

# Package ‘fastLaplace’

October 13, 2022

**Type** Package

**Title** A Fast Laplace Method for Spatial Generalized Linear Mixed Model

**Version** 0.0.2

**Maintainer** Sangwan Lee <sangwanlee@yonsei.ac.kr>

**Description** Fitting a fast Laplace approximation for Spatial Generalized Linear Mixed Model as described in Park and Lee (2021) <<https://github.com/sangwan93/fastLaplace/blob/main/FastLaplaceMain.pdf>>.

**License** GPL-3

**Encoding** UTF-8

**LazyData** false

**RoxygenNote** 7.1.1

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0), spelling, mgcv, ngsatial, MASS

**VignetteBuilder** knitr

**Language** en-US

**Imports** RSpectra, bbmle, fields

**URL** <<https://www.naver.com/>>

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Sangwan Lee [cre, aut],  
Jaewoo Park [aut]

**Repository** CRAN

**Date/Publication** 2021-06-28 08:20:02 UTC

## R topics documented:

fsglmm	2
fsglmm.discrete	4
pred.sglmm	6

<b>Index</b>	<b>7</b>
--------------	----------

---

fsglmm	<i>Fitting Projection Based Laplace Approximation for Spatial Generalized Linear Mixed Model</i>
--------	--

---

### Description

fsglmm is used to fit reduced-dimensional spatial generalized linear mixed models for continuous spatial domain.

### Usage

```
fsglmm(
  formula,
  kappa,
  inits,
  data,
  coords,
  family,
  ntrial = 1,
  offset = NA,
  method.optim,
  method.integrate,
  rank = NULL,
  control = list()
)
```

### Arguments

formula	an object of class "formula."
kappa	the smoothness parameter for the matern process (either 0.5, 1.5, 2.5 or 10).
inits	starting values for the parameters.
data	a data frame containing variables in the model.
coords	a matrix of dimension $N \times 2$ representing the longitude and latitude of each observation.
family	a character string of the error distribution and link function to be used in the model.
ntrial	a numeric vector for a binomial model.
offset	this is used to specify an a priori a known component to be included in the linear predictor during fitting.
method.optim	the method to be used for outer optimization. "CG" for Conjugate Gradient Method.
method.integrate	the method to be used for inner optimization. "NR" for Newton Raphson Method.
rank	an integer of 'rank' to be used for projections. Default is 5 percent of observations.
control	a list of control parameters.

**Value**

a list containing the following components:

summary a summary of the fitted model

vcov a symmetric matrix giving an estimate of the Hessian at the solution found.

mle2 an object of class "mle2"

family the family used.

kappa the matern smoothness parameter used.

Delta a matrix containing the estimated random effects of the reduced dimensional model.

U a matrix whose columns contain the estimated eigenvectors of the reduced dimensional model.

D a matrix whose diagonal components contain the estimated eigenvalues of the reduced dimensional model.

coords the matrix of coordinates used.

**References**

Jaewoo Park and Sangwan Lee - "A Projection-based Laplace Approximation for Spatial Latent Variable Models"

**Examples**

```

if(requireNamespace("mgcv")){
  sigma2 = 1
  phi = 0.2
  beta.true = c(1,1)
  n = 400
  n.pred = 100
  coords.all <- matrix(runif((n+n.pred)*2),ncol=2,nrow=n+n.pred)
  X.all <- matrix(runif((n+n.pred)*2),ncol=2,nrow=(n+n.pred))
  dist.all <- fields::rdist(coords.all,coords.all)
  V.all <- sigma2*(1+sqrt(5)/phi*dist.all+5/(3*phi^2)*dist.all^2)*exp(-sqrt(5)/phi*dist.all)
  set.seed(1)
  r.e.all <- mgcv::rmvn(1,rep(0,nrow(coords.all)),V.all)
  pi.all <- X.all%*%beta.true + r.e.all
  p.all <- exp(pi.all)/(1+exp(pi.all))
  Y.all <- sapply(p.all, function(x) sample(0:1, 1, prob = c(1-x, x)))
  Y <- as.matrix(Y.all[1:n],nrow = n)
  X <- X.all[1:n,]
  coords <- coords.all[1:n,]
  data <- data.frame(cbind(Y,X))
  colnames(data) = c("Y","X1","X2")
  mod.glm <- glm(Y~1+X1+X2,family="binomial",data=data)
  mod.glm.esp <- predict(mod.glm,data, type="response")
  mod.glm.s2 <- var(Y - mod.glm.esp)
  mod.glm.phi <- 0.1*max(dist(coords))
  startinit <- c(mod.glm$coef,log(mod.glm.s2),log(mod.glm.phi))
  names(startinit) <- c("X1","X2","logsigma2","logphi")
  result.bin <- fsglmm(Y~1+X1+X2, kappa=2.5, inits = startinit,

```

```

data = data,coords = coords, family = "binomial", ntrial = 1,
offset = NA,method.optim = "CG", method.integrate = "NR",rank = 50)
}

```

---

fsglmm.discrete	<i>Fitting Projection Based Laplace Approximation for Spatial Generalized Linear Mixed Model</i>
-----------------	--

---

### Description

fsglmm.discrete

### Usage

```

fsglmm.discrete(
  formula,
  inits,
  data,
  family,
  ntrial = 1,
  method.optim,
  method.integrate,
  rank = NULL,
  A,
  offset = NA
)

```

### Arguments

formula	an object of class "formula."
inits	starting values for the parameters.
data	a data frame containing variables in the model.
family	a character string of the error distribution and link function to be used in the model.
ntrial	a numeric vector for binomial model.
method.optim	the method to be used for outer optimization. "CG" for Conjugate Gradient Method.
method.integrate	the method to be used for inner optimization. "NR" for Newton Raphson Method.
rank	an integer of 'rank' to be used for projections. Default is 5 percent of observations.
A	an adjacency matrix
offset	this is used to specify an a priori a known component to be included in the linear predictor during fitting.

**Value**

a list containing the following components:

summary a summary of the fitted model

mle2 an object of class "mle2"

Delta a matrix containing the estimated random effects of the reduced dimensional model.

M the projection matrix used.

**Examples**

```

if(requireNamespace("ngspatial")&
requireNamespace("mgcv")){
n = 30
A = ngspatial::adjacency.matrix(n)
Q = diag(rowSums(A),n^2) - A
x = rep(0:(n - 1) / (n - 1), times = n)
y = rep(0:(n - 1) / (n - 1), each = n)
X = cbind(x, y)
beta = c(1, 1)
P.perp = diag(1,n^2) - X%%solve(t(X)%%X)%%t(X)
eig = eigen(P.perp %% A %% P.perp)
eigenvalues = eig$values
q = 400
M = eig$vectors[,c(1:q)]
Q.s = t(M) %% Q %% M
tau = 6
Sigma = solve(tau*Q.s)
set.seed(1)
delta.s = mgcv::rmvn(1, rep(0,q), Sigma)
lambda = exp( X%%beta + M%%delta.s )
Z = c()
for(j in 1:n^2){Z[j] = rpois(1,lambda[j])}
Y = as.matrix(Z,ncol=1)
data = data.frame("Y"=Y, "X"=X)
colnames(data) = c("Y", "X1", "X2")
linmod <- glm(Y~-1+X1+X2,data=data,family="poisson") # Find starting values
linmod$coefficients
starting <- c(linmod$coefficients,"logtau"=log(1/var(linmod$residuals)) )

result.pois.disc <- fsglmm.discrete(Y~-1+X1+X2, inits = starting, data=data,
family="poisson",ntrial=1, method.optim="BFGS", method.integrate="NR",rank=50, A=A)
}

```

---

`pred.sglm`*Model Predictions*

---

**Description**

`pred.sglm` is a function for predictions from the results of `fsglm`.

**Usage**

```
pred.sglm(fit.sglm, data, coords, ntrial = 1, offset = NA)
```

**Arguments**

<code>fit.sglm</code>	a list from <code>fsglm</code>
<code>data</code>	a data frame to be predicted by the model.
<code>coords</code>	coordinates for prediction
<code>ntrial</code>	a numeric vector of the total number of trials ( binomial )
<code>offset</code>	a numeric vector indicating a known component to be included in the linear predictor for predictions.

**Value**

a vector of predicted mean parameters. (e.g. probabilities for binomial case)

**Examples**

```
## the result from fsglm, data to be predicted, and the coordinates for prediction is required.
```

```
result <- fsglm(Y~1+X1+X2, kappa=2.5, inits = startinit, data = data, coords = coords,  
family = "binomial", ntrial = 1, offset = NA, method.optim = "CG", method.integrate = "NR", rank = 50)  
pred.sglm(fit.sglm=result, data=X.pred,)
```

# Index

fsglmm, 2  
fsglmm.discrete, 4  
pred.sglm, 6