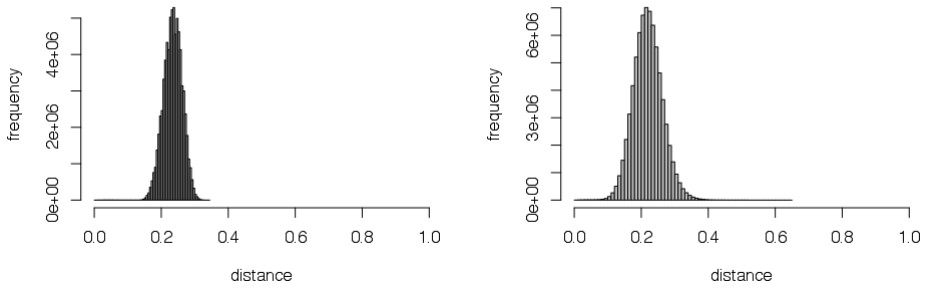
Simulated and UK tree clustered by UCSD soft. - March 2016

S. Le Vu
(Dated: March 23, 2016)

I. INTRO

- Looking for relation between cluster characteristics and heterogenous transmission rates
  - "Time-based" distances from *simulated* coalescent tree are converted to substitutions per site distances with a constant rate from the litterature ($4.3e-3/365$ from *Berry et al. JVI 2007*) (see distributions in Fig. 1)
  - Then matrices of distances are converted in edge lists of pairwise distances with header [ID1, ID2, distance] as needed by UCSD software `hivclustering`
  - Edge lists are inputed in the `hivnetworkcsv` function which returns lists of cluster assignements for thresholds [0.015, 0.02, 0.05, 0.1] for both simulated and UK tree.



(a) Simulation (b) UK

FIG. 1: Distances (subst/site)

```
detail_knitr <- TRUE
source("functions.R")
```

```
library(ape)
```

```
Warning:  package 'ape' was built under R version 3.2.3
```

```
if(TRUE){
  ####---- load sim ----
  ## Load newest Rdata
  l <- list.files(pattern="*.Rdata") # list.files(pattern="Rdata$") list.files(pattern="out")
  l
  load(l[length(l)])
  ls()
  simtree <- tree
  rm(tree)

  ####---- load uk stuff ----
  t_uk <- read.tree(file = "../phylo-uk/data/ExaML_result.subUKogC_noDRM.finaltree.000")
  ## drop OG
```

```
  og <- c("Ref1", "Ref2", "Ref3", "Ref4", "Ref5", "Ref6", "HXB2")
  uktree <- drop.tip(t_uk, og )
 }
```

```
## of (transformed) distances
if(FALSE){

  # function TReeToEdgeList returns path to RDS file

  ##- simtree
  system.time(
  path.simtree_el <- TreeToEdgeList(t = simtree,
                       rate = 4.3e-3/365, # Berry et al. JVI 2007
                       seqlength = 1550)
  )

  ##- uktree (no transformation)
  system.time(
    path.uktree_el <- TreeToEdgeList(uktree,
                         rate = 1) )
}


# Alternatively take TN93 distances
#  dukTN93 <- readRDS(file = "../phylo-uk/source/subUKogC_noDRM_151202_ucsdTN93.rds" )
```

```
## get list of quantiles, commands and list of
## dataframes of cluster members
if(FALSE){
  thresholds = c(0.015, 0.02, 0.05, 0.1)
  system.time(
    simclus <- ucsd_hivclust(path.el = path.simtree_el,
                             thr = thresholds)
  )

  system.time(
    ukclus <- ucsd_hivclust(path.el = path.uktree_el,
                            thr = thresholds)
  )
  names(simclus)
  names(ukclus)
  # unlist(ukclus["warn"] , recursive = T, use.names = F)

  ###--- save
  saveRDS(simclus, file = "data/simclus2.rds")
  saveRDS(ukclus, file = "data/ukclus2.rds")
 }

# rm(dsimtree, duktree)
```

II. UCSD HIVCLUSTERING

Read saved results from UCSD hivclustering

```
### only cluster members (based on given threshold)
simclus <- readRDS(file = "data/simclus2.rds")[["cl"]]
ukclus <- readRDS(file = "data/ukclus2.rds")[["cl"]]

### only cluster members (based on quantiles)
# simclus <- readRDS(file = "data/simclus.rds")[[3]]
# ukclus <- readRDS(file = "data/ukclus.rds")[[3]]
```

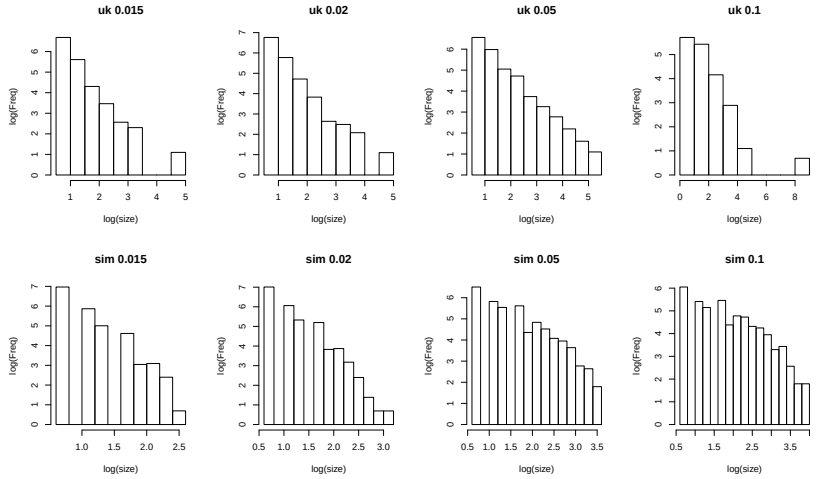
- Now, thresholds for clustering are fixed and not determined by quantiles of distances
- Because of this and because of the applied substitution rate, the number of clusters and mean sizes of clusters are not identical but are of the same magnitude between simulated and real UK data
- Up to threshold = 5%, distributions of cluster size seem to follow a power law for both simulated and UK data

Number of clusters and stats for simulated and UK trees (cluster size 1 does not exist in these (UCSD) outputs)

```
##- Calculate size(=Freq) of each cluster across different threshold
simfreqClust <- lapply(simclus,
                       function(x) as.data.frame(table(x$ClusterID),
                       stringsAsFactors = FALSE))
ukfreqClust <- lapply(ukclus,
                      function(x) as.data.frame(table(x$ClusterID),
                      stringsAsFactors = FALSE))
```

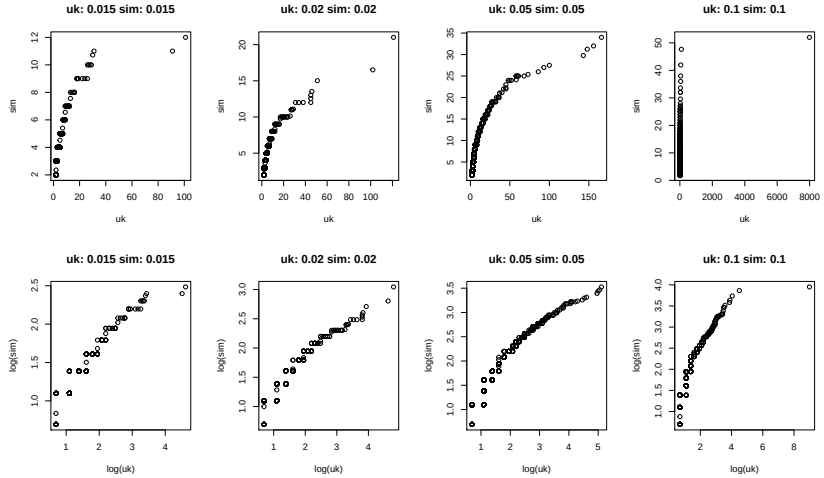
```
##- number of different clusters by threshold
sapply(simfreqClust, function(x) dim(x)[1])

0.015  0.02  0.05   0.1
 1713  2048  2006  1632

sapply(ukfreqClust, function(x) dim(x)[1])

0.015  0.02  0.05   0.1
 1199  1374  1460   611

##- cluster size
sapply(simfreqClust, function(x) summary(x$Freq))

         0.015   0.02   0.05    0.1
Min.     2.000  2.000  2.000  2.000
1st Qu.  2.000  2.000  2.000  2.000
Median   2.000  2.000  3.000  4.000
Mean     2.762  3.138  5.182  6.913
3rd Qu.  3.000  4.000  6.000  9.000
Max.    12.000 21.000 34.000 52.000

sapply(ukfreqClust, function(x) summary(x$Freq))

          0.015    0.02    0.05     0.1
Min.      2.000   2.000   2.000    2.00
1st Qu.   2.000   2.000   2.000    2.00
Median    2.000   2.000   3.000    3.00
Mean      3.188   3.531   5.442   17.84
3rd Qu.   3.000   3.000   4.000    4.00
Max.    101.000 121.000 166.000 7986.00
```

Plots of log(size) for UK and simulated clusters



QQ plots UK vs simulated, untransformed and log-log

```
null device
          1
```



III. ASSOCIATIONS

After merging with co-variates allocated from demes states contained in tree, non-clustering individuals are assigned a cluster size of 1.

The proportion of individuals into clusters (0 = out of, 1 = in a cluster) and stats for "size of cluster for each individuals".

```
demo <- readRDS("demo.rds")
##- date of diagnosis ?
date0 <- as.Date('1979-01-01')
demo$datediag <- date0 + demo$time
# min(demo$datediag) head(demo)
# max(demo$datediag)

###--- add cluster sizes
####... and outdegrees
```

```
### --- start function clust.stats --- ###
##- calculate both numclus and sizeclus for each seqindex into a LIST
##- with same variable names
##- For UCSD files, no ID if no cluster (size < 2) !

clust.stats <- function(clus = simclus, tree = simtree){

  ## get ALL tips names
  tip.names <- data.frame("id" = tree$tip.label ,
                          stringsAsFactors = F)
  ## get size of clusters
  freqClust <- lapply(clus, function(x){
    as.data.frame(table(x$ClusterID),
                  stringsAsFactors = FALSE)
  })

  ## empty list
  l <- list()

  ## loop over thresholds
  for (i in 1:length(clus) ) {
  ## merge cluster number (with NA)
   a <- merge(x = tip.names, y = clus[[i]],
          by.x = "id", by.y = "SequenceID",
          all.x = TRUE, sort = FALSE)

   ## merge cluster size (with NA)
   b <- merge(x = a, y = freqClust[[i]],
              by.x = "ClusterID", by.y = "Var1",
              all.x = TRUE, sort = FALSE)

  #- size 1 if not into a cluster
  b$Freq[is.na(b$Freq)] <- 1

  #- binary clustering variable
  b$binclus <- ifelse(b$Freq > 1 & !is.na(b$Freq), 1, 0)

  ##- pseudo ClusterID when size = 1
  ## starting from max(ClusterID+1)
  ## important for down-sampling
  .start <- max(b$ClusterID, na.rm = T)
  .n <- dim(b[is.na(b$ClusterID),] )[1]
   b$ClusterID[is.na(b$ClusterID)]  <- .start + 1:.n

  #- colnames
  colnames(b)[which(colnames(b) =="Freq")] <- "size"
  l[[i]] <- b
  names(l)[i] <- names(freqClust[i])
  }
  return(l)
}
### --- end function clust.stats --- ###

l_sim <-  clust.stats(clus = simclus, tree = simtree)
l_uk <-  clust.stats(clus = ukclus, tree = uktree)
# str(l_sim)
# str(l_uk)
```

```r
##- number of clusters (counting size=1 cluster !)
sapply(l_sim, function(x) length(unique(x$ClusterID)))
```

```
0.015  0.02  0.05   0.1
 9146  7785  3774  2514
```

```r
sapply(l_uk, function(x) length(unique(x$ClusterID)))
```

```
0.015  0.02  0.05   0.1
 9541  8686  5679  1872
```

```r
##-proportion in or out clusters
sapply(l_sim, function(x) round(prop.table(table(x$binclus)),2))
```

```
  0.015 0.02 0.05  0.1
0  0.61 0.47 0.15 0.07
1  0.39 0.53 0.85 0.93
```

```r
sapply(l_uk, function(x) round(prop.table(table(x$binclus)),2))
```

```
  0.015 0.02 0.05 0.1
0  0.69  0.6 0.35 0.1
1  0.31  0.4 0.65 0.9
```

```r
##- cluster sizes (by individuals having such a size !!)
## probably meaningless
sapply(l_sim, function(x) summary(x$size))
```

```
         0.015  0.02   0.05   0.1
Min.     1.000  1.00  1.000  1.00
1st Qu.  1.000  1.00  2.000  4.00
Median   1.000  2.00  6.000 10.00
Mean     1.951  2.74  7.949 12.29
3rd Qu.  2.000  3.00 11.000 17.00
Max.    12.000 21.00 34.000 52.00
```

```r
sapply(l_uk, function(x) summary(x$size))
```

```
          0.015    0.02   0.05  0.1
Min.      1.000   1.000   1.00    1
1st Qu.   1.000   1.000   1.00   11
Median    1.000   1.000   3.00 7986
Mean      3.941   5.853  18.82 5247
3rd Qu.   2.000   3.000  16.00 7986
Max.    101.000 121.000 166.00 7986
```

```r
##- for simulated data
listclus <- lapply(l_sim, function(x)
merge(x, demo,
      by.x = "id", by.y = "patient",
      all.x = T, sort = FALSE))

# head(listclus[[3]])
# table(listclus[[1]]$binclus, useNA = "ifany")
# table(listclus[[1]]$size, useNA = "ifany")
# saveRDS(listclus, file = "data/listclus_sim.rds")

############################
### --- Regressions --- ###
```

```
############################
```

A. Naive regressions on simulation

Linear regression with co-variates as ordinal or categorical

```
##- linear
lm_model_factor <- "size ~ factor(stage) + factor(risk) + factor(age)"
lm_model_ordinal  <-  "scale(size) ~ scale(age) + scale(stage) + scale(time) + scale(risk)"
lm_model_ordinal_wo_time  <-  "scale(size) ~ scale(age) + scale(stage) + scale(risk)"

##- logistic
logit_model_ord  <-  "binclus ~ scale(age) + scale(stage) + scale(time) + scale(risk)"
logit_model_fact <-  "binclus ~ factor(stage) + factor(risk) + factor(age)"

##- standard output
lapply(listclus , function(x) summary(lm(lm_model_ordinal, data = x)))

$`0.015`

Call:
lm(formula = lm_model_ordinal, data = x)

Residuals:
    Min      1Q  Median      3Q     Max
-1.1738 -0.5708 -0.2452  0.1764  5.6731

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.591e-14  8.664e-03   0.000    1.000
scale(age)   -3.771e-03  8.731e-03  -0.432    0.666
scale(stage) -9.522e-02  8.737e-03 -10.899   <2e-16 ***
scale(time)   2.721e-01  8.698e-03  31.283   <2e-16 ***
scale(risk)  -1.263e-02  8.666e-03  -1.458    0.145
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9555 on 12159 degrees of freedom
Multiple R-squared:  0.08723,Adjusted R-squared:  0.08693
F-statistic: 290.5 on 4 and 12159 DF,  p-value: < 2.2e-16


$`0.02`

Call:
lm(formula = lm_model_ordinal, data = x)

Residuals:
    Min      1Q  Median      3Q     Max
-1.2970 -0.5903 -0.2302  0.2393  6.8647

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)  -1.894e-15  8.605e-03   0.000    1.000
scale(age)   -5.321e-03  8.672e-03  -0.614    0.540
scale(stage) -8.114e-02  8.678e-03  -9.350   <2e-16 ***
```

```
scale(time)   2.984e-01  8.639e-03  34.535   <2e-16 ***
scale(risk)  -1.325e-02  8.607e-03  -1.540    0.124
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9491 on 12159 degrees of freedom
Multiple R-squared:  0.09959,Adjusted R-squared:  0.09929
F-statistic: 336.2 on 4 and 12159 DF,  p-value: < 2.2e-16


$`0.05`

Call:
lm(formula = lm_model_ordinal, data = x)

Residuals:
    Min      1Q  Median      3Q     Max
-1.5705 -0.6518 -0.3025  0.4489  4.3290

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -5.295e-15  8.663e-03   0.000 1.000000
scale(age)   -1.333e-02  8.730e-03  -1.527 0.126680
scale(stage) -3.364e-02  8.736e-03  -3.851 0.000118 ***
scale(time)   2.885e-01  8.697e-03  33.176  < 2e-16 ***
scale(risk)  -3.042e-02  8.665e-03  -3.511 0.000448 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9554 on 12159 degrees of freedom
Multiple R-squared:  0.0875,Adjusted R-squared:  0.0872
F-statistic: 291.5 on 4 and 12159 DF,  p-value: < 2.2e-16


$`0.1`

Call:
lm(formula = lm_model_ordinal, data = x)

Residuals:
    Min      1Q  Median      3Q     Max
-1.5443 -0.6792 -0.2690  0.4122  4.3720

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.997e-15  8.804e-03   0.000   1.0000
scale(age)   -4.255e-03  8.873e-03  -0.480   0.6315
scale(stage) -2.563e-02  8.878e-03  -2.886   0.0039 **
scale(time)   2.351e-01  8.839e-03  26.594   <2e-16 ***
scale(risk)  -2.246e-02  8.806e-03  -2.550   0.0108 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.971 on 12159 degrees of freedom
Multiple R-squared:  0.05747,Adjusted R-squared:  0.05716
F-statistic: 185.4 on 4 and 12159 DF,  p-value: < 2.2e-16

####--- sim naive regressions ---
```

```
reg.sum(ls = listclus, reg = lm, model = lm_model_ordinal)

$model
[1] "scale(size) ~ scale(age) + scale(stage) + scale(time) + scale(risk)"

$parameter
                0.015     0.02     0.05      0.1
(Intercept)   2.6e-14 -1.9e-15 -5.3e-15  4.0e-15
scale(age)   -3.8e-03 -5.3e-03 -1.3e-02 -4.3e-03
scale(stage) -9.5e-02 -8.1e-02 -3.4e-02 -2.6e-02
scale(time)   2.7e-01  3.0e-01  2.9e-01  2.4e-01
scale(risk)  -1.3e-02 -1.3e-02 -3.0e-02 -2.2e-02

$pvalue
             0.015 0.02 0.05 0.1
(Intercept)
scale(age)
scale(stage)   ***  ***  ***  **
scale(time)    ***  ***  *** ***
scale(risk)              ***   *

$r.squared
 0.015   0.02   0.05    0.1
0.0872 0.0996 0.0875 0.0575

reg.sum(ls = listclus, reg = lm, model = lm_model_factor)

$model
[1] "size ~ factor(stage) + factor(risk) + factor(age)"

$parameter
                0.015   0.02  0.05   0.1
(Intercept)     2.500  3.500  9.50 15.00
factor(stage)2 -0.370 -0.430 -0.74 -1.10
factor(stage)3 -0.400 -0.440 -0.53 -0.74
factor(stage)4 -0.570 -0.730 -1.10 -1.50
factor(stage)5 -0.710 -0.960 -1.40 -1.60
factor(risk)2  -0.057 -0.092 -0.54 -0.59
factor(age)2   -0.016 -0.150 -0.26 -0.91
factor(age)3   -0.170 -0.320 -0.94 -1.60
factor(age)4   -0.100 -0.250 -0.72 -1.10

$pvalue
               0.015 0.02 0.05 0.1
(Intercept)      ***  ***  *** ***
factor(stage)2   ***  ***  ***  **
factor(stage)3   ***  ***    *   *
factor(stage)4   ***  ***  *** ***
factor(stage)5   ***  ***  *** ***
factor(risk)2              ***   *
factor(age)2                     *
factor(age)3       *   **  *** ***
factor(age)4           *   **  **

$r.squared
  0.015    0.02    0.05     0.1
0.01590 0.01270 0.00650 0.00453

##- logistic
```

```
##- model: clus ~ age +  stage + time + risk
##- care = 1 for all at diagnosis

# logfit <- lapply(simli , function(x){
#   summary(glm(formula = logit_model_std,
#               data = x, family = binomial(link = "logit")))
#   })
```

- The size of cluster of each invidividuals is always explained by overall stage and time of sampling
- For every threshold of clustering, stage 1 is more likely to belong to large clusters than any other stages
- Age 1 is more likely to belong to large clusters than age 3 or age 4
- Low (!) risk level is associated with larger clusters, only for higher thresholds (fewer and larger clusters on overall)
- Small part of the variance is explained by the variables

Logistic regression with ordinal and categorical variables

```
reg.sum(ls = listclus, reg = glm, model = logit_model_ord, family = binomial(link = "logit"))

$model
[1] "binclus ~ scale(age) + scale(stage) + scale(time) + scale(risk)"

$parameter
              0.015   0.02   0.05   0.1
(Intercept)  -0.500  0.120  2.100  3.00
scale(age)   -0.043 -0.080 -0.065 -0.09
scale(stage) -0.260 -0.240 -0.160 -0.14
scale(time)   0.570  0.580  0.860  0.98
scale(risk)  -0.046 -0.053 -0.130 -0.13

$pvalue
             0.015 0.02 0.05 0.1
(Intercept)    ***  ***  *** ***
scale(age)       *  ***    *   *
scale(stage)   ***  ***  *** ***
scale(time)    ***  ***  *** ***
scale(risk)      *   **  *** ***

reg.sum(ls = listclus, reg = glm, model = logit_model_fact, family = binomial(link = "logit"))

$model
[1] "binclus ~ factor(stage) + factor(risk) + factor(age)"

$parameter
                0.015  0.02  0.05   0.1
(Intercept)     0.290  1.00  2.40  3.50
factor(stage)2 -0.420 -0.42 -0.14 -0.24
factor(stage)3 -0.520 -0.46 -0.20 -0.18
factor(stage)4 -0.790 -0.75 -0.42 -0.48
factor(stage)5 -0.930 -0.91 -0.60 -0.60
factor(risk)2  -0.110 -0.13 -0.30 -0.30
factor(age)2   -0.054 -0.19 -0.17 -0.43
factor(age)3   -0.220 -0.38 -0.38 -0.62
factor(age)4   -0.230 -0.40 -0.37 -0.62
```

```
$pvalue
               0.015 0.02 0.05 0.1
(Intercept)      ***  ***  *** ***
factor(stage)2   ***  ***        .
factor(stage)3   ***  ***    *
factor(stage)4   ***  ***  *** ***
factor(stage)5   ***  ***  *** ***
factor(risk)2      *   **  *** ***
factor(age)2           *        *
factor(age)3      **  ***   **  **
factor(age)4      **  ***   **  **
```

All variables are significantly associated with being into a cluster for all thresholds

B. Regressions on down-sampled simulation

- For each clustering threshold,
- sample one individual by cluster
- re-sample 100 times
- sample size equals number of clusters with each non-clustered individuals counting for one

1. First type of analysis
   - apply 100 linear regressions on each sample
   - calculate proportion of p-values < 0.05 over the 100 iterations

```
### De-correlating: sample one individual by cluster. Repeat many times. Ensure much more p
head(listclus[[1]])
   id ClusterID size binclus  time age care stage risk   datediag
1  40      1520    3       1 12402   1    1     1    1 2012-12-15
2 211      1520    3       1 12311   4    1     5    1 2012-09-15
3  32      1520    3       1 12402   1    1     3    1 2012-12-15
4 611        40    2       1 12098   4    1     1    1 2012-02-15
5 440        40    2       1 12188   3    1     4    1 2012-05-15
6 964       430    2       1 11914   4    1     3    2 2011-08-15
summary(listclus[[1]]$size) ## contains size = 1
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.000   1.000   1.951   2.000  12.000
###--- start function by threshold
# df <- listclus[[1]]

downsample <- function(df, lm_model = lm_model, var = c("time", "age", "care", "stage", "ri
  ##- sampling one id per cluster k times
  k <- iter
  ## loop
    ## empty list
    down_listclus <- vector("list", k)
    ##  k selection of one id from df
    ##   sampled by each ClusterID
    for (i in 1:k){
      down_listclus[[i]] <- df[df$id %in%
                                 tapply(df$id,
                  df$ClusterID,
                  function(x) sample(x, 1)),]
    }
```

```r
  # head(down_listclus[[1]])
  # dim(down_listclus[[1]])

  ##- linear regression for each iteration
  fit <- lapply(down_listclus,
                function(x)
                  summary(lm(lm_model, data = x)))

  ##- extract coefficient
    ## number of variables + intercept
    nvar <- dim(coef(fit[[1]]))[1]

    ## empty matrix of coefficients
    coef_lm <- matrix(NA, nvar * k, 4,
                  dimnames = list(
                    rep(rownames( coef(fit[[1]]) ), k),
                    colnames(coef(fit[[1]]))))
    ## loop
    for (i in 1:length(fit)){
      fill <- (i-1)*nvar + 1:nvar
      coef_lm[ fill,  ] <- coef(fit[[i]])
    }

  ##- number of p-value < 0.05
  sum <- tapply(coef_lm[,4], rownames(coef_lm),
                function(x) sum(x < 0.05) / length(x))

  ##- mean of co-variates over iterations
  # head(down_listclus[[1]])
    ## change structure
    bind_df <- do.call(rbind, down_listclus)
    ## mean by ClusterID of set of variables var
    mean.sample <- aggregate(bind_df[, var],
                             list("ClusterID" = bind_df$ClusterID,
                                  "size" = bind_df$size),

  return(list(# down_listclus = down_listclus,
              #fit = fit, coef_lm = coef_lm,
              percent.signif =  sum, mean.sample = mean.sample))
}
```

```r
### over thresholds

## ordinal variables
  dd_ord <- sapply( listclus, function(x) {
    downsample(df = x,
               lm_model = lm_model_ordinal,
               var = c("time", "age", "care", "stage", "risk"),
               iter = 100)
  }
  )
  ## percent of signif paramater
  t(do.call(rbind, dd_ord["percent.signif", ] ))
```

	0.015	0.02	0.05	0.1
(Intercept)	0.00	0.00	0.00	0.00
scale(age)	0.22	0.43	0.13	0.09
scale(risk)	0.20	0.26	0.48	0.31
scale(stage)	1.00	1.00	0.75	0.62

```
scale(time)   1.00 1.00 1.00 1.00
```

```
## categorical variables
  dd_cat <- sapply( listclus, function(x) {
    downsample(df = x,
               lm_model = lm_model_factor,
               var = c("time", "age", "care", "stage", "risk"),
               iter = 100)
  }
  )
  ## percent of signif paramater
  t(do.call(rbind, dd_cat["percent.signif", ] ))
```

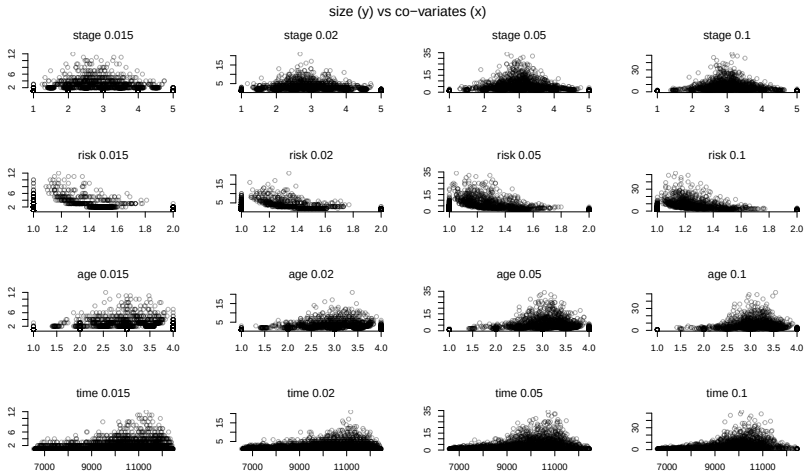
```
               0.015 0.02 0.05  0.1
(Intercept)     1.00 1.00 1.00 1.00
factor(age)2    0.10 0.13 0.09 0.14
factor(age)3    0.34 0.65 0.38 0.35
factor(age)4    0.28 0.69 0.36 0.37
factor(risk)2   0.28 0.30 0.57 0.42
factor(stage)2  1.00 0.98 0.28 0.17
factor(stage)3  1.00 0.99 0.25 0.12
factor(stage)4  1.00 1.00 0.80 0.60
factor(stage)5  1.00 1.00 0.98 0.78
```

- For first two threshold, first stage of infection and recent time of sampling are always (100%) associated with cluster size
- As sizes of clusters increase, young age and low-risk tend to be associated with cluster size (in maximum 47% and 63% of samples respectively)

2. Second type of analysis
   - calculate the mean of co-variates over the 100 iterations
   - apply one linear regression: size mean(covariates)

```
  ## mean of co-variates by cluster
  # str(dd["mean.sample",])
  mean.down <- dd_ord["mean.sample",]

  ##- linear regression ordinal
#    lapply(mean.down, function(x) {
#    summary(lm(lm_model_ordinal, data = x))})
  reg.sum(ls = mean.down, reg = lm, model = lm_model_ordinal)

$model
[1] "scale(size) ~ scale(age) + scale(stage) + scale(time) + scale(risk)"

$parameter
                0.015     0.02     0.05      0.1
(Intercept)  -2.5e-14  2.6e-14  5.1e-15  1.0e-14
scale(age)   -1.2e-02 -2.0e-02 -1.4e-02 -1.4e-02
scale(stage) -8.1e-02 -8.2e-02 -4.6e-02 -5.2e-02
scale(time)   1.8e-01  2.0e-01  3.1e-01  3.5e-01
scale(risk)  -1.4e-02 -1.8e-02 -3.3e-02 -3.2e-02

$pvalue
             0.015 0.02 0.05 0.1
(Intercept)
scale(age)           .
scale(stage)   ***  ***   **  **
scale(time)    ***  ***  *** ***
```

```
scale(risk)                *    .

$r.squared
 0.015   0.02   0.05    0.1
0.0398 0.0487 0.1030 0.1290

  ## plot
  size.vs.covar(mean.down)
  # dev.off()
```



- First stages of infection and recent time of sampling are always (100%) associated with larger cluster size

C. On real UK data

Same process ...

```
##- add demo outcome: stage of infection, treatment status, age group, CHIC or not
load("../phylo-uk/data/sub.RData")
rm(s)
##- selection of df covariates
y <- df[,c("seqindex","patientindex",
           "agediag", "cd4", "vl", "onartflag",
           "ydiag", "agediag_cut", "cd4cut",
           "ydiag_cut", "CHICflag", "status")]

y$logvl <- log(y$vl)
y$sqrtcd4 <- sqrt(y$cd4)
```

```
listUKclus <- lapply(l_uk, function(x)
  merge(x, y,
        by.x = "id", by.y = "seqindex",
        all.x = T, sort = FALSE))
# head(listUKclus[[1]])

####--- naive regressions ---
```

Linear regression

```
#### just on low and high threshold (but not too high !)
# li <- listUKclus[ 1:(length(listUKclus)-1) ]
lm_model_uk = "scale(size) ~ scale(agediag) + scale(sqrt(cd4)) +  scale(ydiag)"
# lapply(li, function(x) summary(lm(lm_model_std, data = x)))
reg.sum(ls = listUKclus, reg = lm, model = lm_model_uk)

$model
[1] "scale(size) ~ scale(agediag) + scale(sqrt(cd4)) +  scale(ydiag)"

$parameter
                   0.015    0.02    0.05     0.1
(Intercept)       0.0022  0.0021  0.0024  0.0013
scale(agediag)   -0.0390 -0.0530 -0.0072  0.0970
scale(sqrt(cd4))  0.0380  0.0490  0.0480  0.0130
scale(ydiag)      0.1200  0.1500  0.1100 -0.2500

$pvalue
                 0.015 0.02 0.05 0.1
(Intercept)
scale(agediag)     ***  ***      ***
scale(sqrt(cd4))   ***  ***  ***
scale(ydiag)       ***  ***  *** ***

$r.squared
 0.015   0.02   0.05    0.1
0.0146 0.0234 0.0134 0.0583
```

Young age, high level of CD4 and recent time of diagnosis are associated with larger cluster size

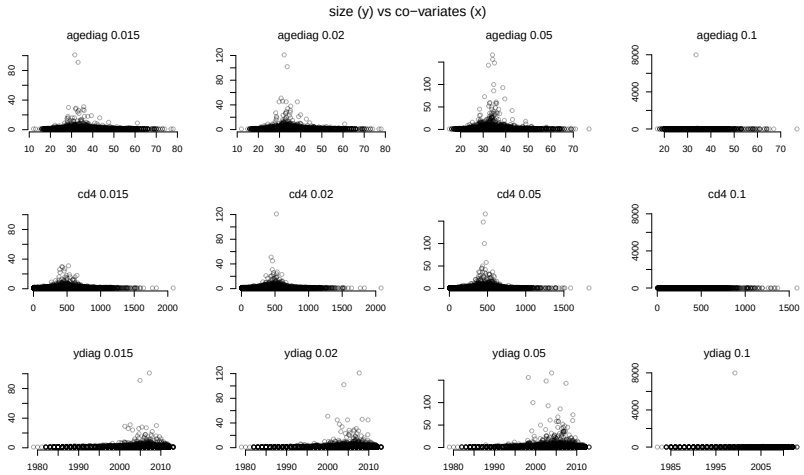
```
##- model: clus ~ age +  stage + time + risk
##- care = 1 for all at diagnosis
## ex.
logit_model_uk = "binclus ~ scale(agediag) + scale(sqrt(cd4)) + scale(ydiag)"

reg.sum(ls = listUKclus, reg = glm, model = logit_model_uk, family = binomial(link = "logit"))

$model
[1] "binclus ~ scale(agediag) + scale(sqrt(cd4)) + scale(ydiag)"

$parameter
                 0.015  0.02  0.05     0.1
(Intercept)      -0.88 -0.47  0.71  2.2000
scale(agediag)   -0.17 -0.19 -0.14  0.0980
scale(sqrt(cd4))  0.22  0.22  0.25  0.2000
scale(ydiag)      0.73  0.75  0.72 -0.0032

$pvalue
                 0.015 0.02 0.05 0.1
(Intercept)        ***  ***  *** ***
scale(agediag)     ***  ***  ***  **
scale(sqrt(cd4))   ***  ***  *** ***
scale(ydiag)       ***  ***  ***
```

Same associations with binary cluster membership

```
### over thresholds

## ordinal variables
dd_ord_uk <- sapply( listUKclus, function(x) {
  downsample(df = x,
             lm_model = lm_model_uk,
             var = c("agediag", "cd4", "ydiag"),
             iter = 100)
}
)
## percent of signif paramater
t(do.call(rbind, dd_ord_uk["percent.signif", ] ))
```

```
                 0.015 0.02 0.05  0.1
(Intercept)       0.00 0.00 0.00 0.02
scale(agediag)    0.63 0.60 0.13 0.04
scale(sqrt(cd4))  0.85 0.92 0.76 0.07
scale(ydiag)      1.00 1.00 1.00 0.10
```

```
## mean of co-variates by cluster
# str(dd["mean.sample",])
mean.down_uk <- dd_ord_uk["mean.sample",]
# head(mean.down_uk[[2]])
##- linear regression ordinal
#    lapply(mean.down, function(x) {
#    summary(lm(lm_model_ordinal, data = x))})
reg.sum(ls = mean.down_uk, reg = lm, model = lm_model_uk)

$model
[1] "scale(size) ~ scale(agediag) + scale(sqrt(cd4)) +  scale(ydiag)"

$parameter
                  0.015   0.02    0.05      0.1
(Intercept)      -0.018 -0.022 -0.0450 -2.4e-02
scale(agediag)   -0.017 -0.020 -0.0094 -8.5e-05
scale(sqrt(cd4))  0.034  0.038  0.0420  1.4e-03
scale(ydiag)      0.074  0.080  0.0730  1.9e-03

$pvalue
                 0.015 0.02 0.05 0.1
(Intercept)         **    *  *** ***
scale(agediag)       *    *
scale(sqrt(cd4))   ***  ***  ***  **
scale(ydiag)       ***  ***  *** ***

$r.squared
 0.015   0.02   0.05    0.1
0.0162 0.0121 0.0131 0.0170

## plot
size.vs.covar(l = mean.down_uk, depvar = "size",
indepvar = c("agediag", "cd4", "ydiag"))
# dev.off()
```



CD4 and year of diagnosis systematically associated with cluster size / membership.