# Package 'phcfM'

April 9, 2013

**Version** 1.2

**Type** Package

**Date** 2012-06-13

**Title** Modelling anthropogenic deforestation

**Depends** coda

**Author** Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

**Maintainer** Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

**Description** phcfM is an R package for modelling anthropogenic deforestation. It was initially developed to obtain REDD+ baseline scenarios of deforestation for the ''programme holistique de conservation des forets a Madagascar'' (from which the package is named after). Parameter inference is done in a hierarchical Bayesian framework. Markov chains Monte Carlo (MCMC) are coded in C++ using the Scythe statistical library to maximize computation efficiency.

**License** GPL-3 | file LICENSE

**URL** http://phcfM.sf.net

**LazyLoad** yes

## R topics documented:

## phcfM-package *Modelling anthropogenic deforestation in tropical forests*

### Description

phcfM is an R package for modelling anthropogenic deforestation. It was initially developed to obtain REDD baseline scenarios of deforestation for the "programme holistique de conservation des forets a Madagascar" (from which the package is named after). It includes two main functions: (i) demography(), to model the population growth with time using population census data and Gaussian mixed linear regressions and (ii) deforestation(), to model the deforestation process using land-cover change data and Binary logistic regressions with variable time-intervals between land-cover observations. Parameter inference is done in a hierarchical Bayesian framework. Markov chains Monte Carlo (MCMC) are coded in C++ using the Scythe statistical library to maximize computation efficiency.

### Details

| | |
|---|---|
| Package: | phcfM |
| Type: | Package |
| Version: | 1.1 |
| Date: | 2012-06-13 |
| License: | GPL-3 |
| LazyLoad: | yes |

### Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

### References

Ghislain Vieilledent, Clovis Grinand and Romuald Vaudry. 2013. Forecasting deforestation and carbon emissions in tropical developing countries facing demographic expansion: a case study in Madagascar. *Ecology and Evolution*. DOI: 10.1002/ece3.550

## deforestation *Binary logistic regression with variable time-intervals between observations for modelling deforestation*

## Description

The `deforestation()` function estimates the parameters of a Binary logistic regression model with variable time-intervals between observations in a hierarchical Bayesian framework. To estimate the posterior distribution of the parameters, an adaptive Metropolis algorithm is used. The user supplies data and priors and a sample from the posterior distribution is returned as an MCMC object, which can be subsequently analyzed with functions provided in the `coda` package.

## Usage

```
deforestation(formula, interval=1, data, burnin=1000, mcmc=1000,
   thin=1, verbose=1, seed=NA, tune=1, beta.start=NA, mubeta=0,
   Vbeta=1.0E6)
```

## Arguments

| | |
|---|---|
| `formula` | A two-sided linear formula of the form 'y~x1+...+xp' describing the model, with the response on the left of a '~' operator and the p fixed terms, separated by '+' operators, on the right. Response variable y must be 0 or 1 (Bernoulli trials). |
| `interval` | A numeric scalar or a vector of length equal to the number of observations. `interval` specifies the time interval between land cover observations. Default to 1. |
| `data` | A data frame containing the variables of the model. |
| `burnin` | The number of burnin iterations for the sampler. |
| `mcmc` | The number of Gibbs iterations for the sampler. Total number of Gibbs iterations is equal to `burnin+mcmc`. `burnin+mcmc` must be divisible by 10 and superior or equal to 100 so that the progress bar can be displayed. |
| `thin` | The thinning interval used in the simulation. The number of Gibbs iterations must be divisible by this value. |
| `verbose` | A switch (0,1) which determines whether or not the progress of the sampler is printed to the screen. Default is 1: a progress bar is printed, indicating the step (in %) reached by the Gibbs sampler. |
| `seed` | The seed for the random number generator. If NA, the Mersenne twister generator is used with default seed 12345; if an integer is passed it is used to seed the Mersenne twister. |
| `tune` | Metropolis tuning parameter. Must be a positive scalar. The tuning parameter is updated during the burning period to approach an acceptance rate of 0.44. |
| `beta.start` | The starting values for the $\beta$ vector. This can either be a scalar or a p-length vector. The default value of NA will use the OLS $\beta$ estimate of the corresponding Binary logistic regression. If this is a scalar, that value will serve as the starting value for all of the betas. |
| `mubeta` | The prior mean of $\beta$. This can either be a scalar or a p-length vector. If this takes a scalar value, then that value will serve as the prior mean for all of the betas. The default value of 0 will use a vector of zeros for an uninformative prior. |

`Vbeta` The prior variance of $\beta$. This can either be a scalar or a square p-dimension matrix. If this takes a scalar value, then that value times an identity matrix serves as the prior variance of beta. Default value of 1.0E6 will use a diagonal matrix with very large variance for an uninformative flat prior.

## Details

The `deforestation()` function estimates the parameters of a logistic regression model with variable time-intervals between observations. The estimation is done in a hierarchical Bayesian framework using an adaptive Metropolis algorithm. Function is developped in C++ code using the Scythe statistical library (Pemstein et al. 2007) to maximize efficiency.

The model takes the following form:

$$y_i \sim \mathcal{B}ernoulli(\theta_i')$$

With $\theta_i' = 1 - (1 - \theta_i)^{I_i}$, where $I_i$ stands for the time-interval for observation $i$. Thus, $\theta_i$ is a probability by unit of time (an annual rate for example if the unit of the time-interval is in year).

Using the logit link function denoted $\phi$, we set:

$$\phi(\theta_i) = X_i\beta$$

By default, we assume a multivariate Normal prior on $\beta$:

$$\beta \sim \mathcal{N}(\mu_\beta, V_\beta)$$

For the Metropolis algorithm, the proposal distribution is centered at the current value of $\beta$ and has variance-covariance $V = T(V_\beta^{-1} + C^{-1})^{-1}T$, where $T$ is the diagonal positive definite matrix formed from the `tune`, $V_\beta$ is the prior variance, and $C$ is the large sample variance-covariance matrix of the MLEs without considering the variation of the time-interval between observations. This last calculation is done via an initial call to `glm`.

## Value

`mcmc` An MCMC object that contains the posterior sample. This object can be summarized by functions provided by the coda package.

`deviance` The posterior mean of the deviance $D$, with $D = -2\log(\prod_i P(y_i|\theta_i))$.

`tune` The optimized value for the tuning parameter. This value can be used for potential future runs.

## Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

## References

Daniel Pemstein, Kevin M. Quinn, and Andrew D. Martin. 2007. *Scythe Statistical Library 1.0.* http://scythe.wustl.edu.

Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines. 2002. *Output Analysis and Diagnostics for MCMC (CODA).* http://www-fis.iarc.fr/coda/.

Ghislain Vieilledent, Clovis Grinand and Romuald Vaudry. 2013. Forecasting deforestation and carbon emissions in tropical developing countries facing demographic expansion: a case study in Madagascar. *Ecology and Evolution*. DOI: 10.1002/ece3.550

## See Also

plot.mcmc, summary.mcmc

## Examples

```
## Not run:
#=========================================================================
# Logistic regression with variable time-interval between observations
#=========================================================================

#= Library
library(phcfM)

#== Generating data

# Random seed
set.seed(1234)

# Constants
nobs <- 3000

# Covariates
X1 <- runif(n=nobs,min=-10,max=10)
X2 <- runif(n=nobs,min=-10,max=10)
X <- cbind(rep(1,nobs),X1,X2)
I <- runif(n=nobs,min=1,max=5) # Time-interval

# Target beta parameters
beta.target <- matrix(c(0.3,0.2,0.1),ncol=1)

# Response
theta <- vector()
theta_prim <- vector()
Y <- vector()
for (n in 1:nobs) {
  theta[n] <- inv.logit(X[n,]%*%beta.target)
  theta_prim[n] <- 1-(1-theta[n])^I[n]
  Y[n] <- rbinom(n=1,size=1,prob=theta_prim[n])
}
```

```
# Data-set
Data <- as.data.frame(cbind(Y,I,theta_prim,theta,X1,X2))
plot(Data$X1,Data$theta)
plot(Data$X2,Data$theta)

#== Call to deforestation()
model <- deforestation(formula=Y~X1+X2, interval=Data$I, data=Data, burnin=1000, mcmc=1000,
                       thin=1, verbose=1, seed=NA, tune=1, beta.start=NA, mubeta=0,
                       Vbeta=1.0E6)

#== MCMC analysis

# Graphics
plot(model$mcmc)

# Parameter estimates
str(model)
summary(model$mcmc)
model$deviance
model$tune

## We obtain good parameter estimates
##
##                     Mean       SD  Naive SE Time-series SE
## beta.(Intercept) 0.3362 0.054534 0.0017245       0.006589
## beta.X1          0.2027 0.009345 0.0002955       0.001154
## beta.X2          0.1037 0.007269 0.0002299       0.000838

#== GLM resolution if time-interval is not taken into account

model.glm <- glm(Y~X1+X2,data=Data,family="binomial")
summary(model.glm)

## In this case, the parameter estimates are biased
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 2.000761   0.072069   27.76   <2e-16 ***
## X1          0.247514   0.012060   20.52   <2e-16 ***
## X2          0.123137   0.009803   12.56   <2e-16 ***

## End(Not run)
```

---

demography *Gaussian mixed linear regression for modelling population growth*

---

## Description

The demography() function estimates the parameters of a Gaussian linear mixed model in a hierarchical Bayesian framework. To estimate the posterior distribution of the parameters, block sampling is used by applying the algorithm 2 of Chib and Carlin (1999). The user supplies data and priors and a sample from the posterior distribution is returned as an MCMC object, which can be subsequently analyzed with functions provided in the coda package.

## Usage

```
demography(fixed, random, group, data, burnin=1000, mcmc=10000,
thin=10, verbose=1, seed=NA, beta.start=NA, sigma2.start=NA,
Vb.start=NA, mubeta=0, Vbeta=1.0E6, r, R, nu=0.001, delta=0.001)
```

## Arguments

| | |
|---|---|
| fixed | A two-sided linear formula of the form 'y~x1+...+xp' describing the fixed-effects part of the model, with the response on the left of a '~' operator and the p fixed terms, separated by '+' operators, on the right. |
| random | A one-sided formula of the form '~x1+...+xq' specifying the model for the random effects, with the q random terms, separated by '+' operators. If random=NULL, a fixed effect model is fitted. |
| group | String indicating the name of the grouping variable in data, defining the hierarchical structure of the model. |
| data | A data frame containing the variables in the model. |
| burnin | The number of burnin iterations for the sampler. |
| mcmc | The number of Gibbs iterations for the sampler. Total number of Gibbs iterations is equal to burnin+mcmc. burnin+mcmc must be divisible by 10 and superior or equal to 100 so that the progress bar can be displayed. |
| thin | The thinning interval used in the simulation. The number of Gibbs iterations must be divisible by this value. |
| seed | The seed for the random number generator. If NA, the Mersenne Twister generator is used with default seed 12345; if an integer is passed it is used to seed the Mersenne twister. |
| verbose | A switch (0,1) which determines whether or not the progress of the sampler is printed to the screen. Default is 1: a progress bar is printed, indicating the step (in %) reached by the Gibbs sampler. |
| beta.start | The starting values for the $\beta$ vector. This can either be a scalar or a p-length vector. The default value of NA will use the OLS $\beta$ estimate of the corresponding Gaussian linear regression without random effects. If this is a scalar, that value will serve as the starting value for all of the betas. |
| sigma2.start | Scalar for the starting value of the residual error variance. The default value of NA will use the OLS estimates of the corresponding Gaussian linear regression without random effects. |
| Vb.start | The starting value for variance matrix of the random effects. This must be a square q-dimension matrix. Default value of NA uses an identity matrix. |
| mubeta | The prior mean of $\beta$. This can either be a scalar or a p-length vector. If this takes a scalar value, then that value will serve as the prior mean for all of the betas. The default value of 0 will use a vector of zeros for an uninformative prior. |
| Vbeta | The prior variance of $\beta$. This can either be a scalar or a square p-dimension matrix. If this takes a scalar value, then that value times an identity matrix serves as the prior variance of beta. Default value of 1.0E6 will use a diagonal matrix with very large variance for an uninformative flat prior. |

| | |
|---|---|
| r | The shape parameter for the Inverse-Wishart prior on variance matrix for the random effects. r must be superior or equal to q. Set r=q for an uninformative prior. |
| R | The scale matrix for the Inverse-Wishart prior on variance matrix for the random effects. This must be a square q-dimension matrix. Use plausible variance regarding random effects for the diagonal of R. |
| nu | The shape parameter for the Inverse-Gamma prior on the residual error variance. Default value is nu=delta=0.001 for uninformative prior. |
| delta | The rate (1/scale) parameter for the Inverse-Gamma prior on the residual error variance. Default value is nu=delta=0.001 for uninformative prior. |

## Details

The demography() function samples from the posterior distribution using the blocked Gibbs sampler of Chib and Carlin (1999), Algorithm 2. The simulation is done in compiled C++ code to maximize efficiency. Please consult the coda documentation for a comprehensive list of functions that can be used to analyze the posterior sample.

The model takes the following form:

$$y_i = X_i\beta + W_i b_i + \varepsilon_i$$

Where each group $i$ have $k_i$ observations.

Where the random effects:

$$b_i \sim \mathcal{N}_q(0, V_b)$$

And the errors:

$$\varepsilon_i \sim \mathcal{N}(0, \sigma^2 I_{k_i})$$

We assume standard, conjugate priors:

$$\beta \sim \mathcal{N}_p(\mu_\beta, V_\beta)$$

And:

$$\sigma^2 \sim \mathcal{IG}amma(\nu, 1/\delta)$$

And:

$$V_b \sim \mathcal{IW}ishart(r, rR)$$

See Chib and Carlin (1999) for more details.

*NOTE:* We do not provide default parameters for the priors on the variance matrix for the random effects. When fitting one of these models, it is of utmost importance to choose a prior that reflects your prior beliefs about the random effects. Using the dwish and rwish functions might be useful in choosing these values.

## Value

| | |
|---|---|
| mcmc | An MCMC object that contains the posterior sample. This object can be summarized by functions provided by the coda package. |
| deviance | The posterior mean of the deviance $D$, with $D = -2\log(\prod_i P(y_i\|\beta, b_i, \sigma^2))$, is also provided. |
| Y.pred | Predictive posterior mean for each observation. |

## Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

## References

Siddhartha Chib and Bradley P. Carlin. 1999. "On MCMC Sampling in Hierarchical Longitudinal Models." *Statistics and Computing*. 9: 17-26.

Daniel Pemstein, Kevin M. Quinn, and Andrew D. Martin. 2007. *Scythe Statistical Library 1.0*. http://scythe.wustl.edu.

Andrew D. Martin and Kyle L. Saunders. 2002. "Bayesian Inference for Political Science Panel Data." Paper presented at the 2002 Annual Meeting of the American Political Science Association.

Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines. 2002. *Output Analysis and Diagnostics for MCMC (CODA)*. http://www-fis.iarc.fr/coda/.

Ghislain Vieilledent, Clovis Grinand and Romuald Vaudry. 2013. Forecasting deforestation and carbon emissions in tropical developing countries facing demographic expansion: a case study in Madagascar. *Ecology and Evolution*. DOI: 10.1002/ece3.550

## See Also

plot.mcmc, summary.mcmc

## Examples

```
## Not run:
#=======================================
# Hierarchical Gaussian Linear Regression
#=======================================

library(phcfM)

#== Generating data

# Constants
nobs <- 1000
ntown <- 20
town <- c(1:ntown,sample(c(1:ntown),(nobs-ntown),replace=TRUE))

# Covariates
X1 <- runif(n=nobs,min=0,max=10)
X2 <- runif(n=nobs,min=0,max=10)
X <- cbind(rep(1,nobs),X1,X2)
W <- X

# Target parameters
# beta
beta.target <- matrix(c(0.1,0.3,0.2),ncol=1)
# Vb
Vb.target <- c(0.5,0.2,0.1)
# b
```

```
b.target <- cbind(rnorm(ntown,mean=0,sd=sqrt(Vb.target[1])),
                  rnorm(ntown,mean=0,sd=sqrt(Vb.target[2])),
                  rnorm(ntown,mean=0,sd=sqrt(Vb.target[3])))
# sigma2
sigma2.target <- 0.02

# Response
Y <- vector()
for (n in 1:nobs) {
  Y[n] <- rnorm(n=1,
                mean=X[n,]%*%beta.target+W[n,]%*%b.target[town[n],],
                sd=sqrt(sigma2.target))
}

# Data-set
Data <- as.data.frame(cbind(Y,X1,X2,town))
plot(Data$X1,Data$Y)

#== Call to demography
model <- demography(fixed=Y~X1+X2, random=~X1+X2, group="town",
              data=Data, burnin=1000, mcmc=1000, thin=1,verbose=1,
              seed=NA, beta.start=0, sigma2.start=1,
              Vb.start=1, mubeta=0, Vbeta=1.0E6,
              r=3, R=diag(c(1,0.1,0.1)), nu=0.001, delta=0.001)

#== MCMC analysis

# Graphics
pdf("Posteriors-demography.pdf")
plot(model$mcmc)
dev.off()

# Summary
summary(model$mcmc)

# Predictive posterior mean for each observation
model$Y.pred

# Predicted-Observed
plot(Data$Y,model$Y.pred)
abline(a=0,b=1)


## End(Not run)
```

---

InvGamma *The Inverse Gamma Distribution*

---

## Description

Density function and random generation from the inverse gamma distribution.

## Usage

```
rinvgamma(n, shape, scale = 1)
dinvgamma(x, shape, scale = 1)
```

## Arguments

| | |
|---|---|
| x | Scalar location to evaluate density. |
| n | Number of draws from the distribution. |
| shape | Scalar shape parameter. |
| scale | Scalar scale parameter (default value one). |

## Details

An inverse gamma random variable with shape $a$ and scale $b$ has mean $\frac{b}{a-1}$ (assuming $a > 1$) and variance $\frac{b^2}{(a-1)^2(a-2)}$ (assuming $a > 2$).

## Value

`dinvgamma` evaluates the density at `x`. `rinvgamma` takes `n` draws from the inverse Gamma distribution. The parameterization is consistent with the Gamma Distribution in the stats package.

## Author(s)

Andrew D. Martin <admartin@wustl.edu>, Kevin M. Quinn <kquinn@law.berkeley.edu>, and Jong Hee Park <jhp@uchicago.edu>

## References

Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. 2004. *Bayesian Data Analysis*. 2nd Edition. Boca Raton: Chapman & Hall.

## See Also

GammaDist

## Examples

```
## Not run:
density <- dinvgamma(4.2, 1.1)
draws <- rinvgamma(10, 3.2)

## End(Not run)
```

InvWishart *The Inverse Wishart Distribution*

## Description

Density function and random generation from the Inverse Wishart distribution.

## Usage

```
diwish(W, v, S)
riwish(v, S)
```

## Arguments

| | |
|---|---|
| W | Positive definite matrix W $(p \times p)$. |
| v | Degrees of freedom (scalar). |
| S | Scale matrix $(p \times p)$. |

## Details

The mean of an inverse Wishart random variable with v degrees of freedom and scale matrix S is $(v - p - 1)^{-1}S$.

## Value

diwish evaluates the density at positive definite matrix W. riwish generates one random draw from the distribution.

## Author(s)

Andrew D. Martin <admartin@wustl.edu>, Kevin M. Quinn <kquinn@law.berkeley.edu>, and Jong Hee Park <jhp@uchicago.edu>

## Examples

```
## Not run:
density <- diwish(matrix(c(2,-.3,-.3,4),2,2), 3, matrix(c(1,.3,.3,1),2,2))
draw <- riwish(3, matrix(c(1,.3,.3,1),2,2))

## End(Not run)
```

`logit` *Generalized logit and inverse logit function*

## Description

Compute generalized logit and generalized inverse logit functions.

## Usage

```
logit(x, min = 0, max = 1)
inv.logit(x, min = 0, max = 1)
```

## Arguments

| | |
|---|---|
| `x` | value(s) to be transformed |
| `min` | Lower end of logit interval |
| `max` | Upper end of logit interval |

## Details

The generalized logit function takes values on [min, max] and transforms them to span [-Inf,Inf] it is defined as:

$$y = log(\frac{p}{(1-p)})$$

where

$$p = \frac{(x - min)}{(max - min)}$$

The generized inverse logit function provides the inverse transformation:

$$x = p'(max - min) + min$$

where

$$p' = \frac{exp(y)}{(1 + exp(y))}$$

## Value

Transformed value(s).

## Author(s)

Gregory R. Warnes <greg@warnes.net>

## Examples

```
## Not run:
  x <- seq(0,10, by=0.25)
  xt <- logit(x, min=0, max=10)
  cbind(x,xt)

  y <- inv.logit(xt, min=0, max=10)
  cbind(x,xt,y)

## End(Not run)
```

---

Wishart *The Wishart Distribution*

---

## Description

Density function and random generation from the Wishart distribution.

## Usage

```
dwish(W, v, S)
rwish(v, S)
```

## Arguments

| | |
|---|---|
| W | Positive definite matrix W $(p \times p)$. |
| v | Degrees of freedom (scalar). |
| S | Inverse scale matrix $(p \times p)$. |

## Details

The mean of a Wishart random variable with v degrees of freedom and inverse scale matrix S is $vS$.

## Value

dwish evaluates the density at positive definite matrix W. rwish generates one random draw from the distribution.

## Author(s)

Andrew D. Martin <admartin@wustl.edu>, Kevin M. Quinn <kquinn@law.berkeley.edu>, and Jong Hee Park <jhp@uchicago.edu>

## Examples

```
## Not run:
density <- dwish(matrix(c(2,-.3,-.3,4),2,2), 3, matrix(c(1,.3,.3,1),2,2))
draw <- rwish(3, matrix(c(1,.3,.3,1),2,2))

## End(Not run)
```

# Index