1 `logit.bayes`: Bayesian Logistic Regression

Logistic regression specifies a dichotomous dependent variable as a function of a set of explanatory variables using a random walk Metropolis algorithm. For a maximum likelihood implementation, see Section **??**.

1.1 Syntax

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

1.2 Additional Inputs

Use the following arguments to monitor 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.
- `tune`: Metropolis tuning parameter, either a positive scalar or a vector of length k, where k is the number of coefficients. The tuning parameter should be set such that the acceptance rate of the Metropolis algorithm is satisfactory (typically between 0.20 and 0.5) before using the posterior density for inference. The default value is 1.1.
- `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 maximum likelihood 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 value, that value will be the prior mean for all the coefficients. The default is 0.

- B0: prior precision parameter for the coefficients, either a square matrix (with the dimensions equal to the number of coefficients) or a scalar. If a scalar value, that value times an identity matrix will be the prior precision parameter. The default is 0, which leads to an improper prior.

1.3 Examples

1. Basic Example
   Attaching the sample dataset:

```
> data(turnout)
```

   Estimating the logistic regression using logit.bayes:

```
> z.out <- zelig(vote ~ race + educate, model = "logit.bayes",
+                  data = turnout, verbose = FALSE)

 How to cite this model in Zelig:
  NAMELESS AUTHOR. 2012.
  "logit.bayes: "
  in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
  http://gking.harvard.edu/zelig
```

   Convergence diagnostics before summarizing the estimates:

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

Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5

(Intercept)   racewhite     educate
     1.8457      0.1102     -1.8773

> heidel.diag(z.out$result$coefficients)

            Stationarity start     p-value
            test         iteration
(Intercept) passed       1         0.430
racewhite   passed       1         0.323
educate     passed       1         0.345

            Halfwidth Mean   Halfwidth
            test
(Intercept) passed    -1.220 0.01433
racewhite   passed     0.508 0.00938
educate     passed     0.161 0.00107

> raftery.diag(z.out$result$coefficients)
```

```
Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

             Burn-in  Total Lower bound  Dependence
             (M)      (N)   (Nmin)       factor (I)
 (Intercept) 21       22362 3746         5.97
 racewhite   20       21329 3746         5.69
 educate     18       19684 3746         5.25
```

```
> summary(z.out)

Call: MCMClogit(formula = vote ~ race + educate, data = Data.frame,
    verbose = FALSE, burnin = 1000, mcmc = 10000)

Iterations = 1001:11000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 10000

 Mean, standard deviation, and quantiles for marginal posterior distributions.
               Mean     SD    2.5%     50%   97.5%
(Intercept) -1.2196 0.2197 -1.6679 -1.2134 -0.7968
racewhite    0.5078 0.1393  0.2326  0.5071  0.7889
educate      0.1606 0.0169  0.1289  0.1599  0.1948
```

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)

Model:  logit.bayes
Number of simulations:  1000

Values of X
  (Intercept) racewhite  educate
1           1         1 12.06675
attr(,"assign")
[1] 0 1 2
attr(,"contrasts")
attr(,"contrasts")$race
[1] "contr.treatment"
```

```
Expected Value: E(Y|X)
   mean    sd   50%  2.5% 97.5%
1 0.773 0.011 0.773 0.752 0.794

Predicted Value: Y|X
      0     1
1 0.222 0.778
```

2. Simulating First Differences
   Estimating the first difference (and risk ratio) in individual's probability of voting when education is set to be low (25th percentile) versus high (75th percentile) while all the other variables held at their default values.

```
>  x.high <- setx(z.out, educate = quantile(turnout$educate, prob = 0.75))

> x.low <- setx(z.out, educate = quantile(turnout$educate, prob = 0.25))

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

> summary(s.out2)

Model:  logit.bayes
Number of simulations:  1000

Values of X
  (Intercept) racewhite educate
1           1         1      14
attr(,"assign")
[1] 0 1 2
attr(,"contrasts")
attr(,"contrasts")$race
[1] "contr.treatment"


Values of X1
  (Intercept) racewhite educate
1           1         1      10
attr(,"assign")
[1] 0 1 2
attr(,"contrasts")
attr(,"contrasts")$race
[1] "contr.treatment"


Expected Value: E(Y|X)
```

```
   mean    sd   50%  2.5% 97.5%
1 0.823 0.011 0.823 0.802 0.844

Predicted Value: Y|X
      0     1
1 0.185 0.815

Expected Value (for X1): E(Y|X1)
  mean    sd  50%  2.5% 97.5%
1 0.71 0.013 0.71 0.682 0.735

Predicted Value (for X1): Y|X1
      0     1
1 0.292 0.708

First Differences: E(Y|X1)-E(Y|X)
    mean    sd    50%   2.5% 97.5%
1 -0.113 0.012 -0.112 -0.138 -0.09
```

1.4 Model

Let Y_i be the binary dependent variable for observation i which takes the value of either 0 or 1.

- The *stochastic component* is given by

$$\begin{aligned} Y_i &\sim \text{Bernoulli}(\pi_\text{i}) \\ &= \pi_i^{Y_i}(1-\pi_i)^{1-Y_i}, \end{aligned}$$

  where $\pi_i = \Pr(Y_i = 1)$.

- The *systematic component* is given by

$$\pi_i = \frac{1}{1+\exp(-x_i\beta)},$$

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

- The *prior* for β is given by

$$\beta \sim \text{Normal}_k\left(b_0, B_0^{-1}\right)$$

  where b_0 is the vector of means for the k explanatory variables and B_0 is the $k \times k$ precision matrix (the inverse of a variance-covariance matrix).

1.5 Quantities of Interest

- The expected values (`qi$ev`) for the logit model are simulations of the predicted probability of a success:

$$E(Y) = \pi_i = \frac{1}{1 + \exp(-x_i\beta)},$$

given the posterior draws of β from the MCMC iterations.

- The predicted values (`qi$pr`) are draws from the Bernoulli distribution with mean equal to the simulated expected value π_i.

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

$$\text{FD} = \Pr(Y = 1 \mid X_1) - \Pr(Y = 1 \mid X).$$

- The risk ratio (`qi$rr`)is defined as

$$\text{RR} = \Pr(Y = 1 \mid X_1) \ / \ \Pr(Y = 1 \mid X).$$

- 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.

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

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

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

1.6 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 = "logit.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$coefficients`, and a default summary of information through `summary(z.out)`. Other elements available through the `$` operator are listed below.

- From the `zelig()` output object `z.out`, you may extract:
  - `coefficients`: draws from the posterior distributions of the estimated parameters.
  - `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$ev`: the simulated expected values(probabilities) for the specified values of `x`.
  - `qi$pr`: the simulated predicted values for the specified values of `x`.
  - `qi$fd`: the simulated first difference in the expected values for the values specified in `x` and `x1`.
  - `qi$rr`: the simulated risk ratio for the expected values simulated from `x` and `x1`.
  - `qi$att.ev`: the simulated average expected treatment effect for the treated from conditional prediction models.
  - `qi$att.pr`: the simulated average predicted treatment effect for the treated from conditional prediction models.

How to Cite the Zelig Software Package

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`.

Imai, Kosuke, Gary King, and Olivia Lau. (2008). "Toward A Common Framework for Statistical Analysis and Development." Journal of Computational and Graphical Statistics, Vol. 17, No. 4 (December), pp. 892-913.

See also

Bayesian logistic regression is part of the MCMCpack library by Andrew D. Martin and Kevin M. Quinn [1]. The convergence diagnostics are part of the CODA library by Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines [2].

2 mlogit.bayes: Bayesian Multinomial Logistic Regression

Use Bayesian multinomial logistic regression to model unordered categorical variables. The dependent variable may be in the format of either character strings or integer values. The model is estimated via a random walk Metropolis algorithm or a slice sampler. See Section **??** for the maximum-likelihood estimation of this model.

2.1 Syntax

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

2.2 Additional Inputs

`zelig()` accepts the following arguments for `mlogit.bayes`:

- `baseline`: either a character string or numeric value (equal to one of the observed values in the dependent variable) specifying a baseline category. The default value is `NA` which sets the baseline to the first alphabetical or numerical unique value of the dependent variable.

The model accepts the following additional arguments to monitor the Markov chains:

- `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.
- `mcmc.method`: either `"MH"` or `"slice"`, specifying whether to use Metropolis Algorithm or slice sampler. The default value is `"MH"`.
- `tune`: tuning parameter for the Metropolis-Hasting step, either a scalar or a numeric vector (for k coefficients, enter a k vector). The tuning parameter should be set such that the acceptance rate is satisfactory (between 0.2 and 0.5). The default value is 1.1.
- `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 a vector (for k coefficients, enter a k vector). The default is `NA` where the maximum likelihood estimates are used as the starting values.

Use the following arguments to specify the priors for the model:

- `b0`: prior mean for the coefficients, either a scalar or vector. If a scalar, that value will be the prior mean for all the coefficients. The default is 0.
- `B0`: prior precision parameter for the coefficients, either a square matrix with the dimensions equal to the number of 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.

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

2.3 Examples

1. Basic Example
   Attaching the sample dataset:

```
> data(mexico)
```

   Estimating multinomial logistics regression using `mlogit.bayes`:

```
> z.out <- zelig(vote88 ~ pristr + othcok + othsocok, model = "mlogit.bayes",
+                data = mexico)

Calculating MLEs and large sample var-cov matrix.
This may take a moment...
Inverting Hessian to get large sample var-cov matrix.


 How to cite this model in Zelig:
  NAMELESS AUTHOR. 2012.
  "mlogit.bayes: "
  in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
  http://gking.harvard.edu/zelig
```

   Checking for convergence before summarizing the estimates:

```
> heidel.diag(z.out$result$coefficients)

             Stationarity start     p-value
             test         iteration
(Intercept).2 passed       1         0.712
(Intercept).3 passed       1         0.263
pristr.2      passed       1         0.678
pristr.3      passed       1         0.530
```

```
othcok.2      passed       1         0.978
othcok.3      passed       1         0.258
othsocok.2    passed       1         0.633
othsocok.3    passed       1         0.357

              Halfwidth Mean   Halfwidth
              test
(Intercept).2 passed    -2.484 0.00822
(Intercept).3 passed    -2.882 0.00848
pristr.2      passed    -0.726 0.00196
pristr.3      passed    -0.601 0.00191
othcok.2      passed     1.109 0.00235
othcok.3      passed     1.250 0.00236
othsocok.2    passed     0.352 0.00325
othsocok.3    passed     0.302 0.00295
```

```
> raftery.diag(z.out$result$coefficients)

Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

               Burn-in  Total Lower bound  Dependence
               (M)      (N)   (Nmin)       factor (I)
 (Intercept).2 6        7195  3746         1.92
 (Intercept).3 8        9554  3746         2.55
 pristr.2      7        7534  3746         2.01
 pristr.3      8        10866 3746         2.90
 othcok.2      6        7984  3746         2.13
 othcok.3      6        8492  3746         2.27
 othsocok.2    5        5672  3746         1.51
 othsocok.3    6        9514  3746         2.54
```

```
> summary(z.out)

Call: MCMCmnl(formula = vote88 ~ pristr + othcok + othsocok, data = Data.frame,
    burnin = 1000, mcmc = 10000, verbose = 0)

Iterations = 1001:11000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 10000

 Mean, standard deviation, and quantiles for marginal posterior distributions.
                 Mean     SD    2.5%     50%   97.5%
(Intercept).2 -2.4837 0.4050 -3.2943 -2.4789 -1.6941
(Intercept).3 -2.8815 0.4032 -3.6919 -2.8763 -2.1031
```

```
pristr.2      -0.7259 0.0955 -0.9199 -0.7243 -0.5439
pristr.3      -0.6012 0.0932 -0.7871 -0.6009 -0.4170
othcok.2       1.1091 0.1146  0.8906  1.1066  1.3332
othcok.3       1.2495 0.1116  1.0298  1.2464  1.4777
othsocok.2     0.3521 0.1563  0.0503  0.3515  0.6604
othsocok.3     0.3021 0.1504  0.0133  0.3020  0.5960
```

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)

Model:  mlogit.bayes
Number of simulations:  1000

Values of X
  (Intercept)   pristr   othcok othsocok
1           1 1.966887 2.195732 1.395143
attr(,"assign")
[1] 0 1 2 3

Expected Value: E(Y|X)
  mean    sd   50%  2.5% 97.5%
 0.333 0.162 0.231 0.191 0.584

Predicted Value: Y|X
     1     2     3
 0.573 0.205 0.222
```

2. Simulating First Differences
   Estimating the first difference (and risk ratio) in the probabilities of voting different candidates when **pristr** (the strength of the PRI) is set to be weak (equal to 1) versus strong (equal to 3) while all the other variables held at their default values.

```
> x.weak <- setx(z.out, pristr = 1)
> x.strong <- setx(z.out, pristr = 3)

> s.out2 <- sim(z.out, x = x.strong, x1 = x.weak)

> summary(s.out2)
```

```
Model:  mlogit.bayes
Number of simulations:  1000

Values of X
  (Intercept) pristr   othcok othsocok
1           1      3 2.195732 1.395143
attr(,"assign")
[1] 0 1 2 3

Values of X1
  (Intercept) pristr   othcok othsocok
1           1      1 2.195732 1.395143
attr(,"assign")
[1] 0 1 2 3

Expected Value: E(Y|X)
  mean    sd   50%  2.5% 97.5%
 0.333 0.271 0.158 0.106 0.746

Predicted Value: Y|X
     1     2     3
 0.722 0.125 0.153

Expected Value (for X1): E(Y|X1)
  mean    sd   50% 2.5% 97.5%
 0.333 0.054 0.313 0.26 0.437

Predicted Value (for X1): Y|X1
     1     2     3
 0.408 0.297 0.295

First Differences
 mean    sd   50%   2.5% 97.5%
    0 0.224 0.133 -0.363 0.217
```

2.4 Model

Let Y_i be the (unordered) categorical dependent variable for observation i which takes an integer values $j = 1, \ldots, J$.

- The *stochastic component* is given by:

$$Y_i \sim \text{Multinomial}(Y_i \mid \pi_{ij}).$$

  where $\pi_{ij} = \Pr(Y_i = j)$ for $j = 1, \ldots, J$.

- The *systematic component* is given by

$$\pi_{ij} = \frac{\exp(x_i\beta_j)}{\sum_{k=1}^{J}\exp(x_i\beta_k)}, \text{ for } j = 1, \ldots, J-1,$$

where x_i is the vector of k explanatory variables for observation i and β_j is the vector of coefficient for category j. Category J is assumed to be the baseline category.

- The *prior* for β is given by

$$\beta_j \sim \text{Normal}_k\left(b_0, B_0^{-1}\right) \text{ for } j = 1, \ldots, J-1,$$

where b_0 is the vector of means for the k explanatory variables and B_0 is the $k \times k$ precision matrix (the inverse of a variance-covariance matrix).

2.5 Quantities of Interest

- The expected values (`qi$ev`) for the multinomial logistics regression model are the predicted probability of belonging to each category:

$$\Pr(Y_i = j) = \pi_{ij} = \frac{\exp(x_i\beta_j)}{\sum_{k=1}^{J}\exp(x_J\beta_k)}, \quad \text{for } j = 1, \ldots, J-1,$$

and

$$\Pr(Y_i = J) = 1 - \sum_{j=1}^{J-1}\Pr(Y_i = j)$$

given the posterior draws of β_j for all categories from the MCMC iterations.

- The predicted values (`qi$pr`) are the draws of Y_i from a multinomial distribution whose parameters are the expected values(`qi$ev`) computed based on the posterior draws of β from the MCMC iterations.

- The first difference (`qi$fd`) in category j for the multinomial logistic model is defined as

$$\text{FD}_j = \Pr(Y_i = j \mid X_1) - \Pr(Y_i = j \mid X).$$

- The risk ratio (`qi$rr`) in category j is defined as

$$\text{RR}_j = \Pr(Y_i = j \mid X_1) \;/\; \Pr(Y_i = j \mid X).$$

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

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

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups, and n_j is the number of treated observations in category j.

- In conditional prediction models, the average predicted treatment effect (`qi$att.pr`) for the treatment group in category j is

$$\frac{1}{n_j} \sum_{i:t_i=1}^{n_j} [Y_i(t_i = 1) - Y_i(\widehat{t_i = 0})],$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups, and n_j is the number of treated observations in category j.

2.6 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 = "mlogit.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 coefficients β for each category except the baseline category.
  - `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$ev`: the simulated expected values(probabilities) of each of the J categories given the specified values of `x`.
  - `qi$pr`: the simulated predicted values drawn from the multinomial distribution defined by the expected values(`qi$ev`) given the specified values of `x`.
  - `qi$fd`: the simulated first difference in the expected values of each of the J categories for the values specified in `x` and `x1`.
  - `qi$rr`: the simulated risk ratio for the expected values of each of the J categories simulated from `x` and `x1`.
  - `qi$att.ev`: the simulated average expected treatment effect for the treated from conditional prediction models.
  - `qi$att.pr`: the simulated average predicted treatment effect for the treated from conditional prediction models.

How to Cite the Bayesian Multinomial Logit Model

Ben Goodrich and Ying Lu. 2007. "mlogit.bayes: Bayesian Multinomial Logistic Regression for Dependent Variables with Unordered Categorical Values ," in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software," `http://gking.harvard.edu/zelig`.

How to Cite the Zelig Software Package

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`.

Imai, Kosuke, Gary King, and Olivia Lau. (2008). "Toward A Common Framework for Statistical Analysis and Development." Journal of Computational and Graphical Statistics, Vol. 17, No. 4 (December), pp. 892-913.

See also

Bayesian logistic regression is part of the MCMCpack library by Andrew D. Martin and Kevin M. Quinn [1]. The convergence diagnostics are part of the CODA library by Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines [2].

3 `normal.bayes`: Bayesian Normal Linear Regression

Use Bayesian regression to specify a continuous dependent variable as a linear function of specified explanatory variables. The model is implemented using a Gibbs sampler. See Section **??** for the maximum-likelihood implementation or Section **??** for the ordinary least squares variation.

3.1 Syntax

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

3.2 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`, which uses the least squares estimates as the starting values.

Use the following arguments 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 the 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(MCMCregress)` for more information.

3.3 Examples

1. Basic Example
   Attaching the sample dataset:

```
> data(macro)
```

   Estimating linear regression using `normal.bayes`:

```
> z.out <- zelig(unem ~ gdp + capmob + trade, model = "normal.bayes",
+                   data = macro, verbose = FALSE)

 How to cite this model in Zelig:
  NAMELESS AUTHOR. 2012.
  "normal.bayes: "
  in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
  http://gking.harvard.edu/zelig
```

Checking for convergence before summarizing the estimates:

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

Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5

(Intercept)         gdp      capmob       trade      sigma2
    -0.1178     -0.3130     -0.5891      0.5155     -1.6965

> heidel.diag(z.out$result$coefficients)

            Stationarity start     p-value
            test         iteration
(Intercept) passed       1         0.756
gdp         passed       1         0.963
capmob      passed       1         0.355
trade       passed       1         0.339
sigma2      passed       1         0.626

            Halfwidth Mean    Halfwidth
            test
(Intercept) passed     6.1773 0.008849
gdp         passed    -0.3240 0.001244
capmob      passed     1.4207 0.003246
trade       passed     0.0199 0.000105
sigma2      passed     7.5834 0.011691

> raftery.diag(z.out$result$coefficients)

Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

             Burn-in  Total Lower bound  Dependence
             (M)      (N)   (Nmin)       factor (I)
 (Intercept) 2        3834  3746         1.020
 gdp         2        3650  3746         0.974
 capmob      2        3771  3746         1.010
 trade       2        3680  3746         0.982
 sigma2      2        3710  3746         0.990

> summary(z.out)

Call: MCMCregress(formula = unem ~ gdp + capmob + trade, data = Data.frame,
    verbose = FALSE, burnin = 1000, mcmc = 10000)
```

```
Iterations = 1001:11000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 10000

 Mean, standard deviation, and quantiles for marginal posterior distributions.
              Mean     SD    2.5%     50%   97.5%
(Intercept)  6.1773 0.4515  5.3093  6.1797  7.0878
gdp         -0.3240 0.0635 -0.4504 -0.3237 -0.2012
capmob       1.4207 0.1656  1.0927  1.4215  1.7461
trade        0.0199 0.0056  0.0087  0.0200  0.0308
sigma2       7.5834 0.5785  6.5346  7.5526  8.7957
```

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)

Model:  normal.bayes
Number of simulations:  1000

Values of X
  (Intercept)      gdp     capmob    trade
1           1 3.254223 -0.8914286 57.07625
attr(,"assign")
[1] 0 1 2 3

Expected Value: E(Y|X)
   mean    sd   50%  2.5% 97.5%
1 4.994 0.148 4.992 4.708 5.283

Predicted Value: Y|X
  mean   sd   50%   2.5%  97.5%
 4.948 2.77 4.905 -0.409 10.443
```

2. Simulating First Differences
   Set explanatory variables to their default(mean/mode) values, with high (80th percentile) and low (20th percentile) trade on GDP:

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

Estimating the first difference for the effect of high versus low trade on unemployment rate:

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

> summary(s.out2)

Model:  normal.bayes
Number of simulations:  1000

Values of X
  (Intercept)      gdp     capmob    trade
1           1 3.254223 -0.8914286 79.10131
attr(,"assign")
[1] 0 1 2 3

Values of X1
  (Intercept)      gdp     capmob    trade
1           1 3.254223 -0.8914286 37.29106
attr(,"assign")
[1] 0 1 2 3

Expected Value: E(Y|X)
   mean    sd   50%  2.5% 97.5%
1 5.433 0.192 5.433 5.062 5.815

Predicted Value: Y|X
  mean    sd   50% 2.5%  97.5%
 5.399 2.757 5.399 0.03 10.868

Expected Value (for X1): E(Y|X1)
  mean    sd   50%  2.5% 97.5%
1  4.6 0.185 4.601 4.241 4.966

Predicted Value (for X1): Y|X1
  mean    sd   50%   2.5%  97.5%
 4.608 2.763 4.594 -0.744 10.086

First Differences: E(Y|X1) - E(Y|X)
    mean    sd    50%   2.5%  97.5%
1 -0.833 0.235 -0.836 -1.287 -0.365
```

3.4 Model

- 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 to be *a priori* independent.

3.5 Quantities of Interest

- The expected values (`qi$ev`) for the linear regression model are calculated as following:

$$E(Y) = x_i \beta,$$

given posterior draws of β based on the MCMC iterations.

- The first difference (`qi$fd`) for the linear 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

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

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

- In conditional prediction models, the average predicted treatment effect (`att.pr`) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{Y_i(t_i = 1) - Y_i(\widehat{t_i = 0})\right\},$$

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

3.6 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 = "normal.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$ev`: the simulated expected values for the specified values of `x`.
  - `qi$fd`: the simulated first difference in the expected values for 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 the Bayesian Gaussian Model

Ben Goodrich and Ying Lu. 2007. "normal.bayes: Bayesian Normal Linear Regression ," in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software," `http://gking.harvard.edu/zelig`.

How to Cite the Zelig Software Package

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`.

Imai, Kosuke, Gary King, and Olivia Lau. (2008). "Toward A Common Framework for Statistical Analysis and Development." Journal of Computational and Graphical Statistics, Vol. 17, No. 4 (December), pp. 892-913.

See also

Bayesian normal regression is part of the MCMCpack library by Andrew D. Martin and Kevin M. Quinn [1]. The convergence diagnostics are part of the CODA library by Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines [2].

4 `oprobit.bayes`: Bayesian Ordered Probit Regression

Use the ordinal probit regression model if your dependent variables are ordered and categorical. They may take either integer values or character strings. The model is estimated using a Gibbs sampler with data augmentation. For a maximum-likelihood implementation of this models, see Section **??**.

4.1 Syntax

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

4.2 Additional Inputs

`zelig()` accepts the following arguments to monitor 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 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.
- `tune`: tuning parameter for the Metropolis-Hasting step. The default value is `NA` which corresponds to 0.05 divided by the number of categories in the response variable.
- `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 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`, which uses the maximum likelihood estimates 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 value, that value will be the prior mean for all the coefficients. The default is 0.
- `B0`: prior precision parameter for the coefficients, either a square matrix (with dimensions equal to the number of coefficients) or a scalar. If a scalar value, that value times an identity matrix will be the prior precision parameter. The default is 0 which leads to an improper prior.

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

4.3 Examples

1. Basic Example
   Attaching the sample dataset:

```
>  data(sanction)
```

   Estimating ordered probit regression using `oprobit.bayes`:

```
>  z.out <- zelig(ncost ~ mil + coop, model = "oprobit.bayes",
+                   data = sanction, verbose = FALSE)

The following object(s) are masked from 'package:MASS':

    coop


 How to cite this model in Zelig:
  Skyler Cranmer. 2012.
  "oprobit.bayes: Ordinal Probit Regression for Bayesian Models"
  in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
  http://gking.harvard.edu/zelig
```

   Creating an ordered dependent variable:

```
> sanction$ncost <- factor(sanction$ncost, ordered = TRUE,
+                          levels = c("net gain", "little effect",
+                          "modest loss", "major loss"))
```

   Checking for convergence before summarizing the estimates:

```
> heidel.diag(z.out$result$coefficients)

            Stationarity start     p-value
            test         iteration
(Intercept) passed       3001      0.190
mil         passed          1      0.949
```

```
coop        passed           1      0.390
gamma2      passed           1      0.909
gamma3      passed           1      0.143

            Halfwidth Mean   Halfwidth
            test
(Intercept) passed     0.685 0.01225
mil         passed    -0.285 0.01296
coop        passed    -0.298 0.00395
gamma2      passed     0.114 0.00986
gamma3      passed     0.431 0.03857
```

```
> raftery.diag(z.out$result$coefficients)

Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

             Burn-in  Total Lower bound  Dependence
             (M)      (N)   (Nmin)       factor (I)
 (Intercept) 3        4302  3746          1.15
 mil         4        4674  3746          1.25
 coop        3        4338  3746          1.16
 gamma2      26       28642 3746          7.65
 gamma3      88       87948 3746         23.50
```

```
> summary(z.out)

Call: MCMCoprobit(formula = ncost ~ mil + coop, data = Data.frame,
    verbose = FALSE, burnin = 1000, mcmc = 10000)

Iterations = 1001:11000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 10000

 Mean, standard deviation, and quantiles for marginal posterior distributions.
               Mean     SD    2.5%     50%   97.5%
(Intercept)  0.6954 0.2962  0.1133  0.6967  1.2756
mil         -0.2850 0.4666 -1.2066 -0.2824  0.6228
coop        -0.2980 0.1421 -0.5856 -0.2966 -0.0222
gamma2       0.1136 0.0534  0.0322  0.1061  0.2386
gamma3       0.4308 0.1064  0.2492  0.4280  0.6800
```

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)

Model:  oprobit.bayes
Number of simulations:  1000

Values of X
  (Intercept)       mil     coop
1           1 0.1025641 1.807692
attr(,"assign")
[1] 0 1 2

Expected Value: E(Y|X)
               mean    sd   50%  2.5% 97.5%
little effect 0.450 0.056 0.449 0.342 0.561
major loss    0.045 0.021 0.042 0.013 0.094
modest loss   0.123 0.040 0.120 0.060 0.228
net gain      0.382 0.055 0.381 0.276 0.492

Predicted Value: Y|X
 little effect major loss modest loss net gain
         0.187      0.273       0.521    0.019
```

2. Simulating First Differences
   Estimating the first difference (and risk ratio) in the probabilities of incurring different level of cost when there is no military action versus military action while all the other variables held at their default values.

```
>  x.high <- setx(z.out, mil=0)
>  x.low <- setx(z.out, mil=1)

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

Model:  oprobit.bayes
Number of simulations:  1000

Values of X
  (Intercept) mil     coop
1           1   0 1.807692
attr(,"assign")
[1] 0 1 2

Values of X1
```

```
  (Intercept) mil     coop
1           1   1 1.807692
attr(,"assign")
[1] 0 1 2

Expected Value: E(Y|X)
               mean    sd   50%  2.5% 97.5%
little effect 0.438 0.058 0.438 0.328 0.554
major loss    0.045 0.021 0.042 0.013 0.094
modest loss   0.124 0.040 0.120 0.060 0.229
net gain      0.393 0.058 0.392 0.283 0.508

Predicted Value: Y|X
 little effect major loss modest loss net gain
         0.149      0.238       0.578    0.035

Expected Value (for X1): E(Y|X1)
               mean    sd   50%  2.5% 97.5%
little effect 0.546 0.161 0.548 0.235 0.845
major loss    0.041 0.020 0.038 0.011 0.088
modest loss   0.108 0.040 0.104 0.042 0.201
net gain      0.305 0.145 0.290 0.074 0.623

Predicted Value (for X1): Y|X1
 little effect major loss modest loss net gain
         0.612      0.096       0.186    0.106

First Differences: E(Y|X1) - E(Y|X)
                mean    sd    50%   2.5% 97.5%
little effect  0.108 0.170  0.111 -0.227 0.426
major loss    -0.004 0.007 -0.001 -0.023 0.002
modest loss   -0.016 0.022 -0.008 -0.076 0.005
net gain      -0.088 0.153 -0.102 -0.345 0.241
```

4.4 Model

Let Y_i be the ordered categorical dependent variable for observation i which takes an integer value $j = 1, \ldots, J$.

- The *stochastic component* is described by an unobserved continuous variable, Y_i^*,

$$Y_i^* \sim \text{Normal}(\mu_i, 1).$$

Instead of Y_i^*, we observe categorical variable Y_i,

$$Y_i = j \quad \text{if } \tau_{j-1} \leq Y_i^* \leq \tau_j \text{ for } j = 1, \ldots, J.$$

where τ_j for $j = 0, \ldots, J$ are the threshold parameters with the following constraints, $\tau_l < \tau_m$ for $l < m$, and $\tau_0 = -\infty, \tau_J = \infty$.

The probability of observing Y_i equal to category j is,

$$\Pr(Y_i = j) \quad = \quad \Phi(\tau_j \mid \mu_i) - \Phi(\tau_{j-1} \mid \mu_i) \text{ for } j = 1, \ldots, J$$

where $\Phi(\cdot \mid \mu_i)$ is the cumulative distribution function of the Normal distribution with mean μ_i and variance 1.

- 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 *prior* for β is given by

$$\beta \sim \text{Normal}_k \left(b_0, B_0^{-1} \right)$$

where b_0 is the vector of means for the k explanatory variables and B_0 is the $k \times k$ precision matrix (the inverse of a variance-covariance matrix).

4.5 Quantities of Interest

- The expected values (`qi$ev`) for the ordered probit model are the predicted probability of belonging to each category:

$$\Pr(Y_i = j) = \Phi(\tau_j \mid x_i \beta) - \Phi(\tau_{j-1} \mid x_i \beta),$$

given the posterior draws of β and threshold parameters τ from the MCMC iterations.

- The predicted values (`qi$pr`) are the observed values of Y_i given the observation scheme and the posterior draws of β and cut points τ from the MCMC iterations.

- The first difference (`qi$fd`) in category j for the ordered probit model is defined as

$$\text{FD}_j = \Pr(Y_i = j \mid X_1) - \Pr(Y_i = j \mid X).$$

- The risk ratio (`qi$rr`) in category j is defined as

$$\text{RR}_j = \Pr(Y_i = j \mid X_1) \ / \ \Pr(Y_i = j \mid X).$$

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

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

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups, and n_j is the number of observations in the treatment group that belong to category j.

- In conditional prediction models, the average predicted treatment effect (`qi$att.pr`) for the treatment group in category j is

$$\frac{1}{n_j} \sum_{i:t_i=1}^{n_j} [Y_i(t_i = 1) - Y_i(\widehat{t_i = 0})],$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups, and n_j is the number of observations in the treatment group that belong to category j.

4.6 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 = "oprobit.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 coefficients β and threshold parameters τ. Note, element τ_1 is normalized to 0 and is not returned in the `coefficients` object.
  - `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$ev`: the simulated expected values (probabilities) of each of the J categories for the specified values of `x`.
  - `qi$pr`: the simulated predicted values (observed values) for the specified values of `x`.

- `qi$fd`: the simulated first difference in the expected values of each of the J categories for the values specified in `x` and `x1`.
- `qi$rr`: the simulated risk ratio for the expected values of each of the J categories simulated from `x` and `x1`.
- `qi$att.ev`: the simulated average expected treatment effect for the treated from conditional prediction models.
- `qi$att.pr`: the simulated average predicted treatment effect for the treated from conditional prediction models.

How to Cite the *oprobit.bayes* Zelig Model

Ben Goodrich and Ying Lu. 2007. "oprobit.bayes: Bayesian Ordered Probit Regression," in Kosuke Imai,Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software," `http://gking.harvard.edu/zelig`.

How to Cite the Zelig Software Package

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`.

Imai, Kosuke, Gary King, and Olivia Lau. (2008). "Toward A Common Framework for Statistical Analysis and Development." Journal of Computational and Graphical Statistics, Vol. 17, No. 4 (December), pp. 892-913.

See also

Bayesian ordinal probit regression is part of the MCMCpack library by Andrew D. Martin and Kevin M. Quinn [1]. The convergence diagnostics are part of the CODA library by Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines [2].

5 `poisson.bayes`: Bayesian Poisson Regression

Use the Poisson regression model if the observations of your dependent variable represents the number of independent events that occur during a fixed period of time. The model is fit using a random walk Metropolis algorithm. For a maximum-likelihood estimation of this model see Section **??**.

5.1 Syntax

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

5.2 Additional Inputs

Use the following argument to monitor 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.
- `tune`: Metropolis tuning parameter, either a positive scalar or a vector of length k, where k is the number of coefficients. The tuning parameter should be set such that the acceptance rate of the Metropolis algorithm is satisfactory (typically between 0.20 and 0.5). The default value is 1.1.
- `verbose`: default 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 maximum likelihood 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 the 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.

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

5.3 Examples

1. Basic Example
   Attaching the sample dataset:

```
> data(sanction)
```

   Estimating the Poisson regression using `poisson.bayes`:

```
> z.out <- zelig(num ~ target + coop, model = "poisson.bayes",
+                   data = sanction, verbose = FALSE)

The following object(s) are masked from 'package:MASS':

    coop


 How to cite this model in Zelig:
  NAMELESS AUTHOR. 2012.
  "poisson.bayes: "
  in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
  http://gking.harvard.edu/zelig
```

   Checking convergence diagnostics before summarizing the estimates:

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

Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5

(Intercept)      target        coop
  2.5342332  -0.0007325  -2.3642654

> heidel.diag(z.out$result$coefficients)

            Stationarity start     p-value
            test         iteration
(Intercept) passed       1         0.107
target      passed       1         0.852
coop        passed       1         0.175

            Halfwidth Mean    Halfwidth
            test
(Intercept) passed    -0.9798 0.01096
target      failed    -0.0176 0.00370
coop        passed     1.2109 0.00298

> raftery.diag(z.out$result$coefficients)
```

```
Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

             Burn-in  Total Lower bound  Dependence
             (M)      (N)   (Nmin)       factor (I)
 (Intercept) 21       22551 3746         6.02
 target      20       21849 3746         5.83
 coop        20       21706 3746         5.79
```

```
> summary(z.out)

Call: MCMCpoisson(formula = num ~ target + coop, data = Data.frame,
    verbose = FALSE, burnin = 1000, mcmc = 10000)

Iterations = 1001:11000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 10000

 Mean, standard deviation, and quantiles for marginal posterior distributions.
               Mean     SD    2.5%     50%   97.5%
(Intercept) -0.9798 0.1746 -1.3310 -0.9777 -0.6456
target      -0.0176 0.0567 -0.1299 -0.0195  0.0944
coop         1.2109 0.0470  1.1208  1.2101  1.3066
```

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)

Model:  poisson.bayes
Number of simulations:  1000

Values of X
  (Intercept)   target     coop
1           1 2.141026 1.807692
attr(,"assign")
[1] 0 1 2

Expected Value: E(Y|X)
   mean    sd   50%  2.5% 97.5%
```

```
1 3.235 0.238 3.235 2.777 3.705

Predicted Value: Y|X
   mean    sd 50% 2.5% 97.5%
1 3.233 1.819   3    0     7
```

2. Simulating First Differences
   Estimating the first difference in the number of countries imposing sanctions when the number of targets is set to be its maximum versus its minimum :

```
> x.max <- setx(z.out, target = max(sanction$target))
> x.min <- setx(z.out, target = min(sanction$target))

> s.out2 <- sim(z.out, x = x.max, x1 = x.min)
> summary(s.out2)

Model:  poisson.bayes
Number of simulations:  1000

Values of X
  (Intercept) target     coop
1           1      3 1.807692
attr(,"assign")
[1] 0 1 2

Values of X1
  (Intercept) target     coop
1           1      1 1.807692
attr(,"assign")
[1] 0 1 2

Expected Value: E(Y|X)
   mean    sd   50%  2.5% 97.5%
1 3.192 0.294 3.183 2.642 3.804

Predicted Value: Y|X
   mean    sd 50% 2.5% 97.5%
1 3.189 1.842   3    0     7

Expected Value (for X1): E(Y|X1)
   mean    sd 50%  2.5% 97.5%
1 3.306 0.306 3.3 2.729 3.944

Predicted Value (for X1): Y|X1
   mean    sd 50% 2.5% 97.5%
1 3.316 1.857   3    0     7
```

```
First Differences: E(Y|X1) - E(Y|X)
   mean    sd   50%   2.5% 97.5%
1 0.115 0.367 0.127 -0.607 0.834
```

5.4 Model

Let Y_i be the number of independent events that occur during a fixed time period.

- The *stochastic component* is given by

$$Y_i \sim \text{Poisson}(\lambda_i)$$

  where λ_i is the mean and variance parameter.

- The *systematic component* is given by

$$\lambda_i = \exp(x_i\beta)$$

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

- The *prior* for β is given by

$$\beta \sim \text{Normal}_k\left(b_0, B_0^{-1}\right)$$

  where b_0 is the vector of means for the k explanatory variables and B_0 is the $k \times k$ precision matrix (the inverse of a variance-covariance matrix).

5.5 Quantities of Interest

- The expected values (`qi$ev`) for the Poisson model are calculated as following:

$$E(Y \mid X) = \lambda_i = \exp(x_i\beta),$$

  given the posterior draws of β based on the MCMC iterations.

- The predicted values (`qi$pr`) are draws from the Poisson distribution with parameter λ_i.

- The first difference (`qi$fd`) for the Poisson 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

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

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

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

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1} [Y_i(t_i = 1) - Y_i(\widehat{t_i = 0})],$$

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

5.6 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 = "poisson.bayes", data)
```

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.
  - `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$ev`: the simulated expected values for the specified values of `x`.
  - `qi$pr`: the simulated predicted values for the specified values of `x`.
  - `qi$fd`: the simulated first difference in the expected values for the values specified in `x` and `x1`.
  - `qi$att.ev`: the simulated average expected treatment effect for the treated from conditional prediction models.
  - `qi$att.pr`: the simulated average predicted treatment effect for the treated from conditional prediction models.

How to Cite the *poisson.bayes* Zelig Model

Ben Goodrich and Ying Lu. 2007. "poisson.bayes: Bayesian Poisson Regression," in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software," `http://gking.harvard.edu/zelig`.

How to Cite the Zelig Software Package

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`.
>
> Imai, Kosuke, Gary King, and Olivia Lau. (2008). "Toward A Common Framework for Statistical Analysis and Development." Journal of Computational and Graphical Statistics, Vol. 17, No. 4 (December), pp. 892-913.

See also

Bayesian poisson regression is part of the MCMCpack library by Andrew D. Martin and Kevin M. Quinn [1]. The convergence diagnostics are part of the CODA library by Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines [2].

6 `probit.bayes`: Bayesian Probit Regression

Use the probit regression model for model binary dependent variables specified as a function of a set of explanatory variables. The model is estimated using a Gibbs sampler. For other models suitable for binary response variables, see Bayesian logistic regression(Section 1), maximum likelihood logit regression (Section **??**), and maximum likelihood probit regression (Section **??**).

6.1 Syntax

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

6.2 Additional Inputs

Using the following arguments to monitor the Markov chains:

- `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 maximum likelihood 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 value, that value will be the prior mean for all the 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 value, that value times an identity matrix will be the prior precision parameter. The default is 0, which leads to an improper prior.

Use the following arguments to specify optional output for the model:

- `bayes.resid`: defaults to `FALSE`. If `TRUE`, the latent Bayesian residuals for all observations are returned. Alternatively, users can specify a vector of observations for which the latent residuals should be returned.

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

6.3 Examples

1. Basic Example
   Attaching the sample dataset:

```
> data(turnout)
```

   Estimating the probit regression using `probit.bayes`:

```
> z.out <- zelig(vote ~ race + educate, model = "probit.bayes",
+                   data = turnout, verbose = FALSE)

 How to cite this model in Zelig:
  NAMELESS AUTHOR. 2012.
  "probit.bayes: "
  in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
  http://gking.harvard.edu/zelig
```

   Checking for convergence before summarizing the estimates:

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

```
Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5

(Intercept)   racewhite     educate
    -0.3037     -0.8835      0.7866

> heidel.diag(z.out$result$coefficients)

            Stationarity start     p-value
            test         iteration
(Intercept) passed       1         0.1891
racewhite   passed       1         0.7231
educate     passed       1         0.0785

            Halfwidth Mean    Halfwidth
            test
(Intercept) passed    -0.7327 0.004195
racewhite   passed     0.2989 0.002683
educate     passed     0.0977 0.000331

> raftery.diag(z.out$result$coefficients)

Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

             Burn-in  Total Lower bound  Dependence
             (M)      (N)   (Nmin)       factor (I)
 (Intercept) 4        4954  3746         1.32
 racewhite   4        4832  3746         1.29
 educate     4        5080  3746         1.36

> summary(z.out)

Call: MCMCprobit(formula = vote ~ race + educate, data = Data.frame,
    verbose = FALSE, burnin = 1000, mcmc = 10000)

Iterations = 1001:11000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 10000

 Mean, standard deviation, and quantiles for marginal posterior distributions.
              Mean     SD    2.5%     50%   97.5%
(Intercept) -0.7327 0.1281 -0.9854 -0.7325 -0.4826
racewhite    0.2989 0.0848  0.1359  0.2983  0.4671
educate      0.0977 0.0096  0.0791  0.0976  0.1166
```

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)

Model:  probit.bayes
Number of simulations:  1000

Values of X
  (Intercept) racewhite  educate
1           1         1 12.06675
attr(,"assign")
[1] 0 1 2
attr(,"contrasts")
attr(,"contrasts")$race
[1] "contr.treatment"


Expected Value: E(Y|X)
   mean   sd   50%  2.5% 97.5%
1 0.772 0.01 0.772 0.751 0.791

Predicted Value: Y|X
      0     1
1 0.223 0.777
```

2. Simulating First Differences
   Estimating the first difference (and risk ratio) in individual's probability of voting when education is set to be low (25th percentile) versus high (75th percentile) while all the other variables are held at their default values:

```
>  x.high <- setx(z.out, educate = quantile(turnout$educate, prob = 0.75))
>  x.low <- setx(z.out, educate = quantile(turnout$educate, prob = 0.25))

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

> summary(s.out2)

Model:  probit.bayes
Number of simulations:  1000

Values of X
```

```
  (Intercept) racewhite educate
1           1         1      14
attr(,"assign")
[1] 0 1 2
attr(,"contrasts")
attr(,"contrasts")$race
[1] "contr.treatment"


Values of X1
  (Intercept) racewhite educate
1           1         1      10
attr(,"assign")
[1] 0 1 2
attr(,"contrasts")
attr(,"contrasts")$race
[1] "contr.treatment"


Expected Value: E(Y|X)
   mean   sd   50%  2.5% 97.5%
1 0.825 0.01 0.825 0.803 0.844

Predicted Value: Y|X
      0     1
1 0.175 0.825

Expected Value (for X1): E(Y|X1)
   mean    sd   50%  2.5% 97.5%
1 0.706 0.013 0.706 0.681 0.731

Predicted Value (for X1): Y|X1
      0     1
1 0.296 0.704

First Differences: E(Y|X1)-E(Y|X)
    mean    sd    50%   2.5%  97.5%
1 -0.118 0.012 -0.118 -0.141 -0.096
```

6.4 Model

Let Y_i be the binary dependent variable for observation i which takes the value of either 0 or 1.

- The *stochastic component* is given by

$$\begin{aligned} Y_i &\sim \text{Bernoulli}(\pi_i) \\ &= \pi_i^{Y_i}(1-\pi_i)^{1-Y_i}, \end{aligned}$$

  where $\pi_i = \Pr(Y_i = 1)$.

- The *systematic component* is given by

$$\pi_i = \Phi(x_i\beta),$$

  where $\Phi(\cdot)$ is the cumulative density function of the standard Normal distribution with mean 0 and variance 1, x_i is the vector of k explanatory variables for observation i, and β is the vector of coefficients.

- The *prior* for β is given by

$$\beta \sim \text{Normal}_k\left(b_0, B_0^{-1}\right)$$

  where b_0 is the vector of means for the k explanatory variables and B_0 is the $k \times k$ precision matrix (the inverse of a variance-covariance matrix).

6.5 Quantities of Interest

- The expected values (`qi$ev`) for the probit model are the predicted probability of a success:

$$E(Y \mid X) = \pi_i = \Phi(x_i\beta),$$

  given the posterior draws of β from the MCMC iterations.

- The predicted values (`qi$pr`) are draws from the Bernoulli distribution with mean equal to the simulated expected value π_i.

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

$$\text{FD} = \Pr(Y = 1 \mid X_1) - \Pr(Y = 1 \mid X).$$

- The risk ratio (`qi$rr`)is defined as

$$\text{RR} = \Pr(Y = 1 \mid X_1) \;/\; \Pr(Y = 1 \mid X).$$

- 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.

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

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

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

6.6 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 = "probit.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.
  - `zelig.data`: the input data frame if `save.data = TRUE`.
  - `bayes.residuals`: When `bayes.residual` is `TRUE` or a set of observation numbers is given, this object contains the posterior draws of the latent Bayesian residuals of all the observations or the observations specified by the user.
  - `seed`: the random seed used in the model.
- From the `sim()` output object `s.out`:
  - `qi$ev`: the simulated expected values (probabilities) for the specified values of `x`.
  - `qi$pr`: the simulated predicted values for the specified values of `x`.
  - `qi$fd`: the simulated first difference in the expected values for the values specified in `x` and `x1`.
  - `qi$rr`: the simulated risk ratio for the expected values simulated from `x` and `x1`.
  - `qi$att.ev`: the simulated average expected treatment effect for the treated from conditional prediction models.
  - `qi$att.pr`: the simulated average predicted treatment effect for the treated from conditional prediction models.

How to Cite the *probit.bayes* Zelig model

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

How to Cite the Zelig Software Package

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`.

Imai, Kosuke, Gary King, and Olivia Lau. (2008). "Toward A Common Framework for Statistical Analysis and Development." Journal of Computational and Graphical Statistics, Vol. 17, No. 4 (December), pp. 892-913.

See also

Bayesian probit regression is part of the MCMCpack library by Andrew D. Martin and Kevin M. Quinn [1]. The convergence diagnostics are part of the CODA library by Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines [2].

References

[1] Andrew D. Martin and Kevin M. Quinn. *MCMCpack: Markov chain Monte Carlo (MCMC) Package*, 2005.

[2] Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines. *coda: Output analysis and diagnostics for MCMC*, 2005.