

# Package ‘BFI’

September 29, 2023

**Type** Package

**Title** Bayesian Federated Inference

**Version** 0.6.4

**Date** 2023-07-16

**Author** Hassan Pazira [aut, cre],  
Marianne A. Jonker [aut],  
Anthony C.C. Coolen [aut]

**Maintainer** Hassan Pazira <hassan.pazira@radboudumc.nl>

**Description** Bayesian Federated Inference (BFI) method combines inference results obtained from local data sets in the separate centers. In this version of the package, the BFI methodology is programmed for linear and logistic regression models.

**License** MIT + file LICENSE

**URL** <https://hassanpazira.github.io/BFI/>

**Encoding** UTF-8

**Suggests** knitr,  
rmarkdown,  
testthat (>= 3.0.0)

**VignetteBuilder** knitr

**Depends** R (>= 2.10)

**LazyData** true

**Imports** devtools

**Config/testthat/edition** 3

**RoxygenNote** 7.2.3

## R topics documented:

BFI-package . . . . .	2
bfi . . . . .	2
inv.prior.cov . . . . .	6
MAP.estimation . . . . .	9
Nurses . . . . .	12
summary.bfi . . . . .	13
trauma . . . . .	15
<b>Index</b>	<b>16</b>

BFI-package

*Bayesian Federated Inference***Description**

The Bayesian Federated Inference method combines inference results from different (medical) centers without sharing the data. In this version of the package, the user can fit models specifying Gaussian and Binomial (Logistic) families. The package will be updated with more models soon.

**Details**

Package: BFI  
 Type: Package  
 Version: 0.6.4  
 Date: 2023-07-16  
 License: GPL (>=2)

MAP.estimation and bfi are the main functions. All other functions are utility functions.

Some examples are provided in the vignettes accompanying this package in order to show how the package can be applied to real data. The vignettes can be found on the package website at <https://hassanpazira.github.io/BFI/> or from within R once the package has been installed, e.g. via `vignette("BFI", package = "BFI")`.

**Author(s)**

Hassan Pazira, Marianne A. Jonker, Anthony C.C. Coolen  
 Maintainer: Hassan Pazira <hassan.pazira@radboudumc.nl>

**References**

Jonker M.A., Pazira H. and Coolen A.C.C. (2023). *Bayesian Federated Inference for Statistical Models, Statistics in Medicine*, Vol. 0(0), 0-0. <<https://doi.org/10.48550/arXiv.2302.07677>>

bfi

*Bayesian Federated Inference***Description**

bfi function can be used (in the central server) to combine inference results from separate data sets (without combining the data) to approximate what would have been inferred had the data sets been merged. For now the function can handle linear and logistic regression models, but code for more models will be available in the near future. bfi command

**Usage**

```
bfi(theta_hats = NULL, A_hats, Lambda, stratified = FALSE,
    strat_par = NULL, center_spec = NULL, type=c("cat", "con"))
```

## Arguments

theta_hats	a list of $L$ vectors of the maximum a posteriori (MAP) estimates of the model parameters in the $L$ centers. These vectors must have equal dimensions. See ‘Details’.
A_hats	a list of $L$ (minus) curvature matrices for $L$ centers. These matrices must have equal dimensions. See ‘Details’.
Lambda	a list of $L + 1$ matrices. The $k$ matrix is the chosen inverse variance-covariance matrix of the Gaussian distribution that is used as prior distribution in center $k$ , where $k = 1, 2, \dots, L$ . The last matrix is the chosen variance-covariance matrix for the Gaussian prior of the (fictive) combined data set. If stratified = FALSE, all $L + 1$ matrices must have equal dimensions. While, if stratified = TRUE, the first $L$ matrices must have equal dimensions and the last matrix should have a different (greater) dimension than the others. See ‘Details’.
stratified	logical flag for performing the stratified analysis. If stratified = TRUE, the parameter(s) selected in the strat_par argument are allowed to be different across centers, except when the argument center_spec is not NULL. Default is stratified = FALSE. See ‘Details’ and ‘Examples’.
strat_par	a one- or two-element integer vector for indicating the stratification parameter(s). The values 1 and/or 2 are/is used to indicate that the “intercept” and/or “sigma2” are allowed to vary, respectively. This argument is used only when stratified = TRUE and center_spec = NULL. Default is strat_par = NULL, but if stratified = TRUE, strat_par can not be NULL unless there is a center specific variable. For the binomial family the length of the vector should be at most one which refers to “intercept”, and the value of this element should be 1. For gaussian family this vector can be 1 for indicating the “intercept” only, 2 for indicating the “sigma2” only, and c(1, 2) for both “intercept” and “sigma2”. See ‘Details’ and ‘Examples’.
center_spec	a vector of $L$ elements for representing the center specific variable. This argument is used only when stratified = TRUE and strat_par = NULL. Each element represents a specific feature of the corresponding center. There must be only one specific value or attribute for each center. This vector could be a numeric, characteristic or factor vector. Note that, the order of the centers in the vector center_spec must be the same as in the list of the argument theta_hats. If center_spec = NULL (the default) and stratified = TRUE, the stratification parameter(s) given in the argument strat_par are assumed to be different across the centers. See ‘Details’ and ‘Examples’.
type	a description of the used data type in the center_spec argument. Type for ‘categorical’ (type = “cat”) or ‘continuous’ (type = “con”) data. Can be also abbreviated. This argument is used only when center_spec is not NULL.

## Details

bfi function implements the BFI approach described in the paper Jonker et. al. (2023) given in the references. The inference results gathered from different ( $L$ ) centers are combined, and the BFI estimates of the model parameters and curvature matrix evaluated at that point are returned.

The inference result from each center must be obtained using the MAP.estimation function separately, and then all of these results (coming from different centers) should be compiled into a list to be used as an input of bfi(). The models in the different centers should be defined in exactly the same way; among others, exactly the same covariates should be included in the models. The parameter vectors should be defined exactly the same, so that the  $L$  vectors and matrices in the input

lists `theta_hat`'s and `A_hat`'s are defined in the same way (e.g., the covariates need to be included in the models in the same order).

Note that the order of the elements in the lists `theta_hats`, `A_hats` and `Lambda`, must be the same with respect to the centers, so that in every list the element at the  $\ell$  position is from the center  $\ell$ . This should also be the case for the vector `center_spec`.

If `intercept = FALSE` for the locations, the stratified analysis is not possible anymore for the binomial family.

If `stratified = FALSE`, both `strat_par` and `center_spec` must be `NULL` (the defaults), while if `stratified = TRUE` only one of the two must be `NULL`.

If `stratified = FALSE` and all the  $L + 1$  matrices are equal, it is sufficient to give a (list of) one matrix only. In both cases of the `stratified` argument (`TRUE` or `FALSE`), if only the first  $L$  matrices are equal, the argument `Lambda` can be a list of two matrices, so that the first matrix represents the chosen variance-covariance matrix for local centers and the second one is the chosen matrix for the combined data set. The last matrix of the list in the argument `Lambda` can be built by the function `inv.prior.cov()`.

## Value

`bfi` returns a list containing the following components:

<code>theta_hat</code>	the vector of estimates obtained by combining the inference results from the $L$ centers with the 'BFI' methodology. If an intercept was fitted in every center and <code>stratified = FALSE</code> , there is only one general "intercept" in this vector, while if <code>stratified = TRUE</code> and <code>strat_par = 1</code> , there are $L$ different intercepts in the model, for each center one;
<code>A_hat</code>	minus the curvature (or Hessian) matrix obtained by the 'BFI' method for the combined model. If <code>stratified = TRUE</code> , the dimension of the matrix is always greater than when <code>stratified = FALSE</code> . When <code>stratified = TRUE</code> , some elements at the off-diagonal matrix can not be determined and are therefore <code>NA</code> ;
<code>sd</code>	the vector of standard deviation of the estimates in <code>theta_hat</code> obtained from the matrix in <code>A_hat</code> , i.e., the vector equals <code>sqrt(diag(solve(A_hat)))</code> which equals the square root of the elements at the diagonal of the inverse of the <code>A_hat</code> matrix.

## Author(s)

Hassan Pazira  
Maintainer: Hassan Pazira <hassan.pazira@radboudumc.nl>

## References

Jonker M.A., Pazira H. and Coolen A.C.C. (2023). *Bayesian Federated Inference for Statistical Models*. Statistics in Medicine, Vol. 0(0), 0-0. <<https://doi.org/10.48550/arXiv.2302.07677>>

## See Also

[MAP.estimation](#) and [inv.prior.cov](#)

## Examples

```
#-----
# y ~ Binomial
#-----
# model assumption:
beta <- 1:4 # regression coefficients (beta[1] is the intercept)

#-----
# Data Simulation for local center 1
#-----
n1 <- 30 # sample size of center 1
X1 <- data.frame(x1=rnorm(n1), # continuous variable
                 x2=sample(0:2, n1, replace=TRUE)) # categorical variable
# make dummy variables
X1x2_1 <- ifelse(X1$x2 == '1', 1, 0)
X1x2_2 <- ifelse(X1$x2 == '2', 1, 0)
X1$x2 <- as.factor(X1$x2)
# linear predictor:
eta1 <- beta[1] + X1$x1 * beta[2] + X1x2_1 * beta[3] + X1x2_2 * beta[4]
# inverse of the link function (  $g^{-1}(\eta) = \mu$  ):
mu1 <- binomial()$linkinv(eta1)
y1 <- rbinom(n1, 1, mu1)

#-----
# Data Simulation for local center 2
#-----
n2 <- 50 # sample size of center 2
X2 <- data.frame(x1=rnorm(n2), # continuous variable
                 x2=sample(0:2, n2, replace=TRUE)) # categorical variable
# make dummy variables:
X2x2_1 <- ifelse(X2$x2 == '1', 1, 0)
X2x2_2 <- ifelse(X2$x2 == '2', 1, 0)
X2$x2 <- as.factor(X2$x2)
# linear predictor:
eta2 <- beta[1] + X2$x1 * beta[2] + X2x2_1 * beta[3] + X2x2_2 * beta[4]
# inverse of the link function:
mu2 <- binomial()$linkinv(eta2)
y2 <- rbinom(n2, 1, mu2)

#-----
# MAP estimates at center 1
#-----
# assume the same inverse covariance matrix (Lambda) for both centers:
Lambda <- inv.prior.cov(X1, lambda=0.01, family=binomial)
fit1 <- MAP.estimation(y1, X1, family=binomial, Lambda)
theta_hat1 <- fit1$theta_hat # intercept and coefficient estimates
A_hat1 <- fit1$A_hat # minus curvature matrix

#-----
# MAP estimates at center 2
#-----
fit2 <- MAP.estimation(y2, X2, family=binomial, Lambda)
theta_hat2 <- fit2$theta_hat
A_hat2 <- fit2$A_hat

#-----
```

```
# BFI at central center
#-----
A_hats    <- list(A_hat1, A_hat2)
theta_hats <- list(theta_hat1, theta_hat2)
bfi       <- bfi(theta_hats, A_hats, Lambda)
class(bfi)
summary(bfi, cur_mat=TRUE)

#-----
# stratified analysis
#-----
# By running the following line an error appears because when
# stratified = TRUE, both 'strat_par' and 'center_spec' can not be NULL
# bfi(theta_hats, A_hats, Lambda, stratified=TRUE)
#bfi(theta_hats, A_hats, Lambda, stratified=TRUE, strat_par=1)
```

---

inv.prior.cov

*Creates an inverse covariance matrix for a Gaussian prior*


---

## Description

inv.prior.cov builds a diagonal inverse covariance matrix for the Gaussian prior distribution based on the design matrix of covariates, that takes into account the number of regression parameters in case of categorical covariates. In case of a linear model, it also includes a row and column for the variance of the measurement errors.

## Usage

```
inv.prior.cov(X, lambda = 1, family = gaussian, intercept = TRUE,
              stratified = FALSE, strat_par = NULL, L = 2)
```

## Arguments

X	design matrix of dimension $n \times p$ , where $n$ is the number of samples observed, and $p$ is the number of predictors/variables so excluding the intercept.
lambda	the vector used as the diagonal of the (inverse covariance) matrix that will be created by inv.prior.cov(). The length of the vector depends on the number of columns of X, type of the covariates (continuous/dichotomous or categorical), family, whether an intercept is included in the model, and whether stratified analysis is desired. When stratified = FALSE, lambda could be a single positive number (if all values in the vector are equal), a vector of two elements (the first is used for regression parameters including “intercept” and the second for the “sigma2”), or a vector of length equal to the number of model parameters. However, the length of lambda is different when stratified = TRUE, see ‘Details’ for more information. Default is lambda = 1.
family	a description of the error distribution and link function used to specify the model. This can be a character string naming a family function or the result of a call to a family function (see <a href="#">family</a> for details). In the current version, the family of model can be gaussian (with identity link function) and binomial (with logit link function). By default the gaussian family is used. In case of a linear regression model, family = gaussian, there is an extra model parameter for the variance of measurement error.

intercept	logical flag for having an intercept. By changing the intercept the dimension of the inverse covariance matrix changes. If <code>intercept = TRUE</code> (the default), the output matrix created by <code>inv.prior.cov()</code> has one row and one column related to intercept, while if <code>intercept = FALSE</code> , the resulting matrix does not have the row and column called intercept.
stratified	logical flag for performing the stratified analysis. If <code>stratified = TRUE</code> , the parameter(s) selected in the <code>strat_par</code> argument are allowed to be different across centers. This argument should only be used when designing the inverse covariance matrix for the (fictive) combined data, i.e., the last matrix for the <code>Lambda</code> argument in <code>bfi()</code> . If <code>inv.prior.cov()</code> is used for the analysis in the local centers (to built the $L$ first matrices for the <code>Lambda</code> argument in <code>bfi()</code> ), this argument should be <code>FALSE</code> , even if the BFI analysis is stratified. Default is <code>stratified = FALSE</code> . See ‘Details’ and ‘Examples’.
strat_par	a one- or two-element integer vector for indicating the stratification parameter(s). The values 1 and/or 2 are/is used to indicate that the “intercept” and/or “sigma2” are allowed to vary, respectively. This argument is used only when <code>stratified = TRUE</code> . Default is <code>strat_par = NULL</code> , but if <code>stratified = TRUE</code> , <code>strat_par</code> can not be <code>NULL</code> . For the binomial family the length of the vector should be one which refers to “intercept”, and the value of this element should be 1. For gaussian this vector can be 1 for indicating the “intercept” only, 2 for indicating the “sigma2” only, and <code>c(1, 2)</code> for both “intercept” and “sigma2”. See ‘Examples’.
L	the number of centers. This argument is used only when <code>stratified = TRUE</code> . Default is <code>L = 2</code> . See ‘Details’ and ‘Examples’.

## Details

`inv.prior.cov` creates a diagonal matrix with the vector `lambda` as its diagonal. The argument `stratified = TRUE` should only be used to construct a matrix for the prior density in case of stratification in the fictive combined data. Never be used for the construction of the matrix for analysis in the centers.

When `stratified = FALSE`, the length of the vector `lambda` depends on the covariate matrix  $X$ , family, and whether an “intercept” is included in the model. For example, if the design matrix  $X$  has  $p$  columns with continuous or dichotomous covariates, `family = gaussian`, and `intercept = TRUE`, then `lambda` should have  $p + 2$  elements. In this case, if in  $X$  there is a categorical covariate with  $q > 2$  categories, then the length of `lambda` increases with  $q - 2$ . All values of `lambda` should be non-negative as they represent the inverse of the variance of the Gaussian prior. Note that, if all values in the vector `lambda` equal, one value is enough to be given as entry. If `lambda` is a scalar, the function `inv.prior.cov` sets each value at the diagonal equal to `lambda`. In the linear regression model the last parameter is assumed to be the inverse of the variance of the prior distribution for the measurement error. If `lambda` is two dimensional, the first value is used for the prior of the regression parameters and the second for the inverse of the variance of the prior distribution for the measurement error.

If `stratified = TRUE` the length of the vector `lambda` should be equal to the number of parameters in the combined model.

If `intercept = FALSE`, for the binomial family the stratified analysis is not possible therefore `stratified` can not be `TRUE`.

The output of `inv.prior.cov()` can be used in the main functions `MAP.estimation()` and `bfi()`.

**Value**

inv.prior.cov returns a diagonal matrix. The dimension of the matrix depends on the number of columns of X, type of the covariates (continuous/dichotomous or categorical), family, and intercept.

**Author(s)**

Hassan Pazira

Maintainer: Hassan Pazira <hassan.pazira@radboudumc.nl>

**References**

Jonker M.A., Pazira H. and Coolen A.C.C. (2023). *Bayesian Federated Inference for Statistical Models*, Statistics in Medicine, Vol. 0(0), 0-0. <<https://doi.org/10.48550/arXiv.2302.07677>>

**See Also**

[MAP.estimation](#)

**Examples**

```
#-----
# Data Simulation
#-----
X <- data.frame(x1=rnorm(50),                # standard normal variable
               x2=sample(0:2, 50, replace=TRUE), # categorical variable
               x3=sample(0:1, 50, replace=TRUE)) # dichotomous variable
X$x2 <- as.factor(X$x2)
X$x3 <- as.factor(X$x3)

# The (inverse) variance value (lambda=0.05) is assumed to be
# the same for Gaussian prior of all parameters (for non-stratified)

#-----
# Inverse Covariance Matrix for the Gaussian prior
#-----
# y ~ Binomial with intercept
inv.prior.cov(X, lambda=0.05, family=binomial) # returns a 5-by-5 matrix

# y ~ Binomial without intercept
inv.prior.cov(X, lambda=0.05, family="binomial", intercept = FALSE) # a 4-by-4 matrix

# y ~ Gaussian with intercept
inv.prior.cov(X, lambda=0.05, family=gaussian) # returns a 6-by-6 matrix

# Stratified analysis when 'intercept' varies across 3 centers; y ~ Binomial
inv.prior.cov(X, lambda=c(.2, 1), family=binomial, stratified=TRUE, strat_par = 1, L = 3)

# Stratified analysis when 'intercept' and sigma2 vary across 2 centers; y ~ Gaussian
inv.prior.cov(X, lambda=c(1, 2, 3), family=gaussian, stratified=TRUE, strat_par = c(1, 2))

# Stratified analysis when 'sigma2' varies across 2 centers; y ~ Gaussian with 'intercept'
inv.prior.cov(X, lambda=c(1, 2, 3), family=gaussian, stratified=TRUE, strat_par = 2)

# Stratified analysis when 'sigma2' vary across 2 centers; y ~ Gaussian without 'intercept'
inv.prior.cov(X, lambda=c(2, 3), family=gaussian, intercept = F, stratified=TRUE, strat_par = 2)
```



MAP.estimation

*Maximum A Posteriori estimation***Description**

MAP.estimation function is used (in local centers) to compute Maximum A Posterior (MAP) estimators of the parameters for the GLM and soon for Survival models.

**Usage**

```
MAP.estimation(y, X, family = gaussian, Lambda, intercept = TRUE,
              initial = NULL, control = list())
```

**Arguments**

- |           |   |
|-----------|---|
| y         | response vector. If the binomial family is used, this argument is a vector with entries 0 (failure) or 1 (success). Alternatively, the response can be a matrix where the first column is the number of “successes” and the second column is the number of “failures”.  |
| X         | design matrix of dimension $n \times p$ , where $p$ is the number of covariables or predictors.   |
| family    | a description of the error distribution and link function used to specify the model. This can be a character string naming a family function or the result of a call to a family function (see <a href="#">family</a> for details). In the current version of the package, the family of model can be gaussian (with identity link function) and binomial (with logit link function). By default the gaussian family is used. In case of a linear regression model, family = gaussian, there is an extra model parameter for the variance of measurement error. |
| Lambda    | the inverse variance-covariance matrix of the Gaussian distribution that is used as prior distribution for the model parameters. The dimension of the matrix depends on the number of columns of X, type of the covariates (continuous / dichotomous or categorical), family, and intercept. However, Lambda can be easily created by <code>inv.prior.cov()</code> .  |
| intercept | logical flag for fitting an intercept. If intercept=TRUE (the default), the intercept is fitted, i.e., it is included in the model, and if intercept=FALSE it is set to zero, i.e., it's not in the model.  |
| initial   | a vector specifying initial values for the parameters to be optimized over. The length of initial is equal to the number of model parameters and thus, is equal to the number of rows or columns of Lambda. Since the 'L-BFGS-B' method is used in the algorithm, these values should always be finite. Default is a vector of zeros.   |
| control   | a list of control parameters. See ‘Details’.  |

## Details

MAP.estimation function finds the Maximum A Posteriori (MAP) estimates of the model parameters by maximizing the log-posterior density with respect to the parameters, i.e., the estimates equal the values for which the log-posterior density is maximal (the posterior mode). In other words, MAP.estimation() optimizes the log-posterior density with respect to the parameter vector to obtain its MAP estimates. In addition to the model parameters, i.e., coefficients ( $\beta$ 's) and variance error ( $\sigma_e^2$ ), the curvature matrix (Hessian of the log-posterior) is estimated around the mode.

The MAP.estimation function returns an object of class 'bfi'. Therefore, summary() can be used for the object returned by MAP.estimation().

To solve unconstrained and bound-constrained optimization problems, the MAP.estimation function utilizes an optimization algorithm called Limited-memory Broyden-Fletcher-Goldfarb-Shanno with Bound Constraints (L-BFGS-B), Byrd et. al. (1995). The L-BFGS-B algorithm is a limited-memory "quasi-Newton" method that iteratively updates the parameter estimates by approximating the inverse Hessian matrix using gradient information from the history of previous iterations. This approach allows the algorithm to approximate the curvature of the posterior distribution and efficiently search for the optimal solution, which makes it computationally efficient for problems with a large number of variables.

By default, the algorithm uses a relative change in the objective function as the convergence criterion. When the change in the objective function between iterations falls below a certain threshold ('factr') the algorithm is considered to have converged. The convergence can be checked with the argument convergence in the output. See 'Value'.

In case of convergence issue, it may be necessary to investigate and adjust optimization parameters to facilitate convergence. It can be done using the initial and control arguments. By the argument initial the initial points of the iterative optimization algorithm can be changed, and the argument control is a list that can supply any of the following components:

**maxit:** is the maximum number of iterations. Default is 100;

**factr:** controls the convergence of the 'L-BFGS-B' method. Convergence occurs when the reduction in the objective is within this factor of the machine tolerance. Default for factr is 1e7, which gives a tolerance of about 1e-9. The exact tolerance can be checked by multiplying this value by .Machine\$double.eps;

**pgtol:** helps to control the convergence of the 'L-BFGS-B' method. It is a tolerance on the projected gradient in the current search direction, i.e., the iteration will stop when the maximum component of the projected gradient is less than or equal to pgtol, where  $\text{pgtol} \geq 0$ . Default is zero, when the check is suppressed;

**trace:** is a non-negative integer. If positive, tracing information on the progress of the optimization is produced. Higher values may produce more tracing information: for the method 'L-BFGS-B' there are six levels of tracing. To understand exactly what these do see the source code of optim function in the [stats](#) package;

**REPORT:** is the frequency of reports for the 'L-BFGS-B' method if 'control\$trace' is positive. Default is every 10 iterations;

**lmm:** is an integer giving the number of BFGS updates retained in the 'L-BFGS-B' method. Default is 5.

## Value

MAP.estimation returns a list containing the following components:

theta_hat	the vector corresponding to the maximum a posteriori (MAP) estimates of the parameters;
-----------	---

A_hat	minus the curvature (or Hessian) matrix around the point theta_hat. The dimension of the matrix is the same as the argument Lambda;
sd	the vector of standard deviation of the MAP estimates in theta_hat, that is <code>sqrt(diag(solve(A_hat)))</code> ;
Lambda	the inverse variance-covariance matrix of the Gaussian distribution that is used as prior distribution for the parameters. It's exactly the same as the argument Lambda;
formula	the formula applied;
names	the names of the model parameters;
n	sample size;
np	the number of coefficients;
value	the value of minus the log-likelihood posterior density evaluated at theta_hat;
family	the <code>family</code> object used.;
intercept	logical flag used to fit an intercept if TRUE, or set to zero if FALSE;
convergence	an integer value used to encode the warnings and the errors related to the algorithm used to fit the model. The values returned are: <b>0</b> algorithm has converged; <b>1</b> maximum number of iterations ('maxit') has been reached; <b>2</b> Warning from the 'L-BFGS-B' method. See the message after this value;
control	the list of control parameters used to compute the MAP estimates.

**Author(s)**

Hassan Pazira

Maintainer: Hassan Pazira &lt;hassan.pazira@radboudumc.nl&gt;

**References**

Jonker M.A., Pazira H. and Coolen A.C.C. (2023). *Bayesian Federated Inference for Statistical Models*. Statistics in Medicine, Vol. 0(0), 0-0. <<https://doi.org/10.48550/arXiv.2302.07677>>

Byrd R.H., Lu P., Nocedal J. and Zhu C. (1995). *A limited memory algorithm for bound constrained optimization*. SIAM Journal on Scientific Computing, 16, 1190-1208. <<https://doi.org/10.1137/0916069>>

**See Also**

`bfi`, `inv.prior.cov` and `summary.bfi`

**Examples**

```
#-----
# y ~ Gaussian
#-----
# model assumption:
theta <- c(1, 2, 2, 2, 1.5) # model parameters: coefficients and sigma2 = 1.5

#-----
# Data Simulation
#-----
n <- 30 # sample size
p <- 3 # number of coefficients without intercept
```

```

X <- data.frame(matrix(rnorm(n * p), n, p)) # continuous variables
# linear predictor:
eta <- theta[1] + theta[2] * X$X1 + theta[3] * X$X2 + theta[4] * X$X3
# inverse of the link function (  $g^{-1}(\eta) = \mu$  ):
mu <- gaussian()$linkinv(eta)
y <- rnorm(n, mu, sd = sqrt(theta[5]))

#-----
# MAP estimations for theta and curvature matrix
#-----
# MAP estimates with 'intercept'
Lambda <- inv.prior.cov(X, lambda = c(0.1, 1), family = gaussian)
(fit <- MAP.estimate(y, X, family = gaussian, Lambda))
class(fit)
summary(fit, cur_mat = TRUE)

# MAP estimates without 'intercept'
Lambda <- inv.prior.cov(X, lambda = c(0.1, 1), family = gaussian, intercept = FALSE)
(fit1 <- MAP.estimate(y, X, family = gaussian, Lambda, intercept = FALSE))
summary(fit1, cur_mat = TRUE)

```

---

Nurses

---

*Nurses' stress in different hospitals*


---

## Description

This dataset comprises three-level simulated data extracted for a hypothetical study investigating stress levels within hospital settings. The dataset focuses on nurses working in specific wards within various hospitals. It includes several variables, such as nurse age (measured in years), nurse experience (measured in years), nurse gender (0 for male, 1 for female), ward type (0 for general care, 1 for special care), and hospital size (0 for small, 1 for medium, 2 for large). The dataset in the package is obtained from the original dataset by leaving out some of the unused columns.

## Usage

```
data(Nurses)
```

## Source

<https://multilevel-analysis.sites.uu.nl/datasets/>

## References

Hox, J., Moerbeek, M., and van de Schoot, R. (2010). *Multilevel Analysis: Techniques and Applications*, Second Edition (2nd ed.). Routledge. <<https://doi.org/10.4324/9780203852279>>

summary.bfi

*Summarizing BFI Fits***Description**

Summary method for an object with class 'bfi' created by the `MAP.estimate` function.

**Usage**

```
## S3 method for class 'bfi'
summary(object, cur_mat = FALSE,
        digits = max(3, getOption("digits") - 3), ...)
```

**Arguments**

<code>object</code>	fitted bfi object.
<code>cur_mat</code>	logical; if TRUE, minus the curvature matrix around the estimated parameters is returned and printed. Default is FALSE.
<code>digits</code>	significant digits in printout.
<code>...</code>	additional arguments affecting the summary produced.

**Details**

`summary.bfi()` gives information about the MAP estimates of parameters of the model. It can be used for the bfi objects built by the `MAP.estimate` function.

The output of the summary method shows the details of the model, i.e. formula, family and link function used to specify the generalized linear model, followed by information about the estimates, standard deviations and credible intervals. Information about the log-likelihood posterior and convergence status are also provided.

By default, `summary.bfi` function does not return the curvature matrix, but the user can use `cur_mat = TRUE` to print it.

**Value**

`summary.bfi` returns an object of class `summary.bfi`, a list with the following components:

<code>theta_hat</code>	the component from object. The last element of this vector is the estimate of the dispersion parameter ( $\sigma^2$ ) if family = gaussian. See the <a href="#">MAP.estimate</a> function.
<code>A_hat</code>	the component from object. See the <a href="#">MAP.estimate</a> function.
<code>sd</code>	the component from object. If family = gaussian, the last element of this vector is the square root of the estimated dispersion. See the <a href="#">MAP.estimate</a> function.
<code>Lambda</code>	the component from object. See the <a href="#">MAP.estimate</a> function.
<code>formula</code>	the component from object. See the <a href="#">MAP.estimate</a> function.
<code>n</code>	the component from object. See the <a href="#">MAP.estimate</a> function.
<code>np</code>	the component from object. See the <a href="#">MAP.estimate</a> function.
<code>family</code>	the component from object. See the <a href="#">MAP.estimate</a> function.

intercept	the component from object. See the <a href="#">MAP.estimate</a> function.
convergence	the component from object. See the <a href="#">MAP.estimate</a> function.
control	the component from object. See the <a href="#">MAP.estimate</a> function.
estimate	the estimated regression coefficients, i.e., without the estimate sigma2.
logLikPost	the value of the log-likelihood posterior density evaluated at estimates (theta_hat).
link	the link function. By default the gaussian family with identity link function and the binomial family with logit link function are used.
dispersion	the estimated variance of the random error, i.e., sigma2. The dispersion is taken as 1 for the binomial family.
CI	a 95% credible interval of the MAP estimates of the parameters.

### Author(s)

Hassan Pazira

Maintainer: Hassan Pazira <hassan.pazira@radboudumc.nl>

### See Also

[MAP.estimate](#) and [bfi](#)

### Examples

```
#-----
# y ~ Gaussian
#-----
# model assumption:
theta <- c(1, 2, 3, 4, 1.5) # coefficients and sigma2 = 1.5

#-----
# Data Simulation
#-----
n      <- 40
X      <- data.frame(x1=rnorm(n),                # continuous variable
                    x2=sample(1:3, n, replace=TRUE)) # categorical variable
Xx2_1  <- ifelse(X$x2 == '2', 1, 0)
Xx2_2  <- ifelse(X$x2 == '3', 1, 0)
X$x2   <- as.factor(X$x2)
eta    <- theta[1] + theta[2] * X$x1 + theta[3] * Xx2_1 + theta[4] * Xx2_2
mu     <- gaussian()$linkinv(eta)
y      <- rnorm(n, mu, sd = sqrt(theta[5]))

#-----
# MAP estimations
#-----
Lambda <- inv.prior.cov(X, lambda = c(0.1, 0.5), family = gaussian)
fit    <- MAP.estimate(y, X, family = gaussian, Lambda)
class(fit)

#-----
# Summary of MAP estimates
#-----
summary(fit)
sumfit <- summary(fit, cur_mat = TRUE)
sumfit$estimate
```

```
sumfit$logLikPost  
sumfit$dispersion  
sumfit$CI  
class(sumfit)
```

---

trauma

*Trauma patients from different hospitals*

---

## Description

This data set consists of data of 371 trauma patients from three hospitals. The binary variable mortality is used as an outcome, and variables age, sex, the Injury Severity Score (ISS, ranging from 1 (low) to 75 (high)) and the Glasgow Coma Scale (GCS, which expresses the level of consciousness, ranging from 3 (low) to 15 (high)) are used as covariates. There are three types of hospitals: peripheral hospital without a neuro-surgical unit (Status = 1), peripheral hospital with a neuro-surgical unit (Status = 2), and academic medical center (Status = 3). Originally, the data come from a multi center study collected with a different aim. For educational purposes minor changes have been made, see the references below.

## Usage

```
data(trauma)
```

## References

- Jonker M.A., Pazira H. and Coolen A.C.C. (2023). *Bayesian Federated Inference for Statistical Models, Statistics in Medicine*, Vol. 0(0), 0-0. <<https://doi.org/10.48550/arXiv.2302.07677>>
- Draaisma J.M.Th, de Haan A.F.J., Goris R.J.A. (1989). *Preventable Trauma Deaths in the Netherlands - A prospective Multicentre Study*, *The journal of Trauma*, Vol. 29(11), 1552-1557.

# Index

## \* **Bayesian**

- bfi, [2](#)
- BFI-package, [2](#)
- inv.prior.cov, [6](#)
- MAP.estimation, [9](#)
- summary.bfi, [13](#)

## \* **Federated**

- bfi, [2](#)
- BFI-package, [2](#)
- MAP.estimation, [9](#)
- summary.bfi, [13](#)

## \* **datasets**

- Nurses, [12](#)
- trauma, [15](#)

## \* **package**

- BFI-package, [2](#)

bfi, [2](#), [11](#), [14](#)

BFI-package, [2](#)

family, [6](#), [9](#), [11](#)

inv.prior.cov, [4](#), [6](#), [11](#)

MAP.estimation, [4](#), [8](#), [9](#), [13](#), [14](#)

Nurses, [12](#)

stats, [10](#)

summary(summary.bfi), [13](#)

summary.bfi, [11](#), [13](#)

trauma, [15](#)