Package ‘BEAU’

July 18, 2011

Type Package

Title BEAU: Bayesian effect Estimation accounting for Adjustment Uncertainty

Version 1.0

Date 2011-07-18

Author Chi Wang, Giovanni Parmigiani and Francesca Dominici

Maintainer Chi Wang <chi.wang@uky.edu>

Description A Bayesian approach to estimate the effect on the outcome associated with an exposure of interest while accounting for the uncertainty in the confounding adjustment.

License GPL-2

LazyLoad yes

R topics documented:

BEAU-package ................................................................. 1
BAC ......................................................... 2
plotBAC ......................................................... 4
plotTBAC ......................................................... 4
summaryBAC ....................................................... 5
summaryTBAC ....................................................... 5
TBAC ................................................................. 6

Index 9

BEAU-package BEAU: Bayesian effect Estimation accounting for Adjustment Uncertainty

Description

A Bayesian approach to estimate the effect on the outcome associated with an exposure of interest while accounting for the uncertainty in the confounding adjustment.

Details
Package: BEAU
Type: Package
Version: 1.0
Date: 2011-07-18
License: GPL-2
URL: http://sweb.uky.edu/~cwa236/BEAU/

Author(s)
Chi Wang, Giovanni Parmigiani and Francesca Dominici
Maintainer: Chi Wang <chi.wang@uky.edu>

Examples
n_obs = 100
U1 = rnorm(n_obs); U2 = rnorm(n_obs); U3 = rnorm(n_obs)
X = 0.7 * U1 + rnorm(n_obs)
Y = 0.1 * X + 0.1 * U1 + 0.1 * U2 + rnorm(n_obs)
data = data.frame(Y=Y, X=X, U1=U1, U2=U2, U3=U3)
result = BAC(data, num_its=2000, burn=500)
summaryBAC(result)
plotBAC(result)

BAC
Bayesian Adjustment for Confounding

Description
A Bayesian approach to estimate the effect on the outcome associated with an exposure of interest while accounting for the uncertainty in the confounding adjustment.

Usage
BAC(data, num_its, burn, omega = Inf, nu = 2.58, lambda = 0.28, phi = 2.85)

Arguments
data A data frame or a n by p matrix. The first column should be data of the outcome, the second column should be data of the exposure, the rest columns should be data of potential confounders. Observations with missing values will be excluded from the analysis.
num_its Number of MCMC iterations excluding the burn-in iterations
burn Number of burn-in iterations
omega Dependence parameter, which is the prior odds of including a predictor in the outcome model, given that the same predictor is in the exposure model
nu A hyperparameter
lambda A hyperparameter
phi A hyperparameter
**Details**

The dependence parameter \( \omega \) \( (0 < \omega \leq \infty) \) controls the dependence between the outcome model and the exposure model. With \( \omega = \infty \), a predictor selected by the exposure model will be forced into the outcome model, and a predictor not selected by the outcome model will be excluded from the exposure model.

**Value**

- **output**
  A list of MCMC results. The \( \beta \) component is a vector of MCMC samples of exposure effect. The \( \alpha_X \) component is a matrix of MCMC samples of exposure model. The \( \alpha_Y \) component is a matrix of MCMC samples of outcome model.

- **data**
  data matrix used in the analysis, which is the iput data after removing observations with missing values

- **num_its**
  Number of MCMC iterations excluding the burn-in iterations

- **burn**
  Number of burn-in iterations

- **omega**
  Dependence parameter, which is the prior odds of including a predictor in the outcome model, given that the same predictor is in the exposure model

- **nu**
  A hyperparameter

- **lambda**
  A hyperparameter

- **phi**
  A hyperparameter

- **class**
  A character to indicate that the object is obtained from the BAC function

**Note**

The function may run slowly for data with large sample size or many potential confounders. The users are suggested to choose small number of iterations first, evaluate the computational speed, then increase the number of iterations.

**Author(s)**

Chi Wang, Giovanni Parmigiani and Francesca Dominici

**See Also**

summaryBAC, plotBAC

**Examples**

```r
### simulate data
n_obs = 100
U1 = rnorm(n_obs); U2 = rnorm(n_obs); U3 = rnorm(n_obs)
X = 0.7 * U1 + rnorm(n_obs)
Y = 0.1 * X + 0.1 * U1 + 0.1 * U2 + rnorm(n_obs)
data = data.frame(Y=Y, X=X, U1=U1, U2=U2, U3=U3)
### run BAC
result = BAC(data, num_its=2000, burn=500)
### summarize results
summaryBAC(result)
plotBAC(result)
### try a different omega
result1 = BAC(data, num_its=2000, burn=500, omega=2)
```
plotBAC

*Plot posterior inclusion probabilities for BAC*

**Description**

Plot the posterior inclusion probabilities of potential confounders based on the returned object from BAC.

**Usage**

```r
plotBAC(result)
```

**Arguments**

- `result` An object returned by the BAC function

**Author(s)**

Chi Wang, Giovanni Parmigiani and Francesca Dominici

**See Also**

`BAC`, `summaryBAC`

plotTBAC

*Plot posterior inclusion probabilities for TBAC*

**Description**

Plot the posterior inclusion probabilities of potential confounders based on the returned object from TBAC.

**Usage**

```r
plotTBAC(result)
```

**Arguments**

- `result` An object returned by the TBAC function

**Author(s)**

Chi Wang, Giovanni Parmigiani and Francesca Dominici

**See Also**

`TBAC`, `summaryTBAC`
**summaryBAC**

Summarize results from BAC

**Description**

Summarize results from running the BAC function

**Usage**

summaryBAC(result)

**Arguments**

- **result**: An object returned by the BAC function

**Value**

- **posterior.mean**: Posterior mean of exposure effect
- **CI**: 95% posterior interval of exposure effect
- **PIP**: A vector of posterior inclusion probabilities for potential confounders
- **beta**: MCMC samples of exposure effect excluding burn-in iterations
- **alphaX**: MCMC samples of exposure model excluding burn-in iterations
- **alphaY**: MCMC samples of outcome model excluding burn-in iterations

**Author(s)**

Chi Wang, Giovanni Parmigiani and Francesca Dominici

**See Also**

[BAC, plotBAC](#)

---

**summaryTBAC**

Summarize results from TBAC

**Description**

Summarize results from running the TBAC function

**Usage**

summaryTBAC(result)

**Arguments**

- **result**: An object returned by the TBAC function
Value

- **posterior.mean**: Posterior mean of exposure effect
- **CI**: 95% posterior interval
- **PIP**: A vector of posterior inclusion probabilities for potential confounders
- **beta**: MCMC samples of exposure effect. Samples from different exposure models are combined
- **alphaX**: MCMC samples of exposure model
- **alphaY**: MCMC samples of outcome model. Samples from different exposure models are combined

Author(s)

Chi Wang, Giovanni Parmigiani and Francesca Dominici

See Also

- TBAC
- plotTBAC

---

**TBAC**

*Two-stage Bayesian Adjustment for Confounding*

Description

A two-stage Bayesian approach to estimate the effect on the outcome associated with an exposure of interest while accounting for the uncertainty in the confounding adjustment.

Usage

```r
TBAC(data, num_its, burnX, burnY, omega = Inf, nu = 2.58, lambda = 0.28, phi = 2.85)
```

Arguments

- **data**: A data frame or a \( n \times p \) matrix. The first column should be data of the outcome, the second column should be data of the exposure, the rest columns should be data of potential confounders. Observations with missing values will be excluded from the analysis.
- **num_its**: Number of MCMC iterations excluding the burn-in iterations
- **burnX**: Number of burn-in iterations in stage one (MCMC on exposure model)
- **burnY**: Number of burn-in iterations in stage two (MCMC on outcome model and exposure effect)
- **omega**: Dependence parameter, which is the prior odds of including a predictor in the outcome model, given that the same predictor is in the exposure model
- **nu**: A hyperparameter
- **lambda**: A hyperparameter
- **phi**: A hyperparameter
TBAC is similar to BAC, except that it cuts the feedback from outcome model to exposure model. It is implemented as a two-stage procedure. In stage one, MCMC samples of exposure model are generated. In stage two, separate MCMC samples of outcome model and exposure effect will be generated for each exposure model appeared in stage one.

The dependence parameter $\omega$ ($0 < \omega \leq \infty$) controls the dependence of the outcome model on the exposure model. With $\omega = \infty$, a predictor selected by the exposure model will be forced into the outcome model.

Value

- **output**: A list of MCMC results. The length of the list is equal to the number of exposure models appeared in stage one (see Details). Each component is a list of results from a specific exposure model. The $\alpha_X$ component of that list is the exposure model considered. The $\text{count}$ component is the number of times that exposure model appeared in the MCMC samples in stage one. The $\beta$ component is a vector of MCMC samples of outcome model and exposure model. The $\alpha_Y$ component is a matrix of MCMC samples of outcome model given the exposure model.

- **data**: data matrix used in the analysis, which is the input data after removing observations with missing values.

- **num_its**: Number of MCMC iterations excluding the burn-in iterations.

- **burnX**: Number of burn-in iterations in stage one (MCMC on exposure model).

- **burnY**: Number of burn-in iterations in stage two (MCMC on outcome model and exposure effect).

- **omega**: Dependence parameter, which is the prior odds of including a predictor in the outcome model, given that the same predictor is in the exposure model.

- **nu**: A hyperparameter.

- **lambda**: A hyperparameter.

- **phi**: A hyperparameter.

- **class**: A character to indicate that the object is obtained from the TBAC function.

Note

This function runs slower than the BAC function. It may be very slow for data with large sample size or many potential confounders. The users are suggested to choose small number of iterations first, evaluate the computational speed, then increase the number of iterations.

Author(s)

Chi Wang, Giovanni Parmigiani and Francesca Dominici

See Also

- `summaryTBAC`
- `plotTBAC`
Examples

##### simulate data ############
```
n_obs = 100
U1 = rnorm(n_obs); U2 = rnorm(n_obs); U3 = rnorm(n_obs)
X = 0.7 * U1 + rnorm(n_obs)
Y = 0.1 * X + 0.1 * U1 + 0.1 * U2 + rnorm(n_obs)
data = data.frame(Y=Y, X=X, U1=U1, U2=U2, U3=U3)
```

##### run TBAC #################
```
result = TBAC(data, num_its=2000, burnX=500, burnY=500)
```

##### summarize results ########
```
summaryTBAC(result)
plotTBAC(result)
```

##### try a different omega ####
```
result1 = TBAC(data, num_its=2000, burnX=500, burnY=500, omega=2)
```

```
summaryTBAC(result1)
plotTBAC(result1)
```
Index

BAC, 2, 4, 5
BEAU-package, 1

plotBAC, 3, 4, 5
plotTBAC, 4, 6, 7

summaryBAC, 3, 4, 5
summaryTBAC, 4, 5, 7

TBAC, 4, 6, 6