# Package 'BinaryReplicates'

January 23, 2026

**Title** Dealing with Binary Replicates

**Version** 1.0.0

**Description** Statistical methods for analyzing binary replicates, which are noisy
binary measurements of latent binary states. Provides scoring functions
(average, median, likelihood-based, and Bayesian) to estimate the probability
that an individual is in the positive state. Includes maximum a posteriori
estimation via the EM algorithm and full Bayesian inference via Stan.
Supports classification with inconclusive decisions and prevalence estimation.

**License** GPL (>= 3)

**URL** https://github.com/pierrepudlo/BinaryReplicates

**BugReports** https://github.com/pierrepudlo/BinaryReplicates/issues

**Encoding** UTF-8

**RoxygenNote** 7.3.3

**Biarch** true

**Depends** R (>= 3.5.0)

**Imports** methods, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rstan (>=
2.18.1), rstantools (>= 2.4.0), dplyr (>= 0.8.0), magrittr (>=
1.5)

**LinkingTo** BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0),
RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>=
2.18.0)

**SystemRequirements** GNU make

**Suggests** knitr, rmarkdown, tidyverse

**VignetteBuilder** knitr

**NeedsCompilation** yes

**Author** Pierre Pudlo [aut, cre] (ORCID:
<https://orcid.org/0000-0003-0995-716X>),
Manuela Royez-Carenzi [aut] (ORCID:
<https://orcid.org/0000-0003-2392-1120>),
Hadrien Lorenzo [aut] (ORCID: <https://orcid.org/0000-0002-0253-8539>)

# Contents

---

BayesianFit               *Fit the Bayesian model for Binary Replicates*

---

### Description

Fit the Bayesian model for Binary Replicates

### Usage

```
BayesianFit(
  ni,
  si,
  prior = list(a_FP = 2, b_FP = 2, a_FN = 2, b_FN = 2, a_T = 0.5, b_T = 0.5),
  ...
)
```

### Arguments

| | |
|---|---|
| ni | Numeric vector of $n_i$'s, the total numbers of replicates for each individual |
| si | Numeric vector of $s_i$'s, the numbers of replicates equal to 1 for each individual |
| prior | A list of prior parameters for the model, see Details. |
| ... | Arguments passed to rstan::sampling (e.g. iter, chains). |

## Details

The model is a Bayesian model for binary replicates. The prior distribution is as follows:

- The false positive rate: $p \sim \text{Beta}(a_{FP}, b_{FP})$
- The false negative rate: $q \sim \text{Beta}(a_{FN}, b_{FN})$
- The prevalence: $\theta \sim \text{Beta}(a_T, b_T)$

The statistical model considers that the true statuses are the latent

$$T_i \sim \text{Bernoulli}(\theta).$$

And, given the true status $T_i$, the number of positive replicates is

$$S_i \sim \text{Binomial}(n_i, T_i(1-q) + (1-T_i)p).$$

## Value

An object of class `stanfit` returned by `rstan::sampling`.

The Stan model samples the posterior distribution of the fixed parameters $p$, $q$ and $\theta$. It also generates the latent variables $T_i$ according to their predictive distribution.

## See Also

credint, bayesian_scoring, classify_with_scores, bayesian_prevalence_estimate

---

bayesian_computations *Bayesian computations*

---

## Description

Get credible intervals, Bayesian scores, and prevalence estimate from the `stanfit` object returned by BayesianFit.

## Usage

```
credint(fit, level = 0.9)

bayesian_scoring(ni, si, fit)

bayesian_prevalence_estimate(fit)
```

## Arguments

| | |
|---|---|
| `fit` | The `stanfit` object returned by BayesianFit |
| `level` | Posterior probability of the credible intervals |
| `ni` | Numeric vector of $n_i$'s, the total numbers of replicates for each individual |
| `si` | Numeric vector of $s_i$'s, the numbers of replicates equal to 1 for each individual |

## Details

See BayesianFit for details on the Bayesian model.

## Value

The credint function returns the credible interval bounds for the fixed parameters of the Bayesian model. The default posterior probability is 90%.

The bayesian_scoring function returns the Bayesian scores. These scores are the posterior probabilities that the true latent $T_i$'s are equal to 1.

The bayesian_prevalence_estimate function returns the posterior mean of the posterior distribution on the prevalence of $T_i = 1$.

## See Also

classify_with_scores, BayesianFit

## Examples

```
data("periodontal")
theta <- mean(periodontal$ti)
fit <- BayesianFit(periodontal$ni, periodontal$si, chains = 2, iter = 5000)
credint(fit)
Y_B <- bayesian_scoring(periodontal$ni, periodontal$si, fit)
T_B <- classify_with_scores(Y_B, .4, .6)
theta_B <- bayesian_prevalence_estimate(fit)
cat("The Bayesian prevalence estimate is ", theta_B, "\n")
cat("The prevalence in the data is ", theta, "\n")
```

---

classify_with_scores    *Classification based on a thresholding of the scores*

---

## Description

Classification based on a thresholding of the scores

## Usage

```
classify_with_scores(scores, vL, vU)
```

## Arguments

| | |
|---|---|
| scores | Numeric vector of the scores, computed with average_scoring, median_scoring, MAP_scoring or bayesian_scoring |
| vL | The lower threshold |
| vU | The upper threshold |

## Details

Each decision $\hat{t}_i$ is taken according to the following rule:

$$\hat{t}_i = \begin{cases} 0 & \text{if } y_i < v_L, \\ 1/2 & \text{if } v_L \leq y_i \leq v_U, \\ 1 & \text{if } y_i > v_U, \end{cases}$$

where $y_i$ is the score for individual $i$.

## Value

A numeric vector of the classification (where 0.5 = inconclusive)

---

```
classify_with_scores_cvEM
```
*Perform classification on the scores for each fold of a cvEM object*

---

## Description

Note that if $t_i$ is provided, the empirical risk is estimated with $a = v_L$.

## Usage

```
classify_with_scores_cvEM(object, ti = NULL, vL = 0.5, vU = 0.5)
```

## Arguments

| | |
|---|---|
| object | An object of class cvEM |
| ti | Numeric vector of $t_i$'s, the true values of the binary variable for each individual. If NULL, the risk is not computed. Defaults to NULL. |
| vL | The lower threshold for classification. Defaults to 0.5. |
| vU | The upper threshold for classification. Defaults to 0.5. |

## Value

A cvEM object with the following components:

**predictions** A list of the predictions for each fold

**risk** The empirical risk if $t_i$ is provided

## Examples

```
data(periodontal)
modelCV <- cvEM(periodontal$ni, periodontal$si)
modelCV2 <- classify_with_scores_cvEM(modelCV, vL = 0.4)
```

---

cvEM                          *Cross-validation for the EM algorithm*

---

### Description

Cross-validation for the EM algorithm

### Usage

```
cvEM(
  ni,
  si,
  ti = NULL,
  N_cv = NULL,
  N_init = 20,
  maxIter = 1000,
  errorMin = 1e-07,
  prior = list(a_FP = 2, b_FP = 2, a_FN = 2, b_FN = 2)
)
```

### Arguments

| | |
|---|---|
| ni | Numeric vector of $n_i$'s, the total numbers of replicates for each individual |
| si | Numeric vector of $s_i$'s, the numbers of replicates equal to 1 for each individual |
| ti | Numeric vector of $t_i$'s, the true values of the binary variable for each individual. If NULL, the EM algorithm is used to estimate the parameters. Defaults to NULL. See details. |
| N_cv | The number of folds. Defaults to 20. |
| N_init | The number of initializations if ti is not provided. Defaults to 20. |
| maxIter | The maximum number of iterations if the EM algorithm is used. Defaults to 1e3. |
| errorMin | The minimum error for convergence if the EM algorithm is used. Defaults to 1e-7. |
| prior | A list of prior parameters for the model. |

### Details

This function chooses its algorithm according to what is provided in the ti argument:

ti **is fully provided** The function computes the *Maximum-A-Posteriori* estimate, with an explicit formula.

ti **is not provided** The function uses the EM algorithm to estimate the parameters.

ti **is partially provided** The function uses the EM algorithm to estimate the parameters.

## Value

A list with the following components:

**models**  A list of the models for each fold

**predictions**  A list of the predictions for each fold

## See Also

classify_with_scores, EMFit

## Examples

```
data("periodontal")
modelCV <- cvEM(periodontal$ni, periodontal$si)
```

---

EMFit                                    *Compute the* Maximum-A-Posteriori *estimate with the EM algorithm*

---

## Description

Compute the *Maximum-A-Posteriori* estimate with the EM algorithm

## Usage

```
EMFit(
  ni,
  si,
  ti = NULL,
  prior = list(a_FP = 2, b_FP = 2, a_FN = 2, b_FN = 2),
  N_init = 20,
  maxIter = 1000,
  errorMin = 1e-07
)
```

## Arguments

| | |
|---|---|
| ni | Numeric vector of $n_i$'s, the total numbers of replicates for each individual |
| si | Numeric vector of $s_i$'s, the numbers of replicates equal to 1 for each individual |
| ti | Numeric vector of $t_i$'s, the true values of the binary variable for each individual. If NULL, the EM algorithm is used to estimate the parameters. Defaults to NULL. See details. |
| prior | A list of prior parameters for the model. The prior distribution is as follows: |

- The false positive rate: $p \sim \text{Beta}(a_{FP}, b_{FP})$
- The false negative rate: $q \sim \text{Beta}(a_{FN}, b_{FN})$

| N_init | The number of initializations if `ti` is not provided. Defaults to 20. |
| maxIter | The maximum number of iterations if the EM algorithm is used. Defaults to 1e3. |
| errorMin | The minimum error for convergence if the EM algorithm is used. Defaults to 1e-7. |

### Details

This function chooses its algorithm according to what is provided in the `ti` argument:

`ti` **is fully provided**  The function computes the *Maximum-A-Posteriori* estimate, with an explicit formula.

`ti` **is not provided**  The function uses the EM algorithm to estimate the parameters.

`ti` **is partially provided**  The function uses the EM algorithm to estimate the parameters.

### Value

A list with the following components:

**score**  The estimated values of the scores

**parameters_hat**  The estimated values of the parameters $\theta$, $p$ and $q$

### See Also

[classify_with_scores](classify_with_scores)

### Examples

```
data("periodontal")
# Get ML estimate knowing the true values of the latent ti's
periodontal_ml <- EMFit(periodontal$ni, periodontal$si, periodontal$ti)
# Get MAP estimate without knowing the true values of the latent ti's
periodontal_EM <- EMFit(periodontal$ni, periodontal$si, ti = NULL)
```

---

mammography-datasets      *A mammography dataset*

---

### Description

Data from a mammography screening program. The dataset is not the original dataset, but an imputed dataset based on the summary statistics available publicly.

### Usage

```
data(mammography)
```

```
data(observed)
```

## Format

The data frame `mammography` has 148 rows and 3 variables

**ti** True latent state of the individual (1=positive, 0=negative)

**ni** Number of mammography tests performed for each individual

**si** Number of positive mammography tests for each individual

The matrix `observed` is a 110x148 array with the observed replicates, one column for each individual. The rows correspond to the replicates, and the values are either 0 or 1.

## Source

Beam C, Conant E, Sickles E. Association of volume and volume-independent factors with accuracy in screening mammogram interpretation. *JNCI.* 2003;95:282-290.

---

| non_bayesian_scoring | *Non-Bayesian scoring methods* |
|---|---|

---

## Description

Compute the average-, median- and likelihood-based scores

## Usage

```
average_scoring(ni, si)

median_scoring(ni, si)

likelihood_scoring(ni, si, param)

MAP_scoring(ni, si, fit)
```

## Arguments

| | |
|---|---|
| ni | Numeric vector of $n_i$'s, the total numbers of replicates for each individual |
| si | Numeric vector of $s_i$'s, the numbers of replicates equal to 1 for each individual |
| param | A list with 3 entries: `theta` The probability that $T = 1$, i.e., the prevalence, `p` The false positive rate, `q` The false negative rate. |
| fit | The object returned by EMFit containing the results of the EM algorithm |

## Value

A numeric vector of the scores

## Note

For likelihood-based scores, the values of $\theta$, $p$ and $q$ are required. Consequently, likelihood scoring is not directly applicable in practice without parameter estimates.

## See Also

EMFit

## Examples

```
data("periodontal")
Y_A <- average_scoring(periodontal$ni, periodontal$si)
Y_M <- median_scoring(periodontal$ni, periodontal$si)
# In order to compute the likelihood-based scores, we need to know theta,
# p and q which can be estimated in this example as follows:
theta_hat <- mean(periodontal$ti)
cat("The prevalence in the data is ", theta_hat, "\n")
p_hat <- with(periodontal, sum(si[ti == 0]) / sum(ni[ti == 0]))
q_hat <- with(periodontal, 1 - sum(si[ti == 1]) / sum(ni[ti == 1]))
Y_L <- likelihood_scoring(periodontal$ni, periodontal$si,
                          list(theta = theta_hat, p = p_hat, q = q_hat))

data("periodontal")
Y_M <- median_scoring(periodontal$ni, periodontal$si)

data("periodontal")
fit <- EMFit(periodontal$ni, periodontal$si)
Y_MAP <- MAP_scoring(periodontal$ni, periodontal$si, fit)
```

---

periodontal                        *A periodontal dataset*

---

## Description

Data from enzymatic diagnostic tests to detect two organisms Treponema denticola and Bacteroides gingivalis.

## Usage

```
data(periodontal)
```

## Format

A data frame with 50 rows and 3 variables:

**ni** Total numbers of sites tested for each individual

**si** Number of positive tests for each individual

**ti** Status of the individual (1=infected, 0=non-infected)

## References

Hujoel PP, Moulton LH, Loesche WJ. Estimation of sensitivity and specificity of site-specific diagnostic tests. J Periodontal Res. 1990 Jul;25(4):193-6. doi: 10.1111/j.1600-0765.1990.tb00903.x. Erratum in: J Periodontal Res 1990 Nov;25(6):377. PMID: 2197400. Moore et al. (2013) Genetics 195:1077-1086 (PubMed)

## Examples

```
data(periodontal)
hat_prevalence <- mean(periodontal$si/periodontal$ni)
hat_prevalence
# should be compared to:
mean(periodontal$ti)
```

---

| predict_scores | *Compute predictive Bayesian scores* |
| --- | --- |

---

## Description

Compute predictive Bayesian scores

## Usage

```
predict_scores(newdata_ni, newdata_si, fit)
```

## Arguments

| | |
| --- | --- |
| newdata_ni | Numeric vector of the total numbers of replicates per individuals |
| newdata_si | Numeric vector of the numbers of positive replicates per individuals |
| fit | The stanfit object returned by BayesianFit |

## Details

The predict_scores function computes the predictive Bayesian scores. It computes the empirical estimator, for a new individual $n + 1$, of the following integral:

$$Y_{B,n+1} = \int Y_{L,n+1}(\theta_T, p, q)\pi(\theta_T, p, q|S_1, ..., S_n)\mathrm{d}\theta_T\mathrm{d}p\mathrm{d}q$$

where $\pi(\theta, p, q|S_1, ..., S_n)$ is the posterior distribution of the parameters $\theta$, $p$ and $q$ given the data $S_1, ..., S_n$ and $Y_{L,n+1}(\theta_T, p, q)$ is given by the function likelihood_scoring, such as

$$Y_{L,n+1}(\theta_T, p, q) = \boldsymbol{P}(T_{n+1} = 1|S_{n+1} = s_{n+1}) = \frac{\theta_T q^{n_{n+1}-S_{n+1}}(1-q)^{S_{n+1}}}{\theta_T q^{n_{n+1}-S_{n+1}}(1-q)^{S_{n+1}} + (1-\theta_T)p^{S_{n+1}}(1-p)^{n_{n+1}-S_{n+1}}}.$$

Thus the estimator is given by

$$\hat{Y}_{B,n+1} = \frac{1}{K}\sum_{k=1}^{K} Y_{L,n+1}(\theta_{T,k}, p_k, q_k),$$

where each parameter $(\theta_{T,k}, p_k, q_k)_k$ is sampled from the posterior distribution, output of the function BayesianFit. $K$ is the total number of sampled parameters.

## Value

The `predict_scores` function returns the predictive Bayesian scores in a numeric vector. The predictive Bayesian scores are the posterior probabilities that the true latent $T_i$'s are equal to 1 on new data, averaged over the posterior distribution.

## Examples

```
data("periodontal")
theta <- mean(periodontal$ti)
fitBay <- BayesianFit(periodontal$ni, periodontal$si, chains = 2, iter = 500)
fitMAP <- EMFit(periodontal$ni, periodontal$si)

## Comparison Bayesian <--> MAP
ni <- 200
Ni <- rep(ni,ni+1)
Si <- 0:ni
scores <- cbind(predict_scores(Ni,Si,fitBay),
                likelihood_scoring(Ni,Si,fitMAP$parameters_hat))
matplot(Si,scores,type = "l",lty = 1,col = 1:2,
        ylab = "Scores",xlab = "Number of Successes",main = "")
```

---

| prevalence_estimate | *Compute the average-/median- or MAP-based prevalence estimates based on the scores* |
|---|---|

---

## Description

Compute the average-/median- or MAP-based prevalence estimates based on the scores

## Usage

```
prevalence_estimate(scores)
```

## Arguments

scores          Numeric vector of the scores, computed with average_scoring, median_scoring or MAP_scoring

## Value

A numeric value of the prevalence estimate

**Note**

We have shown that the median-based prevalence estimator is better than the average-based prevalence estimator in terms of bias, except when the prevalence is in an interval $J$. The length of $J$ is small when the number of replicates is always large.

**Examples**

```
data("periodontal")
theta <- mean(periodontal$ti)
Y_A <- average_scoring(periodontal$ni, periodontal$si)
Y_M <- median_scoring(periodontal$ni, periodontal$si)
fit <- EMFit(periodontal$ni, periodontal$si)
Y_MAP <- MAP_scoring(periodontal$ni, periodontal$si, fit)
hat_theta_A <- prevalence_estimate(Y_A)
hat_theta_M <- prevalence_estimate(Y_M)
hat_theta_MAP <- prevalence_estimate(Y_MAP)
cat("The average-based prevalence estimate is ", hat_theta_A, "\n")
cat("The median-based prevalence estimate is ", hat_theta_M, "\n")
cat("The MAP-based prevalence estimate is ", hat_theta_MAP, "\n")
cat("The prevalence in the dataset is ", theta, "\n")
```

# Index