking was categorized as 1 if the mother smoked regularly and 0 otherwise.

```
data(airpollution)
str(airpollution)

## 'data.frame': 128 obs. of  5 variables:
##  $ id     : int  1 1 1 1 2 2 2 2 3 3 ...
##  $ wheeze : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ age    : num  7 8 9 10 7 8 9 10 7 8 ...
##  $ smoking: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ counts : num  237 237 237 237 10 10 10 10 15 15 ...
```

### 5.2 `model.qbld`: Estimation of QBLD model

`model.qbld` runs the QBLD sampler as described above, and outputs a `qbld` class object. In this example, we will model wheezing status (`wheeze`) in terms of `age` and `smoking`. We will also treat `counts` as a random-effect parameter and we will allow intercepts for both fixed and random effects.

```
##modelling the output :- Blocked
#no burn, no verbose, no summary

output.block <- model.qbld(fixed_formula = wheeze~smoking+I(age^2)+age,
                           data = airpollution, id="id",
                           random_formula = ~counts+1, p=0.25,
                           nsim=5000, method="block", burn=0,
                           summarize=FALSE, verbose=FALSE)
```

Let us look at the arguments one by one:

- `fixed_formula:` A description of the model to be fitted of the form $response \sim fixed$ effects predictors i.e $X_i$ in the model (3). Response variable is mandatory, and empty formula will throw error.

  In this example, $wheeze \sim smoking + I(age^2) + age$ translates to response variable, $y_i =$ `wheeze`, and $x_i$ as $smoking$, $age$, $age^2$, and $Intercept$.

- `id:` A variable name in the dataset that specifies individual profile. By default, id = "id", and hence, data is expected to contain an id

variable. Note that this is not a covariate, and is omitted while modelling.

- `data:` Data are contained in a `data.frame`. Each element of the data argument must be identifiable by a name. The simplest situation occurs when all subjects are observed at the same time points. Using datasets with different time points should be avoided. NAs are not allowed and should throw errors. All factor variables are auto-converted to numeric levels. Two datasets, `airpollution` and `locust` are built into the package.

- `random_formula:` A description of the model to be fitted of the form $response \sim random$ effects predictors i.e $S_i$ in the model. Response variable is not required, and is ignored. This defaults to $S_i$ being only an intercept.

  In this example, $\sim counts{+}1$ translates to $s_i$ as $counts$`, and` $Intercept$.

- `p:` Quantile for the AL distribution on the error term, $p = 0.25$ by default. For very low ($\leq 0.025$) or very high ($\geq 0.970$) values of $p$, sampler forces to unblock version to avoid errors in the block procedure.

- `nsim:` No. of simulations to run the sampler.

- `b0, B0:` Prior model parameters for Beta as in the model (3). These are defaulted to 0 vector, and Identity matrix of appropriate dimensions. Full Gibbs Sampler is not affected by starting values, and need not be specified.

- `c1, d1:` Prior model parameters for Varphi2 as in the model (3). These are defaulted to 9, 10 (arbitrary) respectively. Full Gibbs Sampler is not affected by starting values, and need not be specified.

- `method:` Choose between the"Block" vs "Unblock" sampler, Block is slower, but produces lower correlation. Check section 3 for a detailed comparsion. I would recommend using "Unblock" for larger datasets. The code uses regex and is impervious to alphabet case related errors.

- `burn:` Burn in percentage, number between (0,1). Burn-in values are discarded while outputting and are not used for summary statistical calculations. No. of simulations are adjusted for burn-in before ESS calculations.

- `summarize:` Outputs a summary table (same as `summary(output)`). In addition to this, also prints Model fit diagonstics such as AIC, BIC, and Log-likelihood values. I recommend using this if model fit values are of significance. False by default.
- `verbose:` False by default. If True, spits out progress reports while the sampler is running. This will print simulation progress for 10 times. i.e prints every 100th simulation if `nsim` = 1000.

The output of this function is a `qbld` class object.

```
str(output.block)

## List of 3
##  $ Beta   : num [1:5000, 1:4] 0 0.133 0.162 -0.695 1.423 ...
##  $ Alpha  : num [1:2, 1:32, 1:5000] -1.7443 -0.0676 -2.1597 1.4258 -1.1158 ...
##  $ Varphi2: num [1:5000, 1] 1 0.754 0.535 0.58 0.636 ...
##  - attr(*, "burn")= logi FALSE
##  - attr(*, "nsim")= num 5000
##  - attr(*, "which")= chr "block"
##  - attr(*, "varnames")= chr [1:5] "Intercept" "smoking" "I(age^2)" "age" ...
##  - attr(*, "class")= chr "qbld"
##  - attr(*, "quantile")= num 0.25
```

`qbld` class object contains the following attributes:

- `Beta:` Matrix of MCMC samples of fixed-effects parameters.
- `Alpha:` 3D Matrix of MCMC samples of random-effects parameters.
- `Varphi2:` Matrix of MCMC samples for varphi2.
- `nsim:` Attribute; No. of simulations of chain run.
- `burn:` Attribute; Whether or not burn-in used.
- `which:` Attribute; "block" or "unblock" sampler used

### 5.3 `summary.qbld`: Summarizing the `qbld` output

One way of summarizing the model is to use the summarize argument. Let us have a look at the Unblocked sampler and `summarize` option.

```
##modelling the output :- Unblocked
#Using burn, no verbose, and summary

output.unblock <- model.qbld(fixed_formula = wheeze~smoking+I(age^2)+age,
                             data = airpollution, id="id",
                             random_formula = ~counts+1, p=0.25,
                             nsim=5000, method="Unblock", burn=0.2,
                             summarize=TRUE, verbose=FALSE)

## Please wait while we're processing your request.
## I recommend listening to Vienna by Billy Joel while you wait.
## https://music.apple.com/in/album/vienna/158617952?i=158618071
##
## Quantile used = 0.25
##
## No. of Iterations = 4000 samples
## Type of Sampler = unblock
## Burn-in Used? = TRUE
##
## 1. Statistics for each variable,
##              Mean    SD  MCSE  ESS Gelman-Rubin
## Intercept -0.052 0.950 0.016 3377     1.000023 *
## smoking   -0.093 0.720 0.046  237     1.001983
## I(age^2)   0.013 0.040 0.003  252     1.001857
## age       -0.260 0.480 0.027  302     1.001529
## Varphi2    0.478 0.141 0.006  478     1.000921
##
## MultiESS value = 457.9996
## Multi Gelman-Rubin = 1.000966
## Note : * indicates enough samples for the covariate
##          *** indicates enough samples for the whole sampler.
##
## 2. Quantiles for each variable,
##              2.5%    25%    50%   75% 97.5%
## Intercept -1.890 -0.696 -0.050 0.599 1.802
```

```
## smoking   -1.475 -0.569 -0.123 0.381 1.308
## I(age^2)  -0.076 -0.015  0.015 0.043 0.096
## age       -1.186 -0.591 -0.266 0.067 0.669
## Varphi2    0.276  0.379  0.455 0.549 0.813
##
##
## 3. Model Selection Criterion
## Log likelihood = -71.31597
## AIC = 154.6317
## BIC = 172.4373
```

The second way is to use the S3 method, `summary` function which produces a `summary.qbld` class object.

```
summary.unblock = summary(output.unblock,
                          quantiles = c(0.025, 0.25, 0.5, 0.75, 0.975),
                          epsilon=0.10)
str(summary.unblock)

## List of 9
##  $ statistics :'data.frame': 5 obs. of  6 variables:
##   ..$ Mean        : num [1:5] -0.052 -0.093 0.013 -0.26 0.478
##   ..$ SD          : num [1:5] 0.95 0.72 0.04 0.48 0.141
##   ..$ MCSE        : num [1:5] 0.016 0.046 0.003 0.027 0.006
##   ..$ ESS         : num [1:5] 3377 237 252 302 478
##   ..$ Gelman-Rubin: num [1:5] 1 1 1 1 1
##   ..$             : chr [1:5] "*" "" "" "" ...
##  $ quantiles  : num [1:5, 1:5] -1.89 -1.475 -0.076 -1.186 0.276 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:5] "Intercept" "smoking" "I(age^2)" "age" ...
##   .. ..$ : chr [1:5] "2.5%" "25%" "50%" "75%" ...
##  $ nsim       : num 4000
##  $ burn       : logi TRUE
##  $ which      : chr "unblock"
##  $ p          : num 0.25
##  $ multiess   : num 458
##  $ multigelman: num 1
##  $ foo        : logi FALSE
##  - attr(*, "class")= chr "summary.qbld"
```

Summary function has the following arguments:

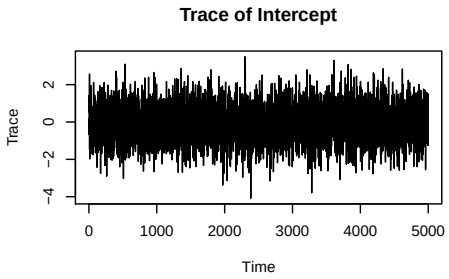
- `quantiles:` Vector of quantiles for summary of the covariates, defaulted to c(0.025, 0.25, 0.5, 0.75, 0.975)
- `epsilon:` 0.05 by default. Epsilon value is used for calculating `target.psrf` values, which estimate the ideal number of effective sample size required for a given level of significance. This value will be compared to generated ESS and significance stars are added accordingly. This process is repeated for individual chains and MultiESS, multi-Gelman by treating all the parameter chains as one multi-variate chain.

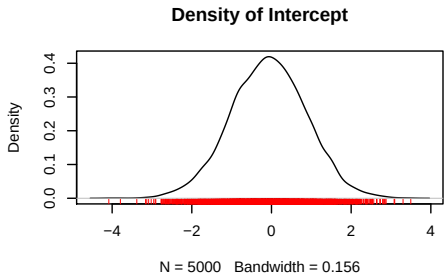
`qbld.summary` class object contains the following attributes:

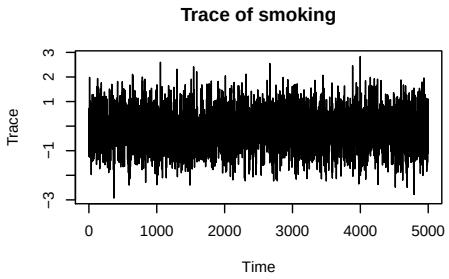
- `statistics:` Contains the mean, sd, markov std error, ess and Gelman-Ruben diagnostic
- `quantiles:` Contains quantile estimates for each variable
- `nsim:` No. of simulations run, adjusted for burn-in
- `burn:` Burn-in used or not
- `which:` Block, or Unblock version of sampler
- `p:` quantile for the AL distribution on the error term
- `multiess:` multiess value for the sample
- `multigelman:` multivariate version of Gelman-Ruben

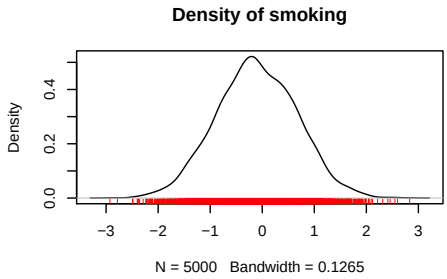
### 5.4 `plot.qbld:` Creating plots

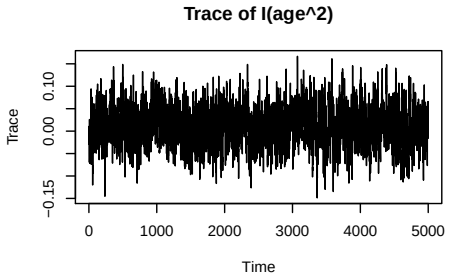
```
plot(output.block, trace = TRUE, density = TRUE,
     auto.layout = TRUE, ask = NULL)
```

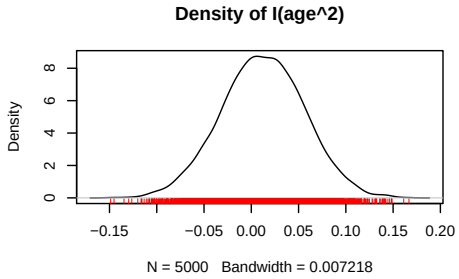

Trace of Intercept
Trace
Time
0
1000
2000
3000
4000
5000
−4
−2
0
2

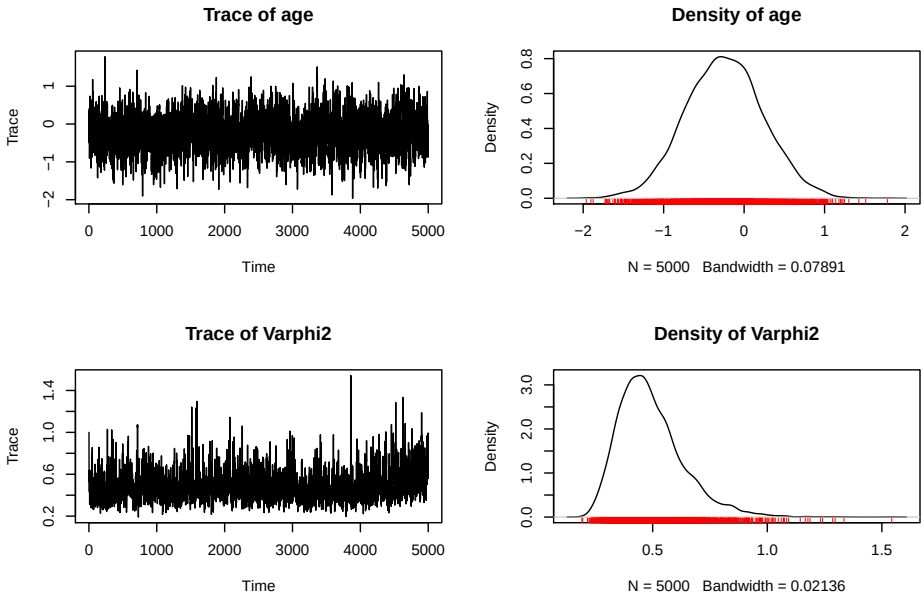

Density of Intercept
Density
N = 5000 Bandwidth = 0.156
0.0
0.1
0.2
0.3
0.4
−4
−2
0
2
4


Trace of smoking
Trace
Time
0
1000
2000
3000
4000
5000
−3
−1
1
2
3


Density of smoking
Density
N = 5000 Bandwidth = 0.1265
0.0
0.2
0.4
−3
−2
−1
0
1
2
3


Trace of I(age^2)
Trace
Time
0
1000
2000
3000
4000
5000
−0.15
0.00
0.10


Density of I(age^2)
Density
N = 5000 Bandwidth = 0.007218
0
2
4
6
8
−0.15
−0.05
0.00
0.05
0.10
0.15
0.20




Plot function has the following arguments:

- `trace:` Whether or not to plot trace plots for covariates, TRUE by default
- `density:` Whether or not to plot density for covariates, TRUE by default.
- `auto.layout:` Auto set layout or not, TRUE as default. Plots according to the local settings if false.

# 6 Appendix

## 6.1 Blocked Sampling

- Sample $(\beta, z_i)$ in one block. These are sampled in following two substeps.
  - Sample $\beta$

$$\begin{aligned} \beta|z, w, \varphi^2 &\sim N(\tilde{\beta}, \tilde{B}), \\ where, \quad \tilde{B}^{-1} &= (\sum_{i=1}^{n} X_i^{'} \Omega_i^{-1} X_i + B_0^{-1}), \\ \tilde{\beta} &= \tilde{B}(\sum_{i=1}^{n} X_i^{'} \Omega_i^{-1}(z_i - w_i\theta) + B_0^{-1}\beta_0), \\ \Omega_i &= (\varphi^2 S_i S_i^{'} + D^2_{\tau\sqrt{w_i}}). \end{aligned} \tag{5}$$

  - Sample the vector $z_i|y_i, \beta, w_i, \varphi^2 \sim TMVN_{B_i}(X_i\beta + w_i\theta, \Omega_i)$ for all $i = 1, ..., n$, where $B_i = (B_{i1} * B_{i2} * ... * B_{iT_i})$ and $B_{it}$ are interval $(0, \infty)$ if $y_{it} = 1$, and the interval $(-\infty, 0]$ if $y_{it} = 0$. This is done by sampling $z_i$ at the $j^{th}$ pass of the MCMC iteration using a series of conditional posteriors:

$$\begin{aligned} z_{it}^j|z_{i1}^j, ...z_{i(t-1)}^j, z_{i(t+1)}^{j-1}, ..., z_{iT_i}^{j-1} &\sim TN_{B_i}(\mu_{t|-t}, \Sigma_{t|-t}), \qquad t = 1, ..., T_i. \\ where, \quad \mu_{t|-t} &= x_{it}^{'}\beta + w_{it}\theta + \Sigma_{t,-t}\Sigma_{-t,-t}^{-1}(z_{i,-t}^j - (X_i\beta + w_i\theta)_{-t}), \\ \Sigma_{t|-t} &= \Sigma_{t,t} - \Sigma_{t,-t}\Sigma_{-t,-t}^{-1}\Sigma_{-t,t}, \end{aligned} \tag{6}$$

    where $z_{i,-t}^j = (z_{i1}^j, ...z_{i(t-1)}^j, z_{i(t+1)}^{j-1}, ..., z_{iT_i}^{j-1})$, $(X_i\beta + w_i\theta)_{-t}$ is column vector with $t^{th}$ element removed, $\Sigma_{t,t}, \Sigma_{t,-t}, \Sigma_{-t,-t}$ are $(t,t)^{th}$ element, $t^{th}$ row with $t^{th}$ element removed, and $t^{th}$ row and column removed respectively.

- Sample $\alpha$

$$\begin{aligned} &\alpha_i|z,\beta,w,\varphi^2 \sim N(\tilde{a},\tilde{A}), \quad \forall i=1,...,n \\ &where, \quad \tilde{A^{-1}} = (S_i' D^{-2}_{\tau\sqrt{w_i}} S_i + \frac{1}{\varphi^2} I_l), \\ &\tilde{a} = \tilde{A}(S_i' D^{-2}_{\tau\sqrt{w_i}} (z_i - X_i\beta - w_i\theta)). \end{aligned} \tag{7}$$

- Sample $w$

$$\begin{aligned} &w_{it}|z_{it},\beta,\alpha_i \sim GIG(0.5,\tilde{\lambda_{it}},\tilde{\eta}) \quad \forall i=1,...,n; t=1,...,T_i, \\ &where, \quad \tilde{\lambda_{it}} = (\frac{z_{it} - x_{it}'\beta - s_{it}'\alpha_i}{\tau})^2 \\ &\tilde{\eta} = (\frac{\theta^2}{\tau^2} + 2). \end{aligned} \tag{8}$$

- Sample $\varphi^2$

$$\begin{aligned} &\varphi^2|\alpha \sim IG(\tilde{c_1}/2, \tilde{d_1}/2), \\ &where, \quad \tilde{c_1} = (nl + c_1), \\ &\tilde{d_1} = (\sum_{i=1}^{n} \alpha_i'\alpha_i + d_1). \end{aligned} \tag{9}$$

.

### 6.2 Unblocked Sampling

- Sample $\beta$

$$\begin{aligned} &\beta|z,w,\varphi^2 \sim N(\tilde{\beta},\tilde{B}), \\ &where, \quad \tilde{B}^{-1} = (\sum_{i=1}^{n} X_i'\Psi_i^{-1}X_i + B_0^{-1}), \\ &\tilde{\beta} = \tilde{B}(\sum_{i=1}^{n} X_i'\Psi_i^{-1}(z_i - w_i\theta - S_i\alpha_i) + B_0^{-1}\beta_0), \\ &\Psi_i = D^2_{\tau\sqrt{w_i}}. \end{aligned} \tag{10}$$

- Sample $\alpha$ as in (7).
- Sample $w$ as in (8).
- Sample $\varphi^2$ as in (9).
- Sample $z|y, \alpha, w \ \forall i = 1, ..., n; t = 1, ..., T_i$, from univariate truncated normal as:

$$z_{it}|y, \beta, w = \begin{cases} TN_{(-\infty,0]}(x_{it}^{'}\beta + s_{it}^{'}\alpha_i + w_{it}\theta, \tau^2 w_{it}) & if\, y_{it} = 0 \\ TN_{(0,\infty)}(x_{it}^{'}\beta + s_{it}^{'}\alpha_i + w_{it}\theta, \tau^2 w_{it}) & if\, y_{it} = 1 \end{cases} \tag{11}$$

## 7 References


- Rahman, Mohammad & Vossmeyer, Angela. (2018). Estimation and Applications of Quantile Regression for Binary Longitudinal Data. Advances in Econometrics. 40.
- Vats, Dootika and Christina Knudson. "Revisiting the Gelman-Rubin Diagnostic." arXiv: Computation (2018): n. pag.
- Keming Yu & Jin Zhang (2005) A Three-Parameter Asymmetric Laplace Distribution and Its Extension, Communications in Statistics - Theory and Methods.
- Kobayashi, Genya. (2011). Gibbs Sampling Methods for Bayesian Quantile Regression. J Stat Comput Simul.
- Devroye, L. Random variate generation for the generalized inverse Gaussian distribution. Stat Comput 24, 239–246 (2014).
- Wolfgang Hörmann and Josef Leydold (2013). Generating generalized inverse Gaussian random variates, Statistics and Computing.
- J. S. Dagpunar (1989). An easily implemented generalised inverse Gaussian generator, Comm. Statist. B – Simulation Comput. 18, 703–710.