R codes for “Competing risks regression models with covariates-adjusted censoring weight under the generalized case-cohort design”

By Soyoung Kim and Yayun Xu

There are three R functions including weight functions, beta estimations, and variance estimations for competing risks regression under generalized case-cohort studies.

```
###weight functions
find.weightcc = function(gamma, disease , v, delta , choice = 1,eta) {
  weightcc = c()
  if (choice == 1){#weight function for sigle case-cohort study
    for (i in unique(v)) {
      num = sum(gamma[v == i] * (1 - (disease[v == i] == 1)))
      deno = sum((1 - (disease[v == i] == 1)))
      deno = deno + (deno== 0)
      alpha.est = num / deno
      alpha.est = alpha.est + (alpha.est == 0)
      denoq = sum(gamma[v==i] == 0 & disease[v==i] == 1)
      numq = sum(eta[v==i] == 1)

      if (denoq == 0){ q.est = 1
      }else{
      q.est = numq / denoq }
      weightcc[v==i] = (disease[v==i] == 1) * gamma[v==i]
      +(1 - (disease[v==i] == 1)) * gamma[v==i] / alpha.est +(eta[v==i] == 1) / q.est
    }
  }else if (choice == 2){
    for (i in unique(v)) {# optimal weight function for single case-cohort study
      num = sum(gamma[v == i] * (1 - (disease[v == i] == 1)))
      deno = sum((1 - (disease[v == i] == 1)))
      deno = deno + (deno== 0)
      alpha.est = num / deno
      alpha.est = alpha.est + (alpha.est == 0)
      denoq = sum(gamma[v==i] == 0 & disease[v==i] == 1)
      numq = sum(eta[v==i] == 1)

      if (denoq == 0){ q.est = 1
      }else{
      q.est = numq / denoq }
      weightcc[v==i] = (1 - (disease[v==i] == 1)) * gamma[v==i] / alpha.est +
        (disease[v==i]==1)*(gamma[v==i]+ (1-gamma[v==i])*(eta[v==i] == 1))/
        (alpha.est+ (1-alpha.est)*q.est)
    }
  }else if (choice == 3) { # efficient weight function for multiple case-cohort studies
```

```
    for (i in unique(v)) {
      num = sum(gamma[v == i] * (1 - delta[v == i]))
      deno = sum(1 - delta[v == i])
      deno = deno + (deno == 0)
      alpha.est = num / deno
      alpha.est = alpha.est + (alpha.est == 0)

      denoq1 = sum(gamma[v==i] == 0 &disease[v==i] == 1)
      numq1 = sum(eta[v==i] == 1)
      if (denoq1 == 0){ q1.est = 1
      }else{
      q1.est = numq1 / denoq1 }

      denoq2 = sum(gamma[v==i] == 0 &disease[v==i] == 2)
      numq2 = sum(eta[v==i] == 2)
      if (denoq2 == 0){ q2.est = 1
      }else{
      q2.est = numq2 / denoq2 }
      weightcc[v==i] = (disease[v==i] != 0) * gamma[v==i] + (disease[v==i] == 0)
       * gamma[v==i] / alpha.est +(eta[v==i] == 1) / q1.est +(eta[v==i] == 2) / q2.est
    }
  }else if (choice ==5){ #optimal weight function for multiple case-cohort studies
    for (i in unique(v)){
      deno = sum(disease[v==i] == 0)
      num = sum(gamma[v==i] == 1 &disease[v==i] == 0)
      alpha.est = num / deno
      denoq1 = sum(gamma[v==i] == 0 & disease[v==i] == 1)
      numq1 = sum(eta[v==i] == 1)
      if (denoq1 == 0){ q1.est = 1
      }else{
      q1.est = numq1 / denoq1 }
      denoq2 = sum(gamma[v==i] == 0 & disease[v==i] == 2)
      numq2 = sum(eta[v==i] == 2)
      if (denoq2 == 0){ q2.est = 1
      }else{
        q2.est = numq2 / denoq2 }

        weightcc[v==i] = (disease[v==i] ==1)
        * gamma[v==i]*(1/(alpha.est+(1-alpha.est)*q1.est)) +
        (disease[v==i] ==2) * gamma[v==i]*(1/(alpha.est+(1-alpha.est)*q2.est)) +
        (disease[v==i] == 0) * gamma[v==i] / alpha.est +
        (eta[v==i] == 1) * (1/(alpha.est+(1-alpha.est)*q1.est)) +
        (eta[v==i] == 2) * (1/(alpha.est+(1-alpha.est)*q2.est))
    }
  }

  return(weightcc)

}

###beta estimation function
# choice: weight functions for case-cohort studies
#            1: weight function for single case-cohort study
#            2: optimal weight function for single case-cohort study
#            3: weight function for multiple case-cohort studies
#            5: optimal weight function for multiple case-cohort studies
# choice.c: weight functions for censoring distribution
#          1: covariate-adjusted weights
#           2: covariate-unadjusted weights
# data
# time: observed time
# delta: failarue indicator
# disease: cause types (1: cause 1, 2: cause 2, 0: censored)
# z: covariate
# gamma: subcohort indicator
# v: stratum variable
# eta: case indicator
beta.est = function(data, beta = 0, delta0 = 3, choice = 1,choice.c=1) {

  n = nrow(data)
  time = data[, 2]
  delta = data[, 3]
  disease = data[, 4]
  z = data$Z
  gamma = data$gamma
  v = data$v
  eta = data$eta
  npop = n
  fail = time[!(disease == 0)]
  L = length(fail)
  gg1 = (matrix(rep(time, L), npop, L) <
  matrix(rep(t(fail), each = npop), npop, L)) * 1
  gg1[disease != 1, ] = 0
  gg2 = matrix(1, npop, L)

  for (i in 1:npop) {
    if (delta[i] == 0) {
      for (j in 1:L) {
        if (time[i] < fail[j])
          gg2[i, j] = 0
      }
    }
  }
  Yr = gg2 - gg1
  weight.cc = find.weightcc(gamma, disease ,v, delta , choice=choice , eta)
  Gcweight = list()
  faili <- time[disease == 1]
  Li <- length(faili)
  censor<-1-delta
  if (choice.c == 1){ ##COX estimates for censoring distribution
    for (i in unique(v)) {
      i.name = paste(i)
      data.cc<-data.frame(time[v == i],censor[v == i],z[v == i],weight.cc[v == i] )
      data.cc1<-data.cc[data.cc$weight.cc>0,]
      colnames(data.cc1) <- c("time", "censor", "z", "weight.cc")
      ss <- Surv(data.cc1$time, data.cc1$censor)
      fit.cox<-survival::coxph(ss ~ z, weight= weight.cc, data=data.cc1)
      Gc.0 = survfit(fit.cox,newdata=data.frame(z=0))
      Gc.1 = survfit(fit.cox,newdata=data.frame(z=1))

      kmest.0 = stepfun(Gc.0$time, c(1,Gc.0$surv))
      kmest.1 = stepfun(Gc.1$time, c(1,Gc.1$surv))

      gg3 = (matrix(rep(time,Li),npop,Li) >= matrix(rep(t(faili),each=npop),npop,Li))
      foo.0 = matrix(rep(t(kmest.0(faili)),each=npop),npop,Li)
      /(matrix(rep(kmest.0(time),Li),npop,Li))
      foo.1 = matrix(rep(t(kmest.1(faili)),each=npop),npop,Li)
      /(matrix(rep(kmest.1(time),Li),npop,Li))
      foo=foo.0*(z==0) +foo.1*(z==1)
      foo[gg3] = 1
      Gcweight[[i.name]] = foo
    }
  }else {  ##KM estimates FOR censoring distribution
    for (i in unique(v)) {
      i.name = paste(i)
      Gc = survfit(Surv(time[v == i], 1 - delta[v == i]) ~ 1)
      kmest = stepfun(Gc$time, c(1, Gc$surv))
      gg3 = (matrix(rep(time, Li), npop, Li) >=
      matrix(rep(t(faili), each = npop), npop, Li))
      foo = matrix(rep(t(kmest(faili)), each = npop), npop, Li)
       / (matrix(rep(kmest(time), Li), npop, Li))
      foo[gg3] = 1
      Gcweight[[i.name]] = foo
    }
  }
  wY = Yr
  dNr = (matrix(rep(time, L), npop, L) ==
   matrix(rep(t(fail), each = npop), npop, L)) *1
  dNr[disease != 1,] = 0
  wdN = dNr
  p = ifelse(is.null(dim(z)[2]), 1, dim(z)[2])
  fail1 <-time[disease == 1]
  L1 <-length(fail1)
  fail = data[disease != 0 , c(4, 2)]
  indi1 = which(fail[, 1] == 1)
  YWeight1 = wY[, indi1]
  wdN1 = wdN[, indi1]

  p = ifelse(is.null(dim(z)[2]), 1, dim(z)[2])

  step = 0
  while (delta0 > 10 ^ -6 & (step <= 20)) {
    step <- step + 1
    z = as.vector(z)
    beta = as.vector(beta)
    Sl1overSl0hat = Sl2overSl0hat =  list()
    for (i in unique(v)) {
      i.name <- paste(i)
      dis1 = (disease[disease == 1]) * 1
      v1 = (v[disease == 1]) * 1
      index2 = which(dis1 == 1 &v1 == i)

      n_l1 = length(index2)
      index3 = which(disease[v == i][disease[v == i] == 1] == 1)
      nl = sum(v == i)
      expz  <- exp(z[v == i] * beta)
      zexpz <- matrix(rep(expz, p), nl, p) * z[v == i]  # nl*p
      temp0 <-t(expz) %*% (weight.cc[v == i]
      * YWeight1[v == i,]*Gcweight[[i.name]][v == i, ])
      Sl0 <- temp0 + (temp0 == 0)
      Sl1 <-t(zexpz) %*% (weight.cc[v == i]
      * YWeight1[v == i  ,]*Gcweight[[i.name]][v == i, ])

      z2expz = matrix(rep(expz, p), nl, p) * z[v == i] * z[v == i]
```

```
      Sl2 = t(z2expz) %*% (weight.cc[v == i]
      * YWeight1[v == i  ,]*Gcweight[[i.name]][v == i, ])
      Sl1overSl0hat[[i.name]] <- Sl1 / Sl0
      Sl2overSl0hat[[i.name]] = Sl2 / Sl0
    }
    SU = 0
    SI = 0
    for (i in unique(v)) {
      i.name = paste(i)
      dis1 = (disease[disease == 1]) * 1
      v1 = (v[disease == 1]) * 1
      index2 = which(dis1 == 1 &v1 == i)

      n_l1 = length(index2)
      n_l = sum(v == i)
      U = (do.call(cbind, replicate(n_l1, z[v == i], simplify = F))
      - do.call(rbind, replicate(n_l, Sl1overSl0hat[[i.name]][index2], simplify = F)))
      *wdN1[v == i, index2]*Gcweight[[i.name]][v == i, index2]*weight.cc[v == i]  # n_l x n_l1
      SU = SU + sum(U)
      ipart1 = do.call(rbind, replicate(n_l, Sl2overSl0hat[[i.name]][index2],
      simplify = F)) *wdN1[v == i, index2]*Gcweight[[i.name]][v == i, index2]
      *weight.cc[v == i]
      IpartIhat = sum(ipart1)
      E2 = Sl1overSl0hat[[i.name]] * Sl1overSl0hat[[i.name]]
      ipart2 = do.call(rbind, replicate(n_l, E2[index2], simplify = F))
       *wdN1[v == i, index2]*Gcweight[[i.name]][v == i, index2]*weight.cc[v == i]
      IpartIIhat = sum(ipart2)
      SI <- SI + (IpartIhat - IpartIIhat)
    }
    tempU = SU
    tempI = SI
    iI <- solve(tempI)
    beta <- beta + iI %*% tempU
    delta0 <- max(abs(iI %*% tempU))
  }
  return(beta)
}

## variance function
find.var = function(data, beta, choice, choice.c){

  gamma = data[,6]
  n = nrow(data)

  time = data[,2]
  delta = data[,3]
  disease = data[,4]
  z = data$Z
  eta = data$eta
  v = data$v

  Nrt = (do.call(cbind, replicate(n, time, simplify = FALSE))
  < do.call(rbind, replicate(n, time, simplify = FALSE))) *1
  Nrt[disease != 1,] = 0
  rt = matrix(1, n, n)

  for (i in 1:n) {
    if (delta[i] == 0) {
```

```
    for (j in 1:n) {
      if (time[i] < time[j])
        rt[i, j] = 0
    }
  }
}
Yrt = rt - Nrt
phit = find.weightcc(gamma, disease ,  v, delta , choice ,eta)
weight.cc = phit
Gcweight = list()
fit.cox = list()

censor<-1-delta
if (choice.c == 1){        ##COX  estimate for censoring time
  for (i in unique(v)) {
    i.name = paste(i)
    n_l=sum(v==i)
    data.cc<-data.frame(time,censor,z,v,weight.cc )
    data.cc1<-data.cc[data.cc$weight.cc>0,]
    nc<-dim(data.cc1)[1]
    nc0<-sum(data.cc1$v==0)

    colnames(data.cc1) <- c("time", "censor", "z", "v","weight.cc")
    ss <- Surv(data.cc1$time, data.cc1$censor)
    fit.cox<-survival::coxph(ss ~ z+strata(v),
    weight= weight.cc, data=data.cc1)

    Gc.0 = survfit(fit.cox,newdata=data.frame(z=0))
    Gc.1 = survfit(fit.cox,newdata=data.frame(z=1))
    if (i==0){
      kmest.0 = stepfun(Gc.0$time[1:nc0], c(1,Gc.0$surv[1:nc0]))
      kmest.1 = stepfun(Gc.1$time[1:nc0], c(1,Gc.1$surv[1:nc0]))
    }else{
      kmest.0 = stepfun(Gc.0$time[(nc0+1):nc], c(1,Gc.0$surv[(nc0+1):nc]))
      kmest.1 = stepfun(Gc.1$time[(nc0+1):nc], c(1,Gc.1$surv[(nc0+1):nc]))
    }
    gg3 = (matrix(rep(time[v == i],n),n_l,n)
     >= matrix(rep(t(time[v == i]),each=n),n_l,n))
    d0 =kmest.0(time[v == i]) +(kmest.0(time)==0)
    d1 =kmest.1(time[v == i]) +(kmest.1(time)==0)
    foo.0 = matrix(rep(t(kmest.0(time[v == i])),each=n),n_l,n)
    /(matrix(rep(d0,n),n_l,n))
    foo.1 = matrix(rep(t(kmest.1(time[v == i])),each=n),n_l,n)
    /(matrix(rep(d1,n),n_l,n))
    foo=foo.0*(z[v == i]==0) +foo.1*(z[v == i]==1)
    foo[gg3] = 1
    Gcweight[[i.name]] = foo
  }
}else {  ##KM EST FOR censoring distribution

  for (i in unique(v)) {
    n_l=sum(v==i)
    i.name = paste(i)
    Gc = survfit(Surv(time[v == i], 1 - delta[v == i]) ~ 1)
    kmest = stepfun(Gc$time, c(1, Gc$surv))
    gg3 = (matrix(rep(time[v == i], n), n_l, n)
    >= matrix(rep(t(time[v == i]), each = n), n_l, n))
    fff  = (matrix(rep(kmest(time[v == i]), n), n_l, n))
    foo = matrix(rep(t(kmest(time[v == i])), each = n), n_l, n) / fff
    foo[gg3] = 1
    Gcweight[[i.name]] = foo
  }
}

wY.all = Yrt

dNr.all = (do.call(cbind, replicate(n, time, simplify = FALSE))
 == do.call(rbind, replicate(n, time, simplify = FALSE))) *1
dNr.all[disease != 1, ] = 0
wdN.all = dNr.all

p = 1
SI = 0
Sl0 = list()
Ec = list()

for (i in unique(v)) {
  i.name <- paste(i)
  dis1 = (disease[disease == 1]) * 1
  v1 = (v[disease == 1]) * 1
  nl = sum(v == i)
  expz  <- exp(z[v == i] * beta)
  zexpz <- expz * z [v == i]
  temp0  <-colSums(expz * (phit[v == i] * wY.all[v == i, ] * Gcweight[[i.name]]))
  Sl0[[i.name]] <- temp0 + (temp0 == 0)
  Sl1 <-colSums (zexpz * (phit[v == i] * wY.all[v == i  , ] * Gcweight[[i.name]]))
  z2expz = expz * z[v == i] * z[v == i]
  Sl2 = colSums(z2expz  * (phit[v == i] * wY.all[v == i  , ] * Gcweight[[i.name]]))
  Sl1overSl0hat <- Sl1 / Sl0[[i.name]]
  Sl2overSl0hat = Sl2 / Sl0[[i.name]]
  Ec[[i.name]] = Sl1overSl0hat

  ipart1 = do.call(rbind, replicate(nl, Sl2overSl0hat, simplify = F))
   *wdN.all[v == i,] * Gcweight[[i.name]]*phit[v == i]
  IpartIhat = sum(ipart1)

  E2 = Sl1overSl0hat * Sl1overSl0hat
  ipart2 = do.call(rbind, replicate(nl, E2, simplify = F))
  *wdN.all[v == i, ] * Gcweight[[i.name]]*phit[v == i]
  IpartIIhat = sum(ipart2)

  SI <- SI + (IpartIhat - IpartIIhat)
}
tempI = SI
Yt = (do.call(cbind,replicate(n,time,simplify = FALSE))
>= do.call(rbind,replicate(n,time,simplify = FALSE)))*1
Nct = (do.call(cbind,replicate(n,time,simplify = FALSE))
<= do.call(rbind,replicate(n,time,simplify = FALSE)))*1
Nct[disease!=0, ] = 0

var.1 = 0
expz = exp(z * beta)
for (i in unique(v)) {
  i.name = paste(i)
  nl = sum(v == i)
  dlamb10t = colSums(wdN.all[v == i,] * Gcweight[[i.name]] *phit[v == i] )
```

```
/ S10[[i.name]]
zminEc = (do.call(cbind, replicate(n, z[v == i], simplify = FALSE)))
 - (do.call(rbind, replicate(sum(v ==i), Ec[[i.name]], simplify = FALSE)))

wdM = wdN.all[v == i,] * Gcweight[[i.name]] - wY.all[v == i,]
*Gcweight[[i.name]] * expz[v == i] * (do.call(rbind, replicate(sum(v ==i),
 dlamb10t, simplify = FALSE)))
eta11 = rowSums(zminEc * wdM)
eta21 = c()
if (choice.c == 1){
  WeightedY<-phit[v==i]*Yt[v==i,]
  gamma.hat<-fit.cox$coef
  expg<-exp(z[v==i]*gamma.hat)
  zexpg<-z[v==i]*expg
  s0c<- colSums( WeightedY *expg)
  s0c<- s0c + (s0c==0)
  s1c<- colSums( WeightedY *zexpg)
  ecox <-s1c/s0c
  S1overS02hat<- ecox/s0c
  zc_min_ec<- (do.call(cbind,replicate(n,z[v==i],simplify = FALSE)))
   - (do.call(rbind,replicate(nl,ecox,simplify = FALSE)))
  iIc_mat<-vcov(fit.cox)[1]
  censor.w<- (disease[v==i]==0)*weight.cc[v==i]
  ht= t(apply((1-Yt[v==i,])*zc_min_ec*(do.call(rbind,replicate
  (nl,censor.w/s0c,simplify = FALSE))),1,cumsum))*expg
  dNc = (do.call(cbind,replicate(n,time,simplify = FALSE))
  == do.call(rbind,replicate(n,time,simplify = FALSE)))*1
  dNc[disease != 0,] = 0
  dlamb10c = colSums(dNc[v==i,]*phit[v==i])/s0c
  dMc = dNc[v==i,] - Yt[v==i,]*expg*
  do.call(rbind,replicate(nl,dlamb10c,simplify=FALSE)))
  zminEc.wdM<-zminEc*wdM
  intt = (1-Yt[v==i,])
  qi11<-sum((1-Yt[v==i,])*ht*iIc_mat*zminEc.wdM)/nl
  qi21<-colSums(intt*zminEc.wdM *expg)
  qi<-matrix(0,nl,n)
  for (j in 1:n){
    u<-time[j]
    ind<-1*(u<=time)
    locu<-which(time==u)
    qi2<- sum(qi21*ind) /s0c[locu]
    for (k in 1:nl){
      qi1<-  zc_min_ec[k,j]*qi11
      qi[k,j]<- -qi1-qi2
    }
  }
  eta21<- rowSums(qi*dMc)
}else{
  for (l in 1:sum(v == i)) {
    yt= (rep(time[v == i][l],n) >= time)
    nc = (rep(time[v == i][l], n) <= time) * (disease[v == i][l] == 0)
    Y.l = colSums(Yt[v==i,])
    Y.l[Y.l ==0 ] = 1
    lambc = colSums(Nct[v==i,]) / Y.l
    wct = (nc - yt * lambc) / Y.l
    syt = (rep(time[v == i][l], nl) >= time[v == i])
    snc = (rep(time[v == i][l], nl) <= time[v == i]) * (disease[v ==i][l] == 0)
    wlict = (snc - syt * lambc[v == i]) / Y.l[v==i]
    deltaw = do.call(cbind, replicate(n, wlict, simplify = F)) -
    do.call(rbind, replicate(nl, wct, simplify = F))
    ind = do.call(cbind, replicate(n, time[v == i], simplify = F))
    < do.call(rbind, replicate(sum(v == i), time, simplify = F))
    eta21 =   c(eta21, sum(deltaw * zminEc * ind * wdM))
  }
}

eta.comb = (eta11 + eta21)
eta.comb2 = eta.comb* eta.comb
qest1 = sum(eta[v==i]==1)/sum(disease[v==i]==1& gamma[v==i]==0)
alphaest = sum(gamma[v == i] ) / nl
V1 = sum( eta.comb2*gamma[v==i])/alphaest
if (choice == 1){
  Q = wY.all[v == i,]* Gcweight[[i.name]]* zminEc * expz[v == i]
  if  (choice.c ==1 ){
    eta31 = rowSums((1-delta[v == i] *(disease[v == i]  ==1))
    *Q*do.call(rbind,replicate(sum(v == i),dlamb10t,simplify = FALSE)))
    eta32 = (1-delta[v == i] *(disease[v == i]  ==1))*eta21
    eta3<- eta31-eta32
  }else{
    eta3 = rowSums((1-delta[v == i] *(disease[v == i]  ==1))
    *Q*do.call(rbind,replicate(sum(v == i),dlamb10t,simplify = FALSE)))
  }
  V31 = sum(eta.comb* eta.comb*(eta[v==i]==1))/sum(eta[v==i]==1)
  V32 = (sum(eta.comb *(eta[v==i]==1))/sum(eta[v==i]==1))^2
  V3 =  (1-alphaest) * (1-qest1)/qest1*(V31-V32)*
  sum(disease[v==i]==1&gamma[v==i]==0)
  V2 = sum(eta3*eta3*gamma[v==i]/alphaest)*(1-alphaest)/alphaest
  var.1 = var.1 + (V1+V2+V3)
}else if (choice ==3){ ##efficient weight
  qest2 = sum(eta[v==i]==2) /sum(disease[v==i]==2 & gamma[v==i]==0)
  Q = wY.all[v == i,]* Gcweight[[i.name]]* zminEc * expz[v == i]
  if  (choice.c ==1 ){
    eta31 = rowSums((1-delta[v == i] )*Q*do.call(rbind,replicate(sum(v == i),
    dlamb10t,simplify = FALSE)))
    eta32 = (1-delta[v == i])*eta21
    eta3<- eta31-eta32
  }else{
    eta3 = rowSums((1 - delta[v == i]) * Q *do.call(rbind, replicate(sum(v == i),
    dlamb10t, simplify = FALSE)))
  }
  V31 = sum(eta.comb* eta.comb*(eta[v==i]==1))/sum(eta[v==i]==1)
  V32 = (sum(eta.comb *(eta[v==i]==1))/sum(eta[v==i]==1))^2
  V3 =  (1-alphaest) * (1-qest1)/qest1*(V31-V32)*sum(disease[v==i]==1
  &gamma[v==i]==0)
  V41 = sum(eta.comb* eta.comb*(eta[v==i]==2))/sum(eta[v==i]==2)
  V42 = (sum(eta.comb *(eta[v==i]==2))/sum(eta[v==i]==2))^2
  V4 =  (1-alphaest) * (1-qest2)/qest2*sum(disease[v==i]==2&gamma[v==i]==0)
  *(V41 - V42)

  V2 = sum(eta3*eta3*gamma[v==i]/alphaest)*(1-alphaest)/alphaest

  var.1 = var.1 +(V1+V2+V3+V4)
}else if (choice ==5){
  qest2 = sum(eta[v==i]==2) /sum(disease[v==i]==2 & gamma[v==i]==0)
  np1= sum(eta[v==i]==1) + sum(gamma[v==i]==1& disease[v==i]==1)
  np2 = sum(eta[v==i]==2) + sum(gamma[v==i]==1 & disease[v==i] ==2)
```

```
      Q = wY.all[v == i,]* Gcweight[[i.name]]* zminEc * expz[v == i]
      if  (choice.c ==1 ){
        eta31 = rowSums((1-delta[v == i]  )*Q*do.call(rbind,replicate(
        sum(v == i),dlamb10t,simplify = FALSE)))
        eta32 = (1-delta[v == i])*eta21
        eta3<- eta31-eta32
      }else{
        eta3 = rowSums((1-delta[v == i]  )*Q*do.call(rbind,replicate(
        sum(v == i),dlamb10t,simplify = FALSE)))
      }
      V331 = sum(eta.comb*eta.comb*(eta[v==i]==1 | (gamma[v==i]==1 &
      disease[v==i]==1)))/ np1
      V332 = sum(eta.comb*(eta[v==i]==1 | (gamma[v==i]==1 & disease[v==i]==1)))/ np1
      V33 = (1/(alphaest+(1-alphaest)*qest1) - 1) *(V331 - V332^2)
      *sum(disease[v==i]==1&gamma[v==i]==0)

      V441 = sum(eta.comb*eta.comb*(eta[v==i]==2 | (gamma[v==i]==1
      & disease[v==i]==2)))/ np2
      V442 = sum(eta.comb*(eta[v==i]==2 | (gamma[v==i]==1 & disease[v==i]==2)))/ np2
      V44 = (1/(alphaest+(1-alphaest)*qest2) - 1) *(V441 - V442^2)
      *sum(disease[v==i]==2&gamma[v==i]==0)
      V2 = sum(eta3*eta3*gamma[v==i]/alphaest)*(1-alphaest)/alphaest
      var.1 = var.1 +(V1+V2+V33+V44)
    }else if (choice ==2){ ##opt weight 1
      np1= sum(eta[v==i]==1) + sum(gamma[v==i]==1& disease[v==i]==1)
      Q = wY.all[v == i,]* Gcweight[[i.name]]* zminEc * expz[v == i]
      if  (choice.c ==1 ){
        eta31 = rowSums((1-delta[v == i]*(disease[v == i]==1) )*Q*
        do.call(rbind,replicate(sum(v == i),dlamb10t,simplify = FALSE)))
        eta32 = (1-delta[v == i]*(disease[v == i]==1))*eta21
        eta3<- eta31-eta32
      }else{
        eta3 = rowSums((1-delta[v == i]*(disease[v == i]==1) )*Q*
        do.call(rbind,replicate(sum(v == i),dlamb10t,simplify = FALSE)))
      }
      V331 = sum(eta.comb*eta.comb*(eta[v==i]==1 | (gamma[v==i]==1
      & disease[v==i]==1)))/ np1
      V332 = sum(eta.comb*(eta[v==i]==1 | (gamma[v==i]==1 & disease[v==i]==1)))/np1
      V33 = (1/(alphaest+(1-alphaest)*qest1) - 1) *(V331 - V332^2)
      *sum(disease[v==i]==1)
      V2 = sum(eta3*eta3*gamma[v==i]/alphaest)*(1-alphaest)/alphaest
      var.1 = var.1 +(V1+V2+V33)
    }
  }
  var = solve(tempI)%*%var.1%*%solve(tempI)
  return(sqrt(var))
}
```