Code_R

Siliangyu Cheng

August 12, 2017

```
## Standardized group mean difference
########################

balm=function(d_obs,n_obs,type,random=TRUE,side=1,pn=1,cpoint=0.5,
              M=15000,burn=15000,thin=35,adjust=TRUE,w_prior=c(0.1,0.1),
              mu_prior=c(0, 10000),tau_prior=c(0, 100),
              mu_start=0, tau_start=0.2) {
  # d_obs reads in the observed effect sizes from the studies included in a meta-analysis.
  # n_obs reads in the corresponding sample sizes of the studies.
  # type specifies the suppression type of unpublished studies: 1 for suppression due to nonsignificant
  #   or due to both nonsignificance and unexpected direction, and 2 for suppression due to unexpected
  # side specifies the type of alternative hypothesis: 1 for two-sided test, 2 for u1-u1 > 0,
  #   and 3 for u1-u1 < 0
  # pn specifies the expected direction: 1 for u1-u1 > 0 as expected direction,
  #   and 2 for u1-u1 < 0 as expected direction. cpoint=0.5 needs to be specified to distinguish positi
  #   vs. negative results.
  # cpoint specifies the cutoff value for p-value intervals.
  # burn and thin specify the burn-in period and the thinning period respectively.
  # M specifies the total of iterations of the Markov Chains including the burn-in and thinning periods
  # adjust=T is to obtain unbiased values if the g values are read in.
  # w_prior specifies a beta prior distribution for w.
  # mu_prior specifies a normal prior distribution for mu.
  # tau_prior specifies a uniform prior distribution for tau.
  # mu_start specifies a starting value for mu.
  # tau_start specifies a starting value for tau.

  n1<- n_obs[,1]
  n2<- n_obs[,2]
  cm<- (1-3/(4*(n1+n2-2)-1))
  if (adjust=="TRUE") {
    d_obs<- cm*d_obs
   }

  nk<- length(cpoint)+1
  index<- matrix(0,nrow=nk,ncol=2)
  j<- matrix(NA,nrow=length(d_obs),ncol=nk)
  index[1,]<- c(0,cpoint[1])

  if (type==1) {
    if (side==1) {
      p<- 2*(1-pt(abs(d_obs/cm)*sqrt(n1/(n1+n2)*n2),df=n1+n2-2))
     } else if (side==2) {
      p<-1- pt(d_obs/cm*sqrt(n1/(n1+n2)*n2),df=n1+n2-2)
     } else if (side==3) {
      p<- pt(d_obs/cm*sqrt(n1/(n1+n2)*n2),df=n1+n2-2)
     }
   } else if (type==2) {
```

```
  if (pn==1) {
    p<- 1-pt(abs(d_obs/cm)*sqrt(n1/(n1+n2)*n2),df=n1+n2-2)
   } else if (pn==2) {
    p<- pt(d_obs/cm*sqrt(n1/(n1+n2)*n2),df=n1+n2-2)
   }
 }

if (length(which(p>index[1,1]&p <=index[1,2], arr.ind = TRUE))==0) {
  j[,1]<- rep(NA,length(d_obs))
 }else {
  j[1:length(which(p>index[1,1]&p <=index[1,2], arr.ind = TRUE)),1]<- which(p>index[1,1]&p <=index[1,

if (nk>2) {
  for (k in 2:(nk-1)) {
    index[k,]<-c (cpoint[k-1],cpoint[k])
    if (length(which(p>index[k,1]&p <=index[k,2], arr.ind = TRUE))==0) {
      j[,k]<- rep(NA,length(d_obs))
     }else {
      j[1:length(which(p>index[k,1]&p <=index[k,2], arr.ind = TRUE)),k]<- which(p>index[k,1]&p <=inde
   }
 }

index[nk,]<-c(cpoint[nk-1],1)
if (length(which(p>index[nk,1]&p <=index[nk,2], arr.ind = TRUE))==0) {
  j[,nk]<- rep(NA,length(d_obs))
 }else {
  j[1:length(which(p>index[nk,1]&p <=index[nk,2], arr.ind = TRUE)),nk]<- which(p>index[nk,1]&p <=inde

a<- length(d_obs)
data<- matrix(NA,nrow=a,ncol=3*nk)
for (k in 1:nk) {
  data[,k]<- c(n1[na.omit(j[,k])],rep(NA,a-length(na.omit(j[,k]))))
  data[,nk+k]<- c(n2[na.omit(j[,k])],rep(NA,a-length(na.omit(j[,k]))))
  data[,2*nk+k]<- c(d_obs[na.omit(j[,k])],rep(NA,a-length(na.omit(j[,k]))))
 }

w<-matrix(0,nrow=M,ncol=nk)
prefix1 <- "w"
suffix <- seq(1:nk)
names1 <- paste(prefix1, suffix, sep = "")
colnames(w)<-names1
m<-matrix(0,nrow=M,ncol=nk)
prefix2 <- "m"
names2 <- paste(prefix2, suffix, sep = "")
colnames(m)<-names2

alpha<- w_prior[1]
beta<- w_prior[2]
a<- mu_prior[1] #mu prior
b<- mu_prior[2] #mu prior
c<- tau_prior[1] # tau prior
d<- tau_prior[2] # tau prior
w[1,]<- rep(0.5,nk) #w initial value
```

```
mu<- c()
mu[1]<- mu_start
tau<- c()
tau[1]<- tau_start
w[1,]<- w[1,]/max(w[1,])

for(t in 2:M) {

  logi<- rep(0,10^3)

  k<- which(w[t-1,]==1)
  if (length(k)>1) {k=1 }
  m[t,k]<- 0

  n_mis<- n_obs[c(sample(c(1:length(d_obs)), 10^3-length(d_obs), replace = T)),]
  n_up<- rbind(n_obs,n_mis)
  td<- rnorm(10^3,mean=mu[t-1], sd=tau[t-1])
  v<- (n_up[,1]+n_up[,2])/n_up[,1]/n_up[,2]+ td^2/(2*(n_up[,1]+n_up[,2]))
  d_up<-rnorm(10^3,mean=td, sd=sqrt(v))
  n1<- n_up[,1]
  n2<- n_up[,2]
  cm<- (1-3/(4*(n1+n2-2)-1))
  if (type==1) {
    if (side==1) {
      p_up<- 2*(1-pt(abs(d_up/cm)*sqrt(n_up[,1]/(n_up[,1]+n_up[,2])*n_up[,2]),df=n_up[,1]+n_up[,2]-2)
     } else if (side==2) {
      p_up<- 1-pt(d_up/cm*sqrt(n_up[,1]/(n_up[,1]+n_up[,2])*n_up[,2]),df=n_up[,1]+n_up[,2]-2) } else
        p_up<- pt(d_up/cm*sqrt(n_up[,1]/(n_up[,1]+n_up[,2])*n_up[,2]),df=n_up[,1]+n_up[,2]-2) }
   }else if (type==2) {
    if (pn==1) {
      p_up<- 1-pt(abs(d_up/cm)*sqrt(n_up[,1]/(n_up[,1]+n_up[,2])*n_up[,2]),df=n_up[,1]+n_up[,2]-2)
     } else if (pn==2) {
      p_up<- pt(d_up/cm*sqrt(n_up[,1]/(n_up[,1]+n_up[,2])*n_up[,2]),df=n_up[,1]+n_up[,2]-2)
     }
   }

  logi[which(p_up > index[k,1] & p_up <= index[k,2], arr.ind = TRUE)]<- 1
  logi<- order(logi, decreasing=T)
  et<- logi[length(na.omit(j[,k]))]
  if(length(et)==0) {et=0 }
  m2<- et-length(d_obs)
  if (m2<0) {m2=0 }

  n_up<- n_up[1:(m2+length(d_obs)),]
  d_up<- d_up[1:(m2+length(d_obs))]
  p_up<- p_up[1:(m2+length(d_obs))]


  for (k in 1:nk) {
    m[t,k]<- length(which(p_up>index[k,1] & p_up <=index[k,2], arr.ind = TRUE))-length(na.omit(j[,k]))
    if(m[t,k]<0) {
      m[t,k]= 0
      id1<- which(p_up>index[k,1] & p_up <=index[k,2], arr.ind = TRUE)
```

```
       nid<- sample(1:length(na.omit(j[,k])),length(id1))
       d_up[id1]<- na.omit(data[,2*nk+k])[nid]
       n_up[id1,]<- na.omit(cbind(data[,k],data[,nk+k]))[nid,]
      } else  {
       id2<- sample(which(p_up>index[k,1] & p_up <=index[k,2], arr.ind = TRUE),length(na.omit(j[,k])))
       d_up[id2]<- na.omit(data[,2*nk+k])
       n_up[id2,]<- na.omit(cbind(data[,k],data[,nk+k]))
      }

     w[t,k]<- rbeta(1,length(na.omit(j[,k]))+alpha,m[t,k]+beta)
    }


   w[t,]<- w[t,]/max(w[t,]) #recale w

   dve<- d_up
   nve<- n_up

   numer<- sum(dve/((nve[,1]+nve[,2])/nve[,1]/nve[,2]+dve^2/2/(nve[,1]+nve[,2])+tau[t-1]^2))
   deno<- sum(1/((nve[,1]+nve[,2])/nve[,1]/nve[,2]+dve^2/2/(nve[,1]+nve[,2])+tau[t-1]^2))+1/b
   mu[t]<- rnorm(1,numer/deno,sqrt(1/deno))

   if (random==TRUE) {
     tau[t] <- tau[t-1] + runif(1,-0.1,0.1) # draw tau
     if(tau[t] >d ||tau[t] <c) {
       tau[t] <- tau[t-1] }
     u <- runif(1,0,1)
     logtau_t<-sum(-1/2*log((nve[,1]+nve[,2])/nve[,1]/nve[,2]+dve^2/2/(nve[,1]+nve[,2])+tau[t]^2))+sum
     logtau_t_1<-sum(-1/2*log((nve[,1]+nve[,2])/nve[,1]/nve[,2]+dve^2/2/(nve[,1]+nve[,2])+tau[t-1]^2))
     if(log(u) > (logtau_t-logtau_t_1))  {
       tau[t] <- tau[t-1]  }

    }  else  {tau[t]=0 }

  }

 results<-as.matrix(cbind(w,m,mu,tau))
 pick = seq(burn,M,thin)
 w=w[pick,]
 m=m[pick,]
 mu=mu[pick]
 tau=tau[pick]
 note<-""
 for (i in 1:nk) {
   note_tem <-  paste(names1[i], "and", names2[i], "are corresponding to p value interval: ",index[i,1]
   note<-paste(note,note_tem )
  }
 results1 = list(note,w=w,m=m,mu=mu,tau=tau)
 results2 = list(note,w=w,m=m,mu=mu)
 if(random==TRUE) {
   return( results1) }else  {return( results2) }
}
```

```r
# Get results from two MCMC chains with different sets of starting values and check convergence.
# If reaching convergence, output point estimate and credible interval
library(coda)
```

```
## Warning: package 'coda' was built under R version 3.3.3
```

```r
balm_output<-function(d_obs,n_obs,type=1,side=1,pn=1,random=TRUE,cpoint=c(0.05),M=15000,
                      burn=5000,thin=35,adjust=TRUE,mu_start=c(0,0.2), tau_start=c(0.2,0.1),
                      w_prior=c(0.1,0.1), mu_prior=c(0, 10000),tau_prior=c(0, 100)) {

  # calculate the modes of mu and tau as estimates of mu and tau
  mode.v<-function(v) {
    dens.y<-density(v)
    dens.y$x[order(dens.y$y,decreasing=T)][1]
    return(dens.y$x[order(dens.y$y,decreasing=T)][1])
   }

  # calcualte credible intervals
  emp.hpd<-function (x, alpha = 0.95)  {
    alpha <- min(alpha, 1 - alpha)
    n <- length(x)
    L.U <- round(n * alpha) # only need to try upto alpha
    x <- sort(x)
    e <- x[(n - L.U + 1):n] - x[1:L.U]
    m <- min(e)
    ind <- which(e == m)[1]
    return(c(x[ind], x[n - L.U + ind]))
   }

  result1<- balm(d_obs=d_obs,n_obs=n_obs,type=type,side=side,pn=pn,random=random,
                 cpoint=cpoint,M=M,burn=burn,thin=thin,adjust=adjust,
                 w_prior=w_prior,mu_prior=mu_prior,tau_prior=tau_prior,
                 mu_start=mu_start[1], tau_start=tau_start[1])
  result2<- balm(d_obs=d_obs,n_obs=n_obs,type=type,side=side,pn=pn,random=random,
                 cpoint=cpoint,M=M,burn=burn,thin=thin,adjust=adjust,
                 w_prior=w_prior,mu_prior=mu_prior,tau_prior=tau_prior,
                 mu_start=mu_start[2], tau_start=tau_start[2])
  # check convergence using Gelman and Rubin's convergence diagnostic
  mu1<-mcmc(result1$mu)
  mu2<-mcmc(result2$mu)
  listmu<-mcmc.list(list(mu1,mu2))

  if (random==TRUE) {
    tau1<-mcmc(result1$tau)
    tau2<-mcmc(result2$tau)
    listtau<-mcmc.list(list(tau1,tau2))

    if ((gelman.diag(listmu)$psrf[2]<1.1 & gelman.diag(listtau)$psrf[2]<1.1)==TRUE)  {
      # combine the results from two MCMC chains to obtain the estimates of mu and tau
      p.mu <- mode.v(c(result1$mu,result2$mu)) # the point estimate of mu
      ci.mu <- emp.hpd(c(result1$mu,result2$mu)) # the credible interval of mu
      p.tau <- mode.v(c(result1$tau,result2$tau)) # the point estimate of tau
      ci.tau <- emp.hpd(c(result1$tau,result2$tau)) # the credible interval of tau
      table <- matrix (c(p.mu,ci.mu, p.tau, ci.tau),nrow=2,ncol=3,byrow=TRUE)
```

```
      colnames(table)<- c("estimate", "CI.lower", "CI.upper")
      rownames(table)<- c("mu", "tau")
      out2<-list("The corrected estimates for the overall population effect size and between-study stand
      return(out2)
     } else  {
      return("MCMC chains do not converge. Please run balm_output again, change the staring values or s
     }
   } else  {
    if ((gelman.diag(listmu)$psrf[2]<1.1)==TRUE)  {
      # combine the results from two MCMC chains to obtain the estimates of mu and tau
      p.mu <- mode.v(c(result1$mu,result2$mu)) # the point estimate of mu
      ci.mu <- emp.hpd(c(result1$mu,result2$mu)) # the credible interval of mu
      table <- matrix (c(p.mu,ci.mu),nrow=1,ncol=3)
      colnames(table)<- c("estimate", "CI.lower", "CI.upper")
      rownames(table)<- c("mu")
      out1<-list("The corrected estimates for the overall population effect size are",table )
      return(out1)
     } else  {
      return("MCMC chains do not converge. Please run balm_output again or change the staring values.")
     }
   }
 }

#######EXAMPLE########
## Meta-analysis on gender differences in social persuasion studies by Becker (1986)
#########################
n1 <- c(68, 126, 159, 103, 90, 127, 98, 183, 59, 78, 72, 107, 166, 204, 30, 72, 96, 131, 68,
        399, 145, 138, 132, 139, 90, 74, 72, 30, 12, 44, 18, 35, 14)
n2 <- c(77, 126, 159, 104, 92, 126, 95, 176, 65, 41, 73, 107, 165, 205, 30, 72, 86, 130, 68,
        579, 146, 104, 80, 191, 76, 67, 40, 30, 8, 31, 29, 21, 14)
g <- c(0.54, 0.03, 0.09, 0.09, 0.04, -0.2, -0.11, -0.15, 0.17, 0.05, 0.2, 0.05, -0.3, 0.26,
       -0.18, 0.22, 0.36, 0.14, -0.13, 0.2, 0.07, 0.37, 0.12, 0.16, -0.09, 0.35, 0.35, 0.63,
       0.5, 0.19, 0.82, 0.18, 0.6)
#########################
####Please run both balm() and balm_output() functions
#########################
balm_output(d_obs=g,n_obs=cbind(n1,n2),type=1,side=2,random=TRUE,cpoint=c(0.05,0.5,0.95),
            M=15000,burn=5000,thin=35,adjust=TRUE,mu_start=c(0.0,0.2), tau_start=c(0.0,0.1))
```

```
## [[1]]
## [1] "The corrected estimates for the overall population effect size and between-study standard deviat
##
## [[2]]
##        estimate    CI.lower  CI.upper
## mu  -0.05535947 -0.28451501 0.2029037
## tau  0.17426579  0.07629642 0.4001606
```

```
#########################
## Pearson correlation
#########################

balm_r=function(r_obs,n_obs,type,random=TRUE,side=1,pn=1,cpoint=0.5,
                M=15000,burn=15000,thin=35,w_prior=c(0.1,0.1),
                mu_prior=c(0, 10000),tau_prior=c(0, 100),
```

```
              mu_start=0, tau_start=0.2) {
# r_obs reads in the observed Pearson correlation from the studies included in a meta-analysis.
# n_obs reads in the corresponding sample size of the studies.
# type specifies the suppression type of unpublished studies: 1 for suppression due to nonsignificant
#   or due to both nonsignificance and unexpected direction, and 2 for suppression due to unexpected
# side specifies the type of alternative hypothesis: 1 for two-sided test, 2 for r > 0,
#   and 3 for r < 0. cpoint=0.5 needs to be specified to distinguish positive
#   vs. negative results.
# pn specifies the expected direction: 1 for r > 0 as expected direction,
#   and 2 for r < 0 as expected direction
# cpoint specifies the cutoff value for p-value intervals.
# burn and thin specify the burn-in period and the thinning period respectively.
# M specifies the total of iterations of the Markov Chains including the burn-in and thinning periods
# adjust=T is to obtain unbiased values if the g values are read in.
# w_prior specifies a beta prior distribution for w.
# mu_prior specifies a normal prior distribution for mu.
# tau_prior specifies a uniform prior distribution for tau.
# mu_start specifies a starting value for mu.
# tau_start specifies a starting value for tau.

n<- n_obs
z_obs<-0.5*log((1+r_obs)/(1-r_obs)) # transfer Pearson correlation by Fisher z-transformation

nk<- length(cpoint)+1
index<- matrix(0,nrow=nk,ncol=2)
j<- matrix(NA,nrow=length(z_obs),ncol=nk)
index[1,]<- c(0,cpoint[1])

if (type==1) {
  if (side==1) {
    p<- 2*(1-pnorm(abs(z_obs),sd=sqrt(1/(n-3))))
   } else if (side==2) {
    p<-1-pnorm(z_obs,sd=sqrt(1/(n-3)))
   } else if (side==3) {
    p<- pnorm(z_obs,sd=sqrt(1/(n-3)))
   }
 } else if (type==2) {
  if (pn==1) {
    p<- 1-pnorm(z_obs,sd=sqrt(1/(n-3)))
   } else if (pn==2) {
    p<- pnorm(z_obs,sd=sqrt(1/(n-3)))
   }
 }

if (length(which(p>index[1,1]&p <=index[1,2], arr.ind = TRUE))==0) {
  j[,1]<- rep(NA,length(z_obs))
 }else {
  j[1:length(which(p>index[1,1]&p <=index[1,2], arr.ind = TRUE)),1]<- which(p>index[1,1]&p <=index[1,

if (nk>2) {
  for (k in 2:(nk-1)) {
    index[k,]<-c (cpoint[k-1],cpoint[k])
    if (length(which(p>index[k,1]&p <=index[k,2], arr.ind = TRUE))==0) {
```

```
      j[,k]<- rep(NA,length(d_obs))
     }else {
      j[1:length(which(p>index[k,1]&p <=index[k,2], arr.ind = TRUE)),k]<- which(p>index[k,1]&p <=index
   }
 }

index[nk,]<-c(cpoint[nk-1],1)
if (length(which(p>index[nk,1]&p <=index[nk,2], arr.ind = TRUE))==0) {
  j[,nk]<- rep(NA,length(d_obs))
 }else {
  j[1:length(which(p>index[nk,1]&p <=index[nk,2], arr.ind = TRUE)),nk]<- which(p>index[nk,1]&p <=index

a<- length(z_obs)
data<- matrix(NA,nrow=a,ncol=2*nk)
for (k in 1:nk) {
  data[,k]<- c(n[na.omit(j[,k])],rep(NA,a-length(na.omit(j[,k]))))
  data[,nk+k]<- c(z_obs[na.omit(j[,k])],rep(NA,a-length(na.omit(j[,k]))))
 }

w<-matrix(0,nrow=M,ncol=nk)
prefix1 <- "w"
suffix <- seq(1:nk)
names1 <- paste(prefix1, suffix, sep = "")
colnames(w)<-names1
m<-matrix(0,nrow=M,ncol=nk)
prefix2 <- "m"
names2 <- paste(prefix2, suffix, sep = "")
colnames(m)<-names2

alpha<- w_prior[1]
beta<- w_prior[2]
a<- mu_prior[1] #mu prior
b<- mu_prior[2] #mu prior
c<- tau_prior[1] # tau prior
d<- tau_prior[2] # tau prior
w[1,]<- rep(0.5,nk) #w initial value
mu<- c()
mu[1]<- mu_start
tau<- c()
tau[1]<- tau_start
w[1,]<- w[1,]/max(w[1,])

for(t in 2:M) {

  logi<- rep(0,10^3)

  k<- which(w[t-1,]==1)
  if (length(k)>1) {k=1 }
  m[t,k]<- 0

  n_mis<- n_obs[c(sample(c(1:length(z_obs)), 10^3-length(z_obs), replace = T))]
  n_up<- c(n_obs,n_mis)
  tz<- rnorm(10^3,mean=mu[t-1], sd=tau[t-1])
```

```
v<- 1/(n_up-3)
z_up<-rnorm(10^3,mean=tz, sd=sqrt(v))

if (type==1) {
  if (side==1) {
    p_up<- 2*(1-pnorm(abs(z_up),sd=sqrt(1/(n_up-3))))
   } else if (side==2) {
    p_up<- 1-pnorm(abs(z_up),sd=sqrt(1/(n_up-3))) } else if (side==3) {
      p_up<- pnorm(abs(z_up),sd=sqrt(1/(n_up-3))) }
 }else if (type==2) {
  if (pn==1) {
    p_up<- 1-pnorm(abs(z_up),sd=sqrt(1/(n_up-3)))
   } else if (pn==2) {
    p_up<- pnorm(abs(z_up),sd=sqrt(1/(n_up-3)))
   }
 }

logi[which(p_up > index[k,1] & p_up <= index[k,2], arr.ind = TRUE)]<- 1
logi<- order(logi, decreasing=T)
et<- logi[length(na.omit(j[,k]))]
if(length(et)==0) {et=0 }
m2<- et-length(z_obs)
if (m2<0) {m2=0 }

n_up<- n_up[1:(m2+length(z_obs))]
z_up<- z_up[1:(m2+length(z_obs))]
p_up<- p_up[1:(m2+length(z_obs))]


for (k in 1:nk) {
  m[t,k]<- length(which(p_up>index[k,1] & p_up <=index[k,2], arr.ind = TRUE))-length(na.omit(j[,k]))
  if(m[t,k]<0) {
    m[t,k]= 0
    id1<- which(p_up>index[k,1] & p_up <=index[k,2], arr.ind = TRUE)
    nid<- sample(1:length(na.omit(j[,k])),length(id1))
    z_up[id1]<- na.omit(data[,nk+k])[nid]
    n_up[id1]<- na.omit(data[,k])[nid]
   } else  {
    id2<- sample(which(p_up>index[k,1] & p_up <=index[k,2], arr.ind = TRUE),length(na.omit(j[,k])))
    z_up[id2]<- na.omit(data[,nk+k])
    n_up[id2]<- na.omit(data[,k])
   }

  w[t,k]<- rbeta(1,length(na.omit(j[,k]))+alpha,m[t,k]+beta)
 }


w[t,]<- w[t,]/max(w[t,]) #recale w

zve<- z_up
nve<- n_up

numer<- sum(zve/(1/(nve-3)+tau[t-1]^2))
```

```
    deno<- sum(1/(1/(nve-3)+tau[t-1]^2))+1/b
    mu[t]<- rnorm(1,numer/deno,sqrt(1/deno))

    if (random==TRUE) {
      tau[t] <- tau[t-1] + runif(1,-0.1,0.1) # draw tau
      if(tau[t] >d ||tau[t] <c) {
        tau[t] <- tau[t-1] }
      u <- runif(1,0,1)
      logtau_t<-sum(-1/2*log(1/(nve-3)+tau[t]^2))+sum(-1/2*(zve-mu[t])^2/(1/(nve-3)+tau[t]^2))
      logtau_t_1<-sum(-1/2*log(1/(nve-3)+tau[t-1]^2))+sum(-1/2*(zve-mu[t])^2/(1/(nve-3)+tau[t-1]^2))
      if(log(u) > (logtau_t-logtau_t_1))  {
        tau[t] <- tau[t-1]  }

     }  else  {tau[t]=0 }

   }

  results<-as.matrix(cbind(w,m,mu,tau))
  pick = seq(burn,M,thin)
  w=w[pick,]
  m=m[pick,]
  mu=mu[pick]
  tau=tau[pick]
  note<-""
  for (i in 1:nk) {
    note_tem <-  paste(names1[i], "and", names2[i], "are corresponding to p value interval: ",index[i,1]
    note<-paste(note,note_tem )
   }
  results1 = list(note,w=w,m=m,mu=mu,tau=tau)
  results2 = list(note,w=w,m=m,mu=mu)
  if(random==TRUE) {
    return( results1) }else  {return( results2) }
 }

# Get results from two MCMC chains with different sets of starting values and check convergence.
# If reaching convergence, output point estimate and credible interval
library(coda)
balmr_output<-function(r_obs,n_obs,type=1,side=2,pn=1,random=TRUE,cpoint=c(0.05),M=15000,
                       burn=5000,thin=35,mu_start=c(0,0.1), tau_start=c(0.1,0.05),
                       w_prior=c(0.1,0.1),mu_prior=c(0, 10000),tau_prior=c(0, 100)) {

  # calculate the modes of mu and tau as estimates of mu and tau
  mode.v<-function(v) {
    dens.y<-density(v)
    dens.y$x[order(dens.y$y,decreasing=T)][1]
    return(dens.y$x[order(dens.y$y,decreasing=T)][1])
   }

  # calcualte credible intervals
  emp.hpd<-function (x, alpha = 0.95)  {
    alpha <- min(alpha, 1 - alpha)
    n <- length(x)
    L.U <- round(n * alpha) # only need to try upto alpha
```

```
  x <- sort(x)
  e <- x[(n - L.U + 1):n] - x[1:L.U]
  m <- min(e)
  ind <- which(e == m)[1]
  return(c(x[ind], x[n - L.U + ind]))
 }

result1<- balm_r(r_obs=r_obs,n_obs=n_obs,type=type,side=side,pn=pn,random=random,
                 cpoint=cpoint,M=M,burn=burn,thin=thin,
                 w_prior=w_prior,mu_prior=mu_prior,tau_prior=tau_prior,
                 mu_start=mu_start[1], tau_start=tau_start[1]) #BALM is used to correct for negative
result2<- balm_r(r_obs=r_obs,n_obs=n_obs,type=type,side=side,pn=pn,random=random,
                 cpoint=cpoint,M=M,burn=burn,thin=thin,
                 w_prior=w_prior,mu_prior=mu_prior,tau_prior=tau_prior,
                 mu_start=mu_start[2], tau_start=tau_start[2])
# check convergence using Gelman and Rubin's convergence diagnostic
mu1<-mcmc(result1$mu)
mu2<-mcmc(result2$mu)
listmu<-mcmc.list(list(mu1,mu2))

if (random==TRUE) {
  tau1<-mcmc(result1$tau)
  tau2<-mcmc(result2$tau)
  listtau<-mcmc.list(list(tau1,tau2))

  if ((gelman.diag(listmu)$psrf[2]<1.1 & gelman.diag(listtau)$psrf[2]<1.1)==TRUE)  {
    # combine the results from two MCMC chains to obtain the estimates of mu and tau
    p.mu <- mode.v(c(result1$mu,result2$mu)) # the point estimate of mu
    ci.mu <- emp.hpd(c(result1$mu,result2$mu)) # the credible interval of mu
    p.tau <- mode.v(c(result1$tau,result2$tau)) # the point estimate of tau
    ci.tau <- emp.hpd(c(result1$tau,result2$tau)) # the credible interval of tau
    table <- matrix (c(p.mu,ci.mu, p.tau, ci.tau),nrow=2,ncol=3)
    colnames(table)<- c("estimate", "CI.lower", "CI.upper")
    rownames(table)<- c("mu", "tau")
    out2<-list("The corrected estimates for the overall population effect size (Fisher's z) and betwe
    return(out2)
   } else  {
    return("MCMC chains do not converge. Please change staring values or simplify the model (e.g., fi
   }
 } else  {
  if ((gelman.diag(listmu)$psrf[2]<1.1)==TRUE)  {
    # combine the results from two MCMC chains to obtain the estimates of mu and tau
    p.mu <- mode.v(c(result1$mu,result2$mu)) # the point estimate of mu
    ci.mu <- emp.hpd(c(result1$mu,result2$mu)) # the credible interval of mu
    table <- matrix (c(p.mu,ci.mu),nrow=1,ncol=3)
    colnames(table)<- c("estimate", "CI.lower", "CI.upper")
    rownames(table)<- c("mu")
    out1<-list("The corrected estimates for the overall population effect size (Fisher's z) are",table
    return(out1)
   } else  {
    return("MCMC chains do not converge. Please change staring values.")
   }
 }
```

```
 }

#######EXAMPLE########
## Correlation between employment interview assessments and job performance by McDaniel et al. (1994)
#########################
library(metafor)
```

```
## Warning: package 'metafor' was built under R version 3.3.3
```

```
## Loading required package: Matrix
```

```
## Loading 'metafor' package (version 2.0-0). For an overview
## and introduction to the package please type: help(metafor).
```

```
data <- dat.mcdaniel1994 # use a data set in R package metafor

#########################
####Please run both balm_r() and balmr_output() functions
#########################
balmr_output(r_obs=data$ri,n_obs=data$ni,type=1,side=2,random=TRUE,cpoint=c(0.05),
             M=15000,burn=5000,thin=35,mu_start=c(0,0.2), tau_start=c(0.2,0.1))
```

```
## [[1]]
## [1] "The corrected estimates for the overall population effect size (Fisher's z) and between-study s
##
## [[2]]
##      estimate  CI.lower  CI.upper
## mu  0.2155240 0.2571679 0.1069934
## tau 0.1807751 0.1321355 0.1665747
```