0.1 `tobit.bayes`: Bayesian Linear Regression for a Censored Dependent Variable

Bayesian tobit regression estimates a linear regression model with a censored dependent variable using a Gibbs sampler. The dependent variable may be censored from below and/or from above. For other linear regression models with fully observed dependent variables, see Bayesian regression (Section **??**), maximum likelihood normal regression (Section **??**), or least squares (Section **??**).

Syntax

```
> z.out <- zelig(Y ~ X1 + X2, below = 0, above = Inf,
                  model = "tobit.bayes", data = mydata)
> x.out <- setx(z.out)
> s.out <- sim(z.out, x = x.out)
```

Inputs

`zelig()` accepts the following arguments to specify how the dependent variable is censored.

- `below`: point at which the dependent variable is censored from below. If the dependent variable is only censored from above, set `below = -Inf`. The default value is 0.
- `above`: point at which the dependent variable is censored from above. If the dependent variable is only censored from below, set `above = Inf`. The default value is `Inf`.

Additional Inputs

Use the following arguments to monitor the convergence of the Markov chain:

- `burnin`: number of the initial MCMC iterations to be discarded (defaults to 1,000).
- `mcmc`: number of the MCMC iterations after burnin (defaults to 10,000).
- `thin`: thinning interval for the Markov chain. Only every `thin`-th draw from the Markov chain is kept. The value of `mcmc` must be divisible by this value. The default value is 1.
- `verbose`: defaults to `FALSE`. If `TRUE`, the progress of the sampler (every 10%) is printed to the screen.
- `seed`: seed for the random number generator. The default is `NA` which corresponds to a random seed of 12345.
- `beta.start`: starting values for the Markov chain, either a scalar or vector with length equal to the number of estimated coefficients. The default is `NA`, such that the least squares estimates are used as the starting values.

Use the following parameters to specify the model's priors:

- `b0`: prior mean for the coefficients, either a numeric vector or a scalar. If a scalar, that value will be the prior mean for all coefficients. The default is 0.
- `B0`: prior precision parameter for the coefficients, either a square matrix (with the dimensions equal to the number of the coefficients) or a scalar. If a scalar, that value times an identity matrix will be the prior precision parameter. The default is 0, which leads to an improper prior.
- `c0`: `c0/2` is the shape parameter for the Inverse Gamma prior on the variance of the disturbance terms.
- `d0`: `d0/2` is the scale parameter for the Inverse Gamma prior on the variance of the disturbance terms.

Zelig users may wish to refer to `help(MCMCtobit)` for more information.

Convergence

Users should verify that the Markov Chain converges to its stationary distribution. After running the `zelig()` function but before performing `setx()`, users may conduct the following convergence diagnostics tests:

- `geweke.diag(z.out$coefficients)`: The Geweke diagnostic tests the null hypothesis that the Markov chain is in the stationary distribution and produces z-statistics for each estimated parameter.
- `heidel.diag(z.out$coefficients)`: The Heidelberger-Welch diagnostic first tests the null hypothesis that the Markov Chain is in the stationary distribution and produces p-values for each estimated parameter. Calling `heidel.diag()` also produces output that indicates whether the mean of a marginal posterior distribution can be estimated with sufficient precision, assuming that the Markov Chain is in the stationary distribution.
- `raftery.diag(z.out$coefficients)`: The Raftery diagnostic indicates how long the Markov Chain should run before considering draws from the marginal posterior distributions sufficiently representative of the stationary distribution.

If there is evidence of non-convergence, adjust the values for `burnin` and `mcmc` and rerun `zelig()`.

Advanced users may wish to refer to `help(geweke.diag)`, `help(heidel.diag)`, and `help(raftery.diag)` for more information about these diagnostics.

Examples

1. Basic Example
   Attaching the sample dataset:

   ```
   > data(tobin)
   ```

   Estimating linear regression using `tobit.bayes`:

   ```
   > z.out <- zelig(durable ~ age + quant, model = "tobit.bayes",
   +     data = tobin, verbose = TRUE)
   ```

   Checking for convergence before summarizing the estimates:

   ```
   > geweke.diag(z.out$coefficients)

   > heidel.diag(z.out$coefficients)

   > raftery.diag(z.out$coefficients)

   > summary(z.out)
   ```

   Setting values for the explanatory variables to their sample averages:

   ```
   > x.out <- setx(z.out)
   ```

   Simulating quantities of interest from the posterior distribution given `x.out`.

   ```
   > s.out1 <- sim(z.out, x = x.out)

   > summary(s.out1)
   ```

2. Simulating First Differences
   Set explanatory variables to their default(mean/mode) values, with high (80th percentile) and low (20th percentile) liquidity ratio (`quant`):

   ```
   > x.high <- setx(z.out, quant = quantile(tobin$quant, prob = 0.8))
   > x.low <- setx(z.out, quant = quantile(tobin$quant, prob = 0.2))
   ```

   Estimating the first difference for the effect of high versus low liquidity ratio on duration(`durable`):

   ```
   > s.out2 <- sim(z.out, x = x.high, x1 = x.low)

   > summary(s.out2)
   ```

Model

Let Y_i^* be the dependent variable which is not directly observed. Instead, we observe Y_i which is defined as following:

$$Y_i = \begin{cases} Y_i^* & \text{if} \quad c_1 < Y_i^* < c_2 \\ c_1 & \text{if} \quad c_1 \geq Y_i^* \\ c_2 & \text{if} \quad c_2 \leq Y_i^* \end{cases}$$

where c_1 is the lower bound below which Y_i^* is censored, and c_2 is the upper bound above which Y_i^* is censored.

- The *stochastic component* is given by

$$\epsilon_i \sim \text{Normal}(0, \sigma^2)$$

  where $\epsilon_i = Y_i^* - \mu_i$.

- The *systematic component* is given by

$$\mu_i = x_i \beta,$$

  where x_i is the vector of k explanatory variables for observation i and β is the vector of coefficients.

- The *semi-conjugate priors* for β and σ^2 are given by

$$\begin{aligned} \beta &\sim \text{Normal}_k\left(b_0, B_0^{-1}\right) \\ \sigma^2 &\sim \text{InverseGamma}\left(\frac{c_0}{2}, \frac{d_0}{2}\right) \end{aligned}$$

  where b_0 is the vector of means for the k explanatory variables, B_0 is the $k \times k$ precision matrix (the inverse of a variance-covariance matrix), and $c_0/2$ and $d_0/2$ are the shape and scale parameters for σ^2. Note that β and σ^2 are assumed *a priori* independent.

Quantities of Interest

- The expected values (`qi$ev`) for the tobit regression model is calculated as following. Let

$$\begin{aligned} \Phi_1 &= \Phi\left(\frac{(c_1 - x\beta)}{\sigma}\right) \\ \Phi_2 &= \Phi\left(\frac{(c_2 - x\beta)}{\sigma}\right) \\ \phi_1 &= \phi\left(\frac{(c_1 - x\beta)}{\sigma}\right) \\ \phi_2 &= \phi\left(\frac{(c_2 - x\beta)}{\sigma}\right) \end{aligned}$$

where $\Phi(\cdot)$ is the (cumulative) Normal density function and $\phi(\cdot)$ is the Normal probability density function of the standard normal distribution. Then the expected values are

$$\begin{aligned} E(Y|x) &= P(Y^* \leq c_1|x)c_1 + P(c_1 < Y^* < c_2|x)E(Y^* \mid c_1 < Y^* < c_2, x) + P(Y^* \geq c_2)c_2 \\ &= \Phi_1 c_1 + x\beta(\Phi_2 - \Phi_1) + \sigma(\phi_1 - \phi_2) + (1 - \Phi_2)c_2, \end{aligned}$$

- The first difference (`qi$fd`) for the tobit regression model is defined as

$$\text{FD} = E(Y \mid x_1) - E(Y \mid x).$$

- In conditional prediction models, the average expected treatment effect (`qi$att.ev`) for the treatment group is

$$\frac{1}{\sum t_i} \sum_{i:t_i=1} [Y_i(t_i = 1) - E[Y_i(t_i = 0)]],$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups.

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run:

```
z.out <- zelig(y ~ x, model = "tobit.bayes", data)
```

then you may examine the available information in `z.out` by using `names(z.out)`, see the draws from the posterior distribution of the `coefficients` by using `z.out$coefficients`, and view a default summary of information through `summary(z.out)`. Other elements available through the `$` operator are listed below.

- From the `zelig()` output object `z.out`, you may extract:
  - `coefficients`: draws from the posterior distributions of the estimated parameters. The first k columns contain the posterior draws of the coefficients β, and the last column contains the posterior draws of the variance σ^2.
  - `zelig.data`: the input data frame if `save.data = TRUE`.
  - `seed`: the random seed used in the model.
- From the `sim()` output object `s.out`:
  - `qi$ev`: the simulated expected value for the specified values of `x`.
  - `qi$fd`: the simulated first difference in the expected values given the values specified in `x` and `x1`.
  - `qi$att.ev`: the simulated average expected treatment effect for the treated from conditional prediction models.

How to Cite

To cite the *tobit.bayes* Zelig model:

> Ben Goodrich and Ying Lu. 2007. "tobit.bayes: Bayesian Linear Regression for a Censored Dependent Variable," in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software," `http://gking.harvard.edu/zelig`.

To cite Zelig as a whole, please reference these two sources:

> Kosuke Imai, Gary King, and Olivia Lau. 2007. "Zelig: Everyone's Statistical Software," `http://GKing.harvard.edu/zelig`.

> Kosuke Imai, Gary King, and Olivia Lau. 2007. "Toward A Common Framework for Statistical Analysis and Development," `http://gking.harvard.edu/files/abs/z-abs.shtml`.

See also

Bayesian tobit regression is part of the MCMCpack library by Andrew D. Martin and Kevin M. Quinn (Martin and Quinn 2005). The convergence diagnostics are part of the CODA library by Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines (Plummer et al. 2005).

Bibliography

Martin, A. D. and Quinn, K. M. (2005), *MCMCpack: Markov chain Monte Carlo (MCMC) Package*.

Plummer, M., Best, N., Cowles, K., and Vines, K. (2005), *coda: Output analysis and diagnostics for MCMC*.