Package ‘BayesBP’

August 27, 2020

Type    Package
Title   Bayesian Estimation using Bernstein Polynomial Fits Rate Matrix
Version 1.1
Author  Li-Syuan Hong [aut, cre]
Maintainer Li-Syuan Hong <lisyuan@nhri.org.tw>
Description Smoothed lexis diagrams with Bayesian method specifically tailored to cancer incidence data. Providing to calculating slope and constructing credible interval.
License GPL (>= 2)
Encoding UTF-8
Date 2020-08-17
LazyLoad yes
LazyData yes
Depends R (>= 3.5.0),parallel,iterators,utils,stats,openxlsx
RoxygenNote 7.1.1
Suggests testthat (>= 2.1.0)
NeedsCompilation no
Repository CRAN
Date/Publication 2020-08-27 09:30:03 UTC

R topics documented:

bin .......................................................... 2
BP2D ......................................................... 2
BP2D_coef .................................................. 5
BP2D_table .................................................. 5
BPbasis ..................................................... 6
BPFhat ..................................................... 7
Credible_interval ........................................ 7
bin(n, i, x)

Arguments

n: Integer.
i: Integer(i < n).
x: Numeric(0 <= x <= 1).

Examples

bin(5, 3, .5)

BP2D

Bayesian estimation using two dimensions Bernstein polynomial

Description

This function runs Metropolis-Hasting algorithm which is given setting prior and data. This algorithm starts storing coefficients when it runs halfway, so we use second halves of coefficients to compute Rhat to check convergence.

Index

bin Binomial function
Usage

BP2D(
    prior,
    ages,
    years,
    disease,
    population,
    Iterations = 2e+05,
    n_chain = 5,
    n_cluster = 1,
    nn = 2,
    interval = 100,
    RJC = 0.35,
    seed = TRUE,
    set = 1,
    double = 4
)

Arguments

prior prior=(n0,alpha,L) where alpha is a Poisson parameter,n0 is upper bound of alpha L can be every number which is bigger than one.
ages Range of ages.
years Range of years.
disease Disease matrix.
population Population matrix.
Iterations Iterations of chain.
n_chain Number of Markov chain.
n_cluster This parameter means number of cores, five cores is recommended.(default:n_cluster=1).
nn The parameter nn is lower bound of alpha.
interval Each hundreds save one coefficient.
RJC Control parameter for transfer dimension.
seed Set seed yes or not.
set Choose seed.(defaults:set=1)
double If R.hat >1.1 then double the iterations of times.

Value

This function will return Bayesian estimate of incidence,Stored parameters,posterior mean,posterior max and table.

\( \hat{F} \text{hat} \) Bayesian estimate of incidence.

\( \text{chain} \) Bayesian estimate of posterior p-value mean.
maxchain  Bayesian estimate of posterior p-value max.
store_coefficients
Two dimensional Bernstein coefficients.
output  When M-H algorithm ends, construct the table which contains norm, mean of Fhat, maximum of Fhat, R.hat, iterations, P-value and elapsed time.

References


See Also

Other Bayesian estimate: BP2D_coef(), BP2D_table()

Examples

```r
# ---------------------------------------- #
library(BayesBP)
ages<-35:85
years<-1988:2007
prior<-c(10,5,2)
data(simulated_data_1)
disease<-simulated_data_1$disease
population<-simulated_data_1$population
result<-BP2D(prior, ages, years, disease, population)
# ---------------------------------------- #
# Bernstein basis
basis<-BPbasis(ages, years, 10)
pdbasis1<-PD_BPbasis(ages, years, 10, by = 1)
pdbasis2<-PD_BPbasis(ages, years, 10, by = 2)
# Bernstein polynomial
coef<-result$store_coefficients$chain_1[[1]]
BPFhat(coef, ages, years, basis)
PD_BPFhat(coef, ages, years, pdbasis1, by = 1)
PD_BPFhat(coef, ages, years, pdbasis2, by = 2)
# Credible interval
Credible_interval(result)
PD_Credible_interval(result, by = 1)
PD_Credible_interval(result, by = 2)
# ---------------------------------------- #
# Given four prior set
ages<-35:85
years<-1988:2007
data(simulated_data_2)
disease<-simulated_data_2$disease
population<-simulated_data_2$population
p<-expand.grid(n0=c(10,20), alpha=c(5,10), LL=c(2,4))
prior_set<-p[p$n0==p$alpha*2,]
```
result_list<-paste0('result',1:nrow(prior_set))
for (i in seq_len(nrow(prior_set))) {
  prior<-prior_set[i,]
  assign(result_list[i],BP2D(prior,ages,years,disease,population))
  write.BP(get(result_list[i]),sprintf('%s.xlsx',result_list[i]))
}

BP2D_coef(result_list)
write.BPtable(tab,'result_table.xlsx')
# ---------------------------------------- #

---

### BP2D_coef

*Getting coefficient from BP2D result.*

**Description**

This function will return coefficient and length of each set of coefficient.

**Usage**

`BP2D_coef(result)`

**Arguments**

- `result` This is output of BP2D.

**Value**

Coefficients table.

**See Also**

Other Bayesain estimate: `BP2D_table()`, `BP2D()`

---

### BP2D_table

*Table and Criterion.*

**Description**

If you give more groups of prior, you can use this function to get the table and T criterion.

**Usage**

`BP2D_table(results_list)`

**Arguments**

- `results_list` A vector of characters.
**Value**

Table and criterion T.

**See Also**

Other Bayesian estimate: `BP2D_coef()`, `BP2D()`

---

**BPbasis**

*Bernstein polynomial basis.*

---

**Description**

This function builds a two-dimensional Bernstein polynomial basis.

**Usage**

```r
BPbasis(ages, years, n0, N = 1)
```

**Arguments**

- `ages` Range of ages.
- `years` Range of years.
- `n0` Upper bound of Poisson random variable.
- `N` Lower bound of Poisson random variable.

**Value**

Bernstein basis.

**See Also**

Other Bernstein basis: `PD_BPbasis()`

**Examples**

```r
ages <- 35:85
years <- 1988:2007
list.basis <- BPbasis(ages, years, 10)
list.basis
```
BPFhat

Two dimensional Bernstein polynomial

**Description**

Given Bernstein polynomial coefficients to compute Fhat.

**Usage**

BPFhat(coef, ages, years, basis)

**Arguments**

- **coef**: Bernstein polynomial coefficients.
- **ages**: Range of ages.
- **years**: Range of years.
- **basis**: Bernstein polynomial basis.

**Value**

This function returns outer Bernstein polynomial using coefficients.

**See Also**

Other outer Bernstein polynomial: PD_BPFhat()

**Examples**

```r
coeff <- runif(9)
ages <- 35:85
years <- 1988:2007
list.basis <- BFBasis(ages, years, 10)
BPFhat(coeff, ages, years, list.basis)
```

---

Credible_interval

Credible interval.

**Description**

Building two dimensional Bernstein polynomial credible interval.

**Usage**

Credible_interval(result, n_cluster = 1, alpha = 0.05)
Arguments

result  This is output of BP2D.
n_cluster  Muticores is remmended.(default:n_cluster=1)
alpha  Level of significance.

Value

Bayesian credible interval with level of significance.

References


See Also

Other Credible interval: PD_Credible_interval()

geren_data
generated data

Description

Generated data

Usage

gen_data(ages, years, FT, M)

Arguments

ages  Ages.
years  Years.
FT  Rate function.
M  Population function.
Risky population function

Description
Risky population function

Usage
M(x, y)

Arguments
- x: Numeric.
- y: Numeric.

Partial differential Bernstein polynomial basis

Description
This function builds a two-dimensional Bernstein polynomial basis.

Usage
PD_BPbasis(ages, years, n0, N = 1, by = 1)

Arguments
- ages: Range of ages.
- years: Range of years.
- n0: Upper bound of Poisson random variable.
- N: Lower bound of Poisson random variable.

Value
Partial differential Bernstein basis.

See Also
Other Bernstein basis: BPbasis()
Examples

```r
age <- 35:85
pdbasis <- PD_BPbasis(ages, years, 10, by = 1)
pdbasis
```

---

**PD_BPFhat**

*Two dimensional Bernstein polynomial*

---

**Description**

Given Bernstein polynomial coefficients to compute Fhat.

**Usage**

```r
PD_BPFhat(coef, ages, years, pdbasis, by = 1)
```

**Arguments**

- `coef`: Bernstein polynomial coefficients.
- `ages`: Range of ages.
- `years`: Range of years.
- `pdbasis`: Partial differential Bernstein polynomial basis.

**Value**

Partial differential Bernstein polynomial given coefficients.

**See Also**

Other outer Bernstein polynomial: `BPFhat()`

**Examples**

```r
ccoef <- runif(9)
age <- 35:85
pdbasis <- PD_BPbasis(ages, years, 10, N=1, by=1)
PD_BPFhat(coef, ages, years, pdbasis, by=1)
```
**PD_Credible_interval**

_Credible interval._

**Description**

Building two dimensional Bernstein polynomial credible interval.

**Usage**

```r
PD_Credible_interval(result, n_cluster = 1, alpha = 0.05, by = 1)
```

**Arguments**

- `result`: This is output of BP2D.
- `n_cluster`: Multi-cores is recommended. (default: `n_cluster = 1`)
- `alpha`: Level of significance.

**Value**

Bayesian credible interval with level of significance.

**References**


**See Also**

Other Credible interval: `Credible_interval()`

---

**Rhat**

_Gelman Rubin statistics._

**Description**

Check Markov chains for convergence.

**Usage**

```r
Rhat(M, burn.in = 0.5)
```
Arguments

M An n x m numeric matrix of Markov Chains.
burn.in The default value 0.5 means that the second halves of chains will be used to compute.

Value

Gelman Rubin statistics.

References


---

**scale_to_01**

Scale to $[0,1]$

Description

Scale to $[0,1]$

Usage

scale_to_01(x)

Arguments

x Vector.

Examples

scale_to_01(35:85)
(35:85-35)/(85-35)
scale_to_01(runif(10))

---

**simulated_data_1**

Generate simulated data

Description

Given rate function 1 generated data.

Usage

data(simulated_data_1)
**simulated_data_2**

**Format**

list of matrix

**Examples**

```r
ages <- 35:85
years <- 1988:2007
FT1 <- function(x,y){0.00148*sin(0.5*pi*x*y)+0.00002}
simulated_data_1 <- gen_data(ages,years,FT1,M)
```

**Description**

Generate simulated data 2

**Usage**

```r
data(simulated_data_2)
```

**Format**

list of matrix

**Examples**

```r
ages <- 35:85
years <- 1988:2007
FT2 <- function(x,y){0.00148*sin(0.5*pi*x*(y+0.2))+0.00002}
simulated_data_2 <- gen_data(ages,years,FT2,M)
```

**write.BP**

**Write xlsx file**

**Description**

This function will write result of BP2D to xlsx file.

**Usage**

```r
write.BP(writedata, filename)
```

**Arguments**

- `writedata` result of BP2D(character or list).
- `filename` xlsx file name.
write.BPtable

**Write BPtable as xlsx file**

**Description**

If your environment has some result of BP2D, then you can use this function to store BPTable.

**Usage**

```r
write.BPtable(BPtable, filename)
```

**Arguments**

- **BPtable**  
  output of BP2D_table.
- **filename**  
  xlsx file name.
Index

* Bayesian estimate
  
  BP2D, 2
  BP2D_coef, 5
  BP2D_table, 5

* Bernstein basis
  
  BPbasis, 6
  PD_BPbasis, 9

* Credible interval
  
  Credible_interval, 7
  PD_Credible_interval, 11

* datasets
  
  gen_data, 8
  M, 9
  simulated_data_1, 12
  simulated_data_2, 13

* outer Bernstein polynomial
  
  BPFhat, 7
  PD_BPFhat, 10

bin, 2
BP2D, 2, 5, 6
BP2D_coef, 4, 5, 6
BP2D_table, 4, 5, 6
BPbasis, 6, 9
BPFhat, 7, 10

Credible_interval, 7, 11

gen_data, 8
M, 9

PD_BPbasis, 6, 9
PD_BPFhat, 7, 10
PD_Credible_interval, 8, 11

Rhat, 11

scale_to_01, 12
simulated_data_1, 12
simulated_data_2, 13