Package 'BAREB'

November 24, 2018

Type Package

Title BAREB: A Bayesian Repulsive Biclustering Model for Periodontal
Data

Version 1.0

Date 2018-11-24

Author Yuliang Li <yli193@jhu.edu>, Yanxun Xu <yxu.jhu@gmail.com>, Dipankar Bandyopad-
hyay <bandyopd@gmail.com>

Maintainer Yuliang Li <yli193@jhu.edu>

Description A Bayesian repulsive biclustering method that can simultaneously cluster the Periodon-
tal diseases (PD) patients and their tooth sites after taking the patient- and site-level covari-
ates into consideration. BAREB uses the determinantal point process (DPP) prior to induce di-
versity among different biclusters to faciliate parsimony and interpretability. Essen-
tially, BAREB is a cluster-wise linear model.

Depends R (>= 3.4.3)

License Johns Hopkins University

Imports Rcpp (>= 0.12.14), RcppArmadillo

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.0.1

R topics documented:

BAREB   *The BAREB package: summary information*

Description

This package implements BAREB (A Bayesian Repulsive Biclustering Model for Periodontal Data)

Main Features

The functions in the pacakge implement BAREB. BAREB can simultaneously cluster periodontal disease patients and their tooth site after taking the patient- and site-level covariates into consideration. BAREB uses the determinantal point process prior to induce diversity among different biclusters to facilitate parsimony and interpretability. In addition, since periodontal diseases are the leading cause for tooth loss, the missing data mechanism is non-ignorable. Such nonrandom missingness is incorporated into BAREB.

Functions

The main functions are `updateBeta`, `update.theta.beta`, `update.theta.gamma`, `updatec`, `updateE`, `updateGamma`, `updatemu`, `updatemustar`, `updateR`, `updateZstar`, `update_RJ`, `update_sigma_square`, `update_w`, and `update_w_beta`; other functions intended for direct access by the user are: `kernelC` and `updateC`. There are undocumented functions which are called by these.

Requirements

R version >= 3.4.3. Packages `Rcpp` and `RcppArmadillo` are used so that complicated functions are implemented in C++ to speed up.

Version

This is version 1.0.

Licence

This package and its documentation are usable under the terms of the "GNU General Public License", a copy of which is distributed with the package.

Author(s)

Yuliang Li (Dept Applied Mathematics and Statistics, Johns Hopkins University, USA) and Yanxun Xu (Dept Applied Mathematics and Statistics, Johns Hopkins University, USA) and Dipankar Bandyopadhyay (Dept Biostatistics, Virginia Commonwealth University, USA) . Please send comments, error reports, etc. to the maintainer (Yuliang) via email.

---

initial *The initial value of patient- and site-level covariates for simulation*

Description

These data record the initial value of patient- and site-level covariates in simulation in the paper "BAREB: A Bayesian Repulsive Biclustering Model for Periodontal Data". It is obtained by simple linear regression.

The variables are:

Beta0	the initial value of patient-level covariates
Gamma0	the initial value of site-level covariates

Usage

```
data("initial")
```

Examples

```
# output patient level covariates
data("initial")
Beta0
Gamma0
```

---

kernelC *The function to get the kernel function value*

Description

This function take two configurations x and y, two parameters of the kernel function and returns its kernel function value.

Usage

```
kernelC(x,y,theta,tau)
```

Arguments

x	a numeric vector, representing one configuration
y	a numeric vector, representing one configuration
theta	a parameter of the DPP's kernel function
tau	a parameter of the DPP's kernel function

Value

kernelC(x,y,theta,tau) returns the value of the kernel function

Author(s)

Yuliang Li

See Also

update_RJ for a complete example for all functions in this package.

Examples

```
x <- rnorm(5)
y <- rnorm(5) + 1
kernelC(x,y,1,1)
```

---

simobs	*The simulation observation*

Description

These data record the simulation observation (observed value and missing indicator) and covariates in the paper "BAREB: A Bayesian Repulsive Biclustering Model for Periodontal Data".

The variables are:

delta	the missing indicator matrix
X	the patient level covariates
Y	the observed (CAL) value matrix
Z	the site level covariates

Usage

```
data("simobs")
```

Examples

```
# output patient level covariates
data("simobs")
X
```

---

simtruth	*The simulation truth*

Description

These data record the simulation truth in the paper "BAREB: A Bayesian Repulsive Biclustering Model for Periodontal Data". It includes the true simulated parameters.

The variables are:

S	the number of patient level clusters
E	the clustering membership of patient level
K	the numbers of site level clusters
R	the site level clustering membership
Beta	the patient level linear coefficients
Gamma	the site level linear coefficients
mu	the underlying mean for CAL values

sigma_square	the variance of noise for CAL values	
noise	the noise for CAL values	
c	the parameter for missingness model	
mu.star	the mean of latent values for missingness model	
z.star	the latent values for missingness model	

Usage

```
data("simtruth")
```

Examples

```
# output true patient level clustering membership
data("simtruth")
truth$E

#get the details of the list
str(truth)
```

---

update.theta.beta   *update.theta.beta*

Description

Update the DPP hyper-parameter for patient level

Usage

```
## S3 method for class 'theta.beta'
update(theta, tau, Beta, sig = 10)
```

Arguments

theta	The DPP hyper-parameter for patient level
tau	A fixed DPP hyper-parameter, which we suggest high value, say 10^5
Beta	The linear coefficients for patient level covariates. Should be a matrix with S rows
sig	The hyper-parameter (sd) with default value being 10

Value

updated DPP hyper-parameter for patient level

See Also

update_RJ for a complete example for all functions in this package.

---

update.theta.gamma *update.theta.gamma*

Description

Update the DPP hyper-parameter for site level

Usage

```
## S3 method for class 'theta.gamma'
update(theta, tau, Gamma, S, Ds, sig = 10)
```

Arguments

theta	The DPP hyper-parameter for patient level
tau	A fixed DPP hyper-parameter, which we suggest high value, say 10^5
Gamma	The linear coefficients for site level covariates. Should be a 3-dimensional array.
S	The number of patient level clusters
Ds	The number for site level clusters for each patient level cluster
sig	The hyper-parameter (sd) with default value being 10

Value

updated DPP hyper-parameter for site level

See Also

update_RJ for a complete example for all functions in this package.

---

updateBeta *Function to update patient level linear coefficients in the BAREB model*

Description

This function takes current parameters and observed data, gives an updated patient level linear coefficients.

Usage

```
updateBeta(X, Y, Z, delta, Beta, Gamma, E, R, S, Ds, mustar, mu, sigma, c, C, step, runif, n, m, T0, p,
```

Arguments

X	the patient level covariate matrix
Y	the CAL observation matrix, with missing values
Z	the site level covariate matrix
delta	the missing indicator matrix, with 1 means missing
Beta	current patient level linear coefficients matrix
Gamma	current site level linear coefficients array
E	current patient level clustering vector
R	current site level clustering matrix
S	number of patient level clusters
Ds	a vector recording numbers of site level clusters
mustar	current matrix of latent value for missingness model
mu	current estimated mean matrix for CAL
sigma	current estimated noise variance
c	current c for missingness model. It is a vector
C	current kernel matrix for DPP
step	a matrix of steps for M-H
runif	a matrix of uniform random variables for deciding whether to accept new proposed point in M-H
n	number of patients
m	number of sites
T0	number of teeth
p	dimension of patient level covariates
q	dimension of site level covariates
D	the D matrix in the paper
theta	parameter for DPP
tau	parameter for DPP

Value

updateBeta(X, Y, Z, delta, Beta, Gamma, E, R, S, Ds, mustar, mu, sigma, c, C, step, runif, n, m, T0, p,
returns a list with following variables:

C	the updated kernel matrix computed by updated Beta
Beta	the updated patient level linear coefficients
mu	the updated mu computed by updated Beta
mustar	the updated mustar computed by updated Beta

Author(s)

Yuliang Li

See Also

update_RJ for a complete example for all functions in this package.

updateC *Function to obtain the kernel matrix of the determinantal point process*

Description

This function takes a matrix and two parameters of kernel function for the determinantal point process (DPP) and gives the kernel matrix for that DPP.

Usage

```
updateC(Z,theta,tau)
```

Arguments

Z	a matrix, whose rows stand for configurations of the DPP.
theta	a parameter of the DPP's kernel function
tau	a parameter of the DPP's kernel function

Value

`updateC(Z,theta,tau)` returns the kernel matrix

Author(s)

Yuliang Li

See Also

update_RJ for a complete example for all functions in this package.

Examples

```
Z <- matrix(rnorm(15), nrow = 5)
updateC(Z,1,1)
```

updatec *Function to update c in missingess model*

Description

This function takes current parameters, gives updated c in missingess model. Note a double type of value is returned

Usage

```
updatec(Zstar, mu,  D, sigmac, n, T0)
```

Arguments

Zstar	generated latent value for missingness model
mu	current estimated mean matrix for CAL
D	the D matrix in the paper
sigmac	hyperparamter for c (variance)
n	number of patients
T0	number of teeth

Value

updatec(Zstar, mu, D, sigmac, n, T0) returns the updated c in missingess model.

Author(s)

Yuliang Li

See Also

update_RJ for a complete example for all functions in this package.

---

updateE	*Function to update patient level clustering in the BAREB model*

Description

This function takes current parameters and observed data, gives an updated patient level clustering.

Usage

updateE(Beta, Gamma,w, X, Y, Z, delta,E, R, S, Ds, mu, mustar, sigma, c, n, m, T0, p, q, D)

Arguments

Beta	current patient level linear coefficients matrix
Gamma	current site level linear coefficients array
w	current patient level clustering prior prob, a vector
X	the patient level covariate matrix
Y	the CAL observation matrix, with missing values
Z	the site level covariate matrix
delta	the missing indicator matrix, with 1 means missing
E	current patient level clustering vector
R	current site level clustering matrix
S	number of patient level clusters
Ds	a vector recording numbers of site level clusters
mu	current estimated mean matrix for CAL
mustar	current matrix of latent value for missingness model

sigma	current estimated noise variance
c	current c for missingness model. It is a vector
n	number of patients
m	number of sites
T0	number of teeth
p	dimension of patient level covariates
q	dimension of site level covariates
D	the D matrix in the paper

Value

updateE(Beta, Gamma,w, X, Y, Z, delta,E, R, S, Ds, mu, mustar, sigma, c, n, m, T0, p, q, D) returns a list with following variables:

E	the updated patient level clustering
Ds	new vector recording the numbers of site level clusters
mu	the updated mu computed by updated E
mustar	the updated mustar computed by updated E

Author(s)

Yuliang Li

See Also

update_RJ for a complete example for all functions in this package.

---

updateGamma	*Function to update site level linear coefficients in the BAREB model*

Description

This function takes current parameters and observed data, gives an updated site level linear coefficients.

Usage

updateGamma(X, Y, Z, delta, Beta, Gamma, E, R, S, Ds, mu, mustar, sigma, c, step, runif, n, m, T0, p, q

Arguments

X	the patient level covariate matrix
Y	the CAL observation matrix, with missing values
Z	the site level covariate matrix
delta	the missing indicator matrix, with 1 means missing
Beta	current patient level linear coefficients matrix
Gamma	current site level linear coefficients array

E	current patient level clustering vector
R	current site level clustering matrix
S	number of patient level clusters
Ds	a vector recording numbers of site level clusters
mu	current estimated mean matrix for CAL
mustar	current matrix of latent value for missingness model
sigma	current estimated noise variance
c	current c for missingness model. It is a vector
step	an array of steps for M-H
runif	an array of uniform random variables for deciding whether to accept new proposed point in M-H
n	number of patients
m	number of sites
T0	number of teeth
p	dimension of patient level covariates
q	dimension of site level covariates
D	the D matrix in the paper
theta	parameter for DPP
tau	parameter for DPP

Value

updateGamma(X, Y, Z, delta, Beta, Gamma, E, R, S, Ds, mu, mustar, sigma, c, step, runif, n, m, T0, p, q
returns a list with following variables:

Gamma	the updated site level linear coefficients
mu	the updated mu computed by updated Gamma
mustar	the updated mustar computed by updated Gamma

Author(s)

Yuliang Li

See Also

update_RJ for a complete example for all functions in this package.

---

updatemu   *Function to update estimated mean CAL values based on current pa-*
                            rameters

Description

This function takes current parameters, gives an estimated mean CAL values.

Usage

```
updatemu(R, Z, X, Gamma, K, Beta, E,  m,n,p, q)
```

Arguments

R	current site level clustering matrix
Z	the site level covariate matrix
X	the patient level covariate matrix
Gamma	current site level linear coefficients array
K	a vector recording numbers of site level clusters
Beta	current patient level linear coefficients matrix
E	current patient level clustering vector
m	number of sites
n	number of patients
p	dimension of patient level covariates
q	dimension of site level covariates

Value

updatemu(R, Z, X, Gamma, K, Beta, E, m,n,p, q) returns the updated estimated mean CAL
matrix.

Author(s)

Yuliang Li

See Also

update_RJ for a complete example for all functions in this package.

---

updatemustar   *Function to update mean latent values for missingness model*

Description

This function takes current parameters, gives updated mean latent values for missingness model

Usage

```
updatemustar(mu, c, n, T0, D)
```

Arguments

mu	current estimated mean matrix for CAL
c	current c for missingness model
n	number of patients
T0	number of teeth
D	the D matrix in the paper

Value

updatemustar(mu, c, n, T0, D) returns the updated mean latent values for missingness model.

Author(s)

Yuliang Li

See Also

update_RJ for a complete example for all functions in this package.

---

updateR   *Function to update site level clustering in the BAREB model*

Description

This function takes current parameters and observed data, gives an updated site level clustering.

Usage

```
updateR( w ,Gamma,Beta, Y, Z, delta, mu, mu_star, c, S, sigma_square, K, E, X, m,n, p, q, T0)
```

Arguments

w	current site level clustering prior prob, a matrix
Gamma	current site level linear coefficients array
Beta	current patient level linear coefficients matrix
Y	the CAL observation matrix, with missing values
Z	the site level covariate matrix
delta	the missing indicator matrix, with 1 means missing
mu	current estimated mean matrix for CAL
mu_star	current matrix of latent value for missingness model
c	current c for missingness model
S	number of patient level clusters
sigma_square	current estimated noise variance
K	a vector recording numbers of site level clusters
E	current patient level clustering vector
X	the patient level covariate matrix
m	number of sites
n	number of patients
p	dimension of patient level covariates
q	dimension of site level covariates
T0	number of teeth

Value

updateR(w ,Gamma,Beta, Y, Z, delta, mu, mu_star, c, S, sigma_square, K, E, X, m,n, p, q, T0) returns the updated site level clustering.

Author(s)

Yuliang Li

See Also

update_RJ for a complete example for all functions in this package.

---

updateZstar	*Function to generate new latent values for missingness model*

Description

This function takes current parameters, gives updated c in missingess model. Note a double type of value is returned

Usage

```
updateZstar(mu_star, delta, n, T0)
```

Arguments

mus_tar	current matrix of latent value for missingness model
delta	the missing indicator matrix, with 1 means missing
n	number of patients
T0	number of teeth

Value

updateZstar(mu_star, delta, n, T0) returns a matrix of new generated latent values.

Author(s)

Yuliang Li

See Also

update_RJ for a complete example for all functions in this package.

---

update_RJ	*update_RJ*

Description

Update the number of site level clusters for each patient level cluster via RJMCMC

Usage

```
update_RJ(w, K, Gamma, Beta, Z, X, R, mu, mu_star, Y, delta, c, sigma_square, C,
  S, theta, q, hyper_delta = 1)
```

Arguments

w	The weights for site level clusters. Should be a matrix with S rows.
K	The number of site level cluster for each patient level cluster. Should be a vector of length S
Gamma	The linear coefficients for site level covariates. Should be a 3-dimensional array.
Beta	The linear coefficients for patient level covariates. Should be a matrix with S rows
Z	The design matrix for site level clusters.
X	The design matrix for patient level clusters.
R	The current site level clustering membership
mu	The current CAL mean matrix
mu_star	The latent value matrix for missingness model
Y	The observed CAL value matrix
delta	The missing indicator matrix
c	The linear coefficients for
sigma_square	The current noise variance

C	The DPP related kernel matrices. Should be an array of 3 dimensions
S	The number of patient level clusters.
theta	The DPP hyper-parameter for site level
q	The number of site level covariates
hyper_delta	The hyper-parameter with default value being 1

Value

A list with following updated parameters:

K	The numbers of site level clusters
w	The weights for site level clusters
Gamma	Linear coefficients for site level covariates
R	The site level clusteirng membership
C	The DPP related kernel matrice

Examples

```
rm(list=ls())
library(BAREB)
data("simobs")
data("simtruth")

set.seed(1)

n<-80
m<-168
T0<-28
q<-3
p<-3
S<-3
theta1 <- theta2 <- 5
tau <- 100000
D<-matrix(0, nrow = T0, ncol = m)
for(i in 1:T0){
  indi<-1:6
  indi<-indi+6*(i-1)
  D[i,indi]<-rep(1/6,6)
}
nu_gamma<-0.05
nu_beta <- 0.05
data("initial")

Niter<-5000
record <- NULL
record$E<-matrix(NA,nrow = Niter, ncol = n)
record$R<-array(NA, dim = c(Niter, S, m))
record$Gamma <-array(NA,dim = c(Niter, 10, q, S))
record$Beta <- array(NA,dim = c(Niter, S, p))
record$K <- matrix(NA,nrow = Niter, ncol = S)
record$sigma_square <-rep(NA,Niter)
record$theta1<-rep(0,Niter)
record$theta2<-rep(0,Niter)
record$c<-matrix(0,nrow =  Niter,ncol = T0)
```

```
record$mu<-array(NA,dim = c(Niter,n,m))
record$w_beta <- array(NA, dim = c(Niter, S))
record$w <- array(NA, dim = c(Niter, S, 10))
set.seed(1)
E<-sample.int(S,n,T)
Beta <- matrix(NA,nrow = S, ncol = p)
Beta[1,] <- Beta[2,] <- Beta[3,]  <- Beta0
Beta<- Beta + matrix(rnorm(S*p, 0, 1) , nrow = S, ncol = p)
Gamma <- array(NA,dim = c(10,q,S))
Gamma[1,,1] <- Gamma[2,,1] <- Gamma[3,,1] <- Gamma0
Gamma[,,2] <- Gamma[,,3]    <- Gamma[,,1]
Gamma <- Gamma + array(rnorm(10*q*S, 0, 5), dim = c(10, q, S))
K <- rep(3,S)
R <- matrix(NA,nrow = S, ncol = m)
R[1,]<- R[2,]<- R[3,]  <- sample.int(3,m,T)
mu<-updatemu(R,Z,X,Gamma,K,Beta,E,m,n,p,q)
mu_star<-updatemustar(mu,rep(0.01,T0),n,T0,D)
z_star<-updateZstar(mu_star,delta,n,T0)
sigma_square <- 10
C<-array(NA,dim = c(10,10,S))
w<-matrix(NA,nrow = S, ncol = 10)
w_beta<-rep(1/S,S)
for(i in 1:S){
 C[1:K[i],1:K[i],i]<-updateC(Gamma[1:K[i],,i],theta2,tau)
  w[i, 1:K[i]]<-rep(1/K[i],K[i])
}
c<-0.01
start <- Sys.time()
for(iter in 1:Niter){
  c<-updatec(z_star, mu,D, 100, n, T0)
  mu_star<-updatemustar(mu,rep(c,T0),n,T0,D)
  z_star<-updateZstar(mu_star,delta,n, T0)
  w <- update_w(K, w, R, S)
  R <- updateR(w, Gamma, Beta,
               Y, Z, delta, mu, mu_star, c, S,
               sigma_square, K, E, X,
               m, n, q, p, T0)
  for(i in unique(E)){
    ind<-sort(unique(R[i,]))
    KK<-length(ind)
    Gamma_temp<-Gamma[ind,,i]
    Gamma[,,i]<-NA
    Gamma[1:KK,,i]<-Gamma_temp
    w_temp<-w[i,ind]
    w_temp<-w_temp/sum(w_temp)
    w[i,]<-NA
    w[i,1:KK]<-w_temp
    for(k in 1:KK){
      R[i,which(R[i,]==ind[k])]<-k
    }
    K[i]<-KK
  }
  mu<-updatemu(R,Z,X,Gamma,K,Beta,E,m,n,p,q)
  mu_star<-updatemustar(mu,rep(c,T0),n,T0,D)
  step <- array(rnorm(max(K) * S *q, 0, nu_gamma),dim=c( max(K), q,S))
  run<- array(runif(max(K) * S *q, 0, 1),dim=c( max(K), q,S))
 A<-updateGamma(X,Y, Z, delta, Beta, Gamma, E, R, S, K , mu, mu_star, sigma_square, rep(c,T0),
```

```
                    step,  run,  n,  m,  T0,  p,  q,  D,theta2,  tau)
Gamma<-A$Gamma
mu<-A$mu
mu_star<-A$mustar
for(i in 1:S){
  if(K[i]==1){
    Gammai = t(as.matrix(Gamma[1:K[i],,i]))
    C[1:K[i],1:K[i],i]<-updateC(Gammai,theta2,tau)
  }
  else{
    C[1:K[i],1:K[i],i]<-updateC(Gamma[1:K[i],,i],theta2,tau)
  }
}
A<-update_RJ(w, K, Gamma,Beta,
             Z, X, R, mu, mu_star, Y, delta, c,sigma_square, C,
             S, theta2)


K<-A$K
w<-A$w
Gamma<-A$Gamma
R<-A$R
C<-A$C
mu<-updatemu(R,Z,X,Gamma,K,Beta,E,m,n,p,q)
mu_star<-updatemustar(mu,rep(c,T0),n,T0,D)
C_beta<-updateC(Beta,theta1,tau)
step<-matrix(rnorm(S*p,0,nu_beta),nrow = S)
runif<-matrix(runif(S*p,0,1),nrow = S)
A<- updateBeta( X,Y, Z, delta,
          Beta, Gamma, E,R,S,K,mu_star, mu,sigma_square,rep(c,T0), C_beta, step,runif,
               n, m, T0,  p,  q,  D,
               theta1, tau)
Beta<-A$Beta
mu<-A$mu
mu_star<-A$mustar
record$Beta[iter,,]<-Beta
A<-updateE( Beta,Gamma, w_beta, X,Y,Z,delta,E, R, S,K, mu, mu_star,sigma_square,rep(c,T0),
          n,  m,  T0,  p,  q,  D)
E<-A$E
mu<-A$mu
mu_star<-A$mustar
K<-A$Ds
record$K[iter,]<-K
record$R[iter,,]<-R
record$Gamma[iter,,,]<-Gamma
record$E[iter,]<-E
record$mu[iter,,] <- mu
w_beta<- update_w_beta(S, w_beta, E)
record$w[iter,,] <- w
record$w_beta[iter,] <- w_beta
theta1<-update.theta.beta(theta1,tau,Beta)
theta2<-update.theta.gamma(theta2,tau,Gamma,S,K)
sigma_square <- update_sigma_squre(Y,mu)
record$sigma_square[iter] <- sigma_square
}
```

---

update_sigma_squre	*update_sigma_squre*

Description

Update the noise variance

Usage

```
update_sigma_squre(Y, mu, a = 1, b = 1)
```

Arguments

Y	
a	The hyper-parameter with default value being 1
b	The hyper-parameter with default value being 1

Value

Updated noise variance

See Also

update_RJ for a complete example for all functions in this package.

---

update_w	*update_w*

Description

This function updates the weights for site level clusters

Usage

```
update_w(K, R, S, hyper_delta = 1)
```

Arguments

K	The number of site level cluster for each patient level cluster. Should be a vector of length S
R	The current site level clustering membership. Should be a matrix with S rows.
S	The number of patient level clusters.
hyper_delta	The hyper-parameter with default value being 1

Details

It returns a matrix with 10 columns. For example, first patient cluster has 2 site level clusters. The first row's first 2 elements give the weights for site level clusters in patient cluster 1. Last 8 elements are NA's

Value

The updated weights for site level clusters. Should be a matrix with S rows.

See Also

[update_RJ](#) for a complete example for all functions in this package.

Examples

```
#Suppose we know the number of patient level cluster is 2, one has 2 site level clusters and one has 3.
#Use the default value, 1, for hyper-parameter
update_w(K=c(2,3),R=matrix(c(1,2,2,1,1,2,3,2),nrow=2,byrow=T), S=2)

#To change the hyper-parameter to, for example 2
update_w(K=c(2,3),R=matrix(c(1,2,2,1,1,2,3,2),nrow=2,byrow=T), S=2, hyper_delta = 2)
```

---

 update_w_beta   *update_w_beta*

Description

This function updates the weights for each patient level cluster

Usage

```
update_w_beta(S, E, hyper_delta = 1)
```

Arguments

S	The number of patient level clusters
E	A vector that records the current clustering membership.
hyper_delta	The hyper-parameter with default value being 1

Value

The updated weights for each patient level cluster

See Also

[update_RJ](#) for a complete example for all functions in this package.

Examples

```
#Suppose we know the number of patient level cluster is 4
#Suppose the current clustering membership indicates 3 patients in cluster 1, 2 patients in cluster 2, 3 patinet
#Use the default value, 1, for hyper-parameter
update_w_beta(S=4,E=c(1,1,1,2,2,3,3,3,4))

#To change the hyper-parameter to, for example 2
update_w_beta(S=4,E=c(1,1,1,2,2,3,3,3,4),hyper_delta = 2)
```

Index