```
opts_chunk$set (warning = FALSE, message = FALSE, tidy=TRUE, echo=TRUE)
options(warn = -1)

rm(list=ls())

library(survival)
library(knitr)
library(kableExtra)

library(glmnet)
```

```
## Loading required package:  Matrix
## Loaded glmnet 4.1-7
```

```
library(ggplot2)

# Following loaded in "forest_search_v0.R"
suppressMessages(library(randomForest))
#library(SPlit)

library(grf)
library(policytree)
library(DiagrammeR)

#library(cowplot)

library(data.table)
library(plyr)
library(aVirtualTwins)
# Not sure formatR is needed?
#library(formatR)
suppressMessages(library(gridExtra))

library(speff2trial)
```

```
## Loading required package:  leaps
```

```
# Location where code is stored
# Modified for MAC
codepath<-c("/Users/larryleon/Documents/GitHub/forestSearch/R/")
source(paste0(codepath,"source_forestsearch_v0.R"))
source_fs_functions(file_loc=codepath)

# Output grf, fs, and fs bootstrap
#outgrf<-c("output/actg_2v3_grf.Rdata")
#outfs<-c("output/actg_2v3_fs.Rdata")
# Boots=2000
outfsboot<-c("output/actg_2v3_fsboot_B=2000.Rdata")
# Set to null if not outputting
outgrf<-outfs<-NULL
#outfsboot<-NULL
```

```
t.start.all <- proc.time()[3]
# GRF analysis To guide selection of binary cutpoints
df.analysis <- subset(ACTG175, arms %in% c(2, 3))
```

```r
df.analysis <- within(df.analysis, {
    id <- as.numeric(c(1:nrow(df.analysis)))
    time_days <- days
    treat <- ifelse(arms == 2, 1, 0)
})

# plot(survfit(Surv(time_days,cens)~treat,data=df.analysis))
coxph(Surv(time_days, cens) ~ treat, data = df.analysis)

## Call:
## coxph(formula = Surv(time_days, cens) ~ treat, data = df.analysis)
##
##          coef exp(coef) se(coef)      z     p
## treat -0.1102    0.8957   0.1303 -0.845 0.398
##
## Likelihood ratio test=0.72  on 1 df, p=0.3974
## n= 1085, number of events= 237

confounders.name <- c("age", "wtkg", "karnof", "cd40", "cd80", "hemo", "homo", "drugs",
    "race", "gender", "oprior", "symptom")
outcome.name <- c("time_days")
event.name <- c("cens")
id.name <- c("id")
treat.name <- c("treat")

n.min <- 60
dmin.grf <- 12
frac.tau <- 0.8

grf.est <- grf.subg.harm.survival(data = df.analysis, confounders.name = confounders.name,
    outcome.name = outcome.name, event.name = event.name, id.name = id.name, treat.name = treat.name,
    n.min = n.min, dmin.grf = dmin.grf, frac.tau = frac.tau, details = TRUE)

## tau= 816.8
##    leaf.node control.mean control.size control.se treated.mean treated.size
## 1          2   -22.193446   827.000000   9.331696    22.193446   827.000000
## 2          3    21.477741   258.000000  20.649937   -21.477741   258.000000
## 3          4    47.718498    65.000000  39.276673   -47.718498    65.000000
## 4          5   -26.055367   639.000000   9.774097    26.055367   639.000000
## 5          6   -82.738791   112.000000  29.260376    82.738791   112.000000
## 6          7    37.181038   269.000000  20.587761   -37.181038   269.000000
## 31        10   -39.873177   456.000000  13.001638    39.873177   456.000000
## 41        11     8.680150   181.000000  10.451914    -8.680150   181.000000
## 51        12   -82.738791   112.000000  29.260376    82.738791   112.000000
## 61        13    74.565713   124.000000  26.012336   -74.565713   124.000000
## 7         14   104.613363    64.000000  56.207068  -104.613363    64.000000
## 8         15   -73.329685    81.000000  30.542688    73.329685    81.000000
##     treated.se       diff depth
## 1     9.331696  -44.38689     1
## 2    20.649937   42.95548     1
## 3    39.276673   95.43700     2
## 4     9.774097  -52.11073     2
## 5    29.260376 -165.47758     2
## 6    20.587761   74.36208     2
## 31   13.001638  -79.74635     3
```

```
## 41   10.451914   17.36030     3
## 51   29.260376 -165.47758     3
## 61   26.012336  149.13143     3
## 7    56.207068  209.22673     3
## 8    30.542688 -146.65937     3
##   leaf.node control.mean control.size control.se treated.mean treated.size
## 7        14    104.61336     64.00000   56.20707   -104.61336     64.00000
##   treated.se     diff depth
## 7   56.20707 209.2267     3
```

```
cat("Truncation point for RMST:", c(grf.est$tau.rmst), "\n")
```

```
## Truncation point for RMST: 816.8
```

```
# Plot manually

# plot(grf.est£tree) plot(grf.est£tree1) plot(grf.est£tree2)
# plot(grf.est£tree3)

df0.grf <- subset(grf.est$data, treat.recommend == 0)
df1.grf <- subset(grf.est$data, treat.recommend == 1)

# Terminal leaf corresponding to selected SG
cat("Terminal leaf:", c(grf.est$sg.harm.id), "\n")
```

```
## Terminal leaf: karnof <= 90
```

```
# action=1 --> recommend control

# Manually identify the subgroup looking at tree and terminal leaf
print(dim(df0.grf))
```

```
## [1] 64 34
```

```
check <- subset(df.analysis, karnof <= 90 & cd80 > 1034 & age > 37)
print(dim(check))
```

```
## [1] 64 29
```

```
# plot(survfit(Surv(time_days,cens)~treat,data=df.analysis))
# coxph(Surv(time_days,cens)~treat,data=df.analysis)

if (!is.null(outgrf)) save(grf.est, file = outgrf)
```

```
t.done <- proc.time()[3]
t.min <- (t.done - t.start.all)/60
cat("Minutes and hours for GRF estimation", c(t.min, t.min/60), "\n")
```

```
## Minutes and hours for GRF estimation 2.92705 0.04878417
```

```
t.start <- proc.time()[3]

cat("GRF variables in selected tree", "\n")
```

```
## GRF variables in selected tree
```

```r
print(grf.est$tree.names)

## [1] "age"    "cd80"   "wtkg"   "cd40"   "karnof"

cat("GRF cuts wrt selected tree:", "\n")

## GRF cuts wrt selected tree:

print(grf.est$tree.cuts)

## [1] "age <= 37"     "cd80 <= 499"   "cd80 <= 1034"  "wtkg <= 64.64"
## [5] "cd40 <= 417"   "cd80 <= 680"   "karnof <= 90"

# Reduce dimension via Cox lasso

xx <- as.matrix(df.analysis[, confounders.name])
yy <- as.matrix(df.analysis[, c("time_days", "cens")])
colnames(yy) <- c("time", "status")

cvfit <- cv.glmnet(xx, yy, family = "cox")  #first do 10-fold cross-validation to select lambda

m <- glmnet(xx, yy, family = "cox", lambda = cvfit$lambda.min)  #plugin the optimal lambda

conflasso.name <- confounders.name[which(m$beta != 0)]

cat("Cox-LASSO selected:", c(conflasso.name), "\n")

## Cox-LASSO selected: age wtkg karnof cd40 cd80 drugs oprior symptom

cat("GRF cuts wrt selected tree:", "\n")

## GRF cuts wrt selected tree:

print(grf.est$tree.cuts)

## [1] "age <= 37"     "cd80 <= 499"   "cd80 <= 1034"  "wtkg <= 64.64"
## [5] "cd40 <= 417"   "cd80 <= 680"   "karnof <= 90"

# Considering continuous factors per GRF cuts Only *also* considering drugs and
# symptom per lasso

df.analysis <- within(df.analysis, {
    z1a <- ifelse(age <= 37, 1, 0)
    z1b <- ifelse(age <= median(age), 1, 0)
    z2 <- ifelse(wtkg <= 65, 1, 0)
    z3 <- ifelse(karnof <= 90, 1, 0)
    z4 <- ifelse(cd40 <= 417, 1, 0)
    z5a <- ifelse(cd80 <= 499, 1, 0)
    z5b <- ifelse(cd80 <= 680, 1, 0)
    z5c <- ifelse(cd80 <= 1034, 1, 0)
    # z6<-hemo z7<-homo
    z8 <- drugs
    # z9<-race z10<-gender z11<-oprior
    z12 <- symptom
    # Convert to factors
    v1a <- as.factor(z1a)
    v1b <- as.factor(z1b)
    v2 <- as.factor(z2)
```

```
    v3 <- as.factor(z3)
    v4 <- as.factor(z4)
    v5a <- as.factor(z5a)
    v5b <- as.factor(z5b)
    v5c <- as.factor(z5c)
    v6 <- as.factor(z8)
    v7 <- as.factor(z12)
})

FSconfounders.name <- c("v1a", "v1b", "v2", "v3", "v4", "v5a", "v5b", "v5c", "v6",
    "v7")

outcome.name <- c("time_days")
event.name <- c("cens")
id.name <- c("id")
treat.name <- c("treat")

df.confounders <- df.analysis[, FSconfounders.name]
df.confounders <- dummy(df.confounders)

hr.threshold <- 1.5  # Inititial candidates
hr.consistency <- 1.25  # Candidates for many splits

pconsistency.threshold <- 0.9
maxk <- 4
# maxk is max # of covariates in combination Since we want to allow generation
# of intervals for single covariate allowing for 4 can yield v1, v2 (say), and
# v3,v4 with v3 and v4 generating intervals for a single covariate

# Limit timing for forestsearch
max.minutes <- 60
nmin.fs <- 60
# stop.threshold<-0.60 # If any sg meets this, then choose this (stop here);
m1.threshold <- Inf  # Turning this off (Default)
stop.threshold <- 1
# =1 will run through all sg's meeting HR criteria
fs.splits <- 1000  # How many times to split for consistency
# vi is % factor is selected in cross-validation --> higher more important
vi.grf.min <- 0.2
# Null, turns off grf screening Set to 5 for this heavily censored data
d.min <- 5  # Min number of events for both arms (d0.min=d1.min=d.min)
# default=5

sg_focus <- "hr"
split_method <- "Random"
pstop_futile <- 0.3
# Stops the consistency evaluation after first subgroup with consistency below
# pstop_futile With idea that since SG's are sorted by hazard ratio estimates,
# once consistency is below pstop_futile it seems unlikely that SG's with lower
# hr's will reach the required consistency criterion


fs.est <- forestsearch(df = df.analysis, confounders.name = FSconfounders.name, df.predict = df.analysis
    details = TRUE, sg_focus = sg_focus, split_method = split_method, pstop_futile = pstop_futile,
    outcome.name = outcome.name, treat.name = treat.name, event.name = event.name,
```

```
    id.name = id.name, n.min = nmin.fs, hr.threshold = hr.threshold, hr.consistency = hr.consistency,
    fs.splits = fs.splits, stop.threshold = stop.threshold, d0.min = d.min, d1.min = d.min,
    pconsistency.threshold = pconsistency.threshold, max.minutes = max.minutes, maxk = maxk,
    plot.sg = FALSE, vi.grf.min = vi.grf.min)

## Confounders per grf screening v2 v7 v1a v3 v5b v5a v6 v5c v4 v1b
## Number of possible subgroups= 1048575
## Number of possible subgroups (in millions)= 1.048575
## # of subgroups based on # variables > k.max and excluded 1042380
## k.max= 4
## Events criteria for control,exp= 5 5
## # of subgroups with events less than criteria: control, experimental 3625 3911
## # of subgroups meeting all criteria = 1492
## # of subgroups fitted (Cox model estimable) = 1492
## Minutes= 0.1183
## Number of criteria not met for subgroup evaluation
## crit.failure
##       0       1       2       3       4
## 1043872     423    3094     486     700
## Number of subgroups meeting HR threshold 95
## Subgroups (1st 10) meeting overall screening thresholds (HR, m1) sorted by focus: (m1,sg_focus)= Inf
##     K   n  E d1  m1  m0   HR L(HR) U(HR) v2.0 v2.1 v7.0 v7.1 v1a.0 v1a.1 v3.0
##  1: 4 106 21 15 Inf Inf 3.01  1.17  7.77    0    0    0    0     1     0    0
##  2: 4  99 16 11 Inf Inf 2.72  0.95  7.84    1    0    0    0     1     0    0
##  3: 3 124 23 16 Inf Inf 2.62  1.08  6.36    0    0    0    0     1     0    0
##  4: 4  88 30 20 Inf Inf 2.49  1.17  5.34    0    0    0    0     1     0    0
##  5: 4  97 29 18 Inf Inf 2.22  1.05  4.70    0    0    0    0     1     0    0
##  6: 4 110 19 13 Inf Inf 2.20  0.84  5.80    0    0    1    0     1     0    1
##  7: 4  98 31 19 Inf Inf 2.05  0.99  4.22    1    0    0    0     1     0    0
##  8: 4 146 33 18 Inf Inf 2.02  1.02  4.01    1    0    0    0     0     0    0
##  9: 3 120 36 22 Inf Inf 2.02  1.03  3.94    0    0    0    0     1     0    0
## 10: 4 157 33 22 Inf Inf 2.01  0.97  4.14    0    0    1    0     1     0    0
##     v3.1 v5b.0 v5b.1 v5a.0 v5a.1 v6.0 v6.1 v5c.0 v5c.1 v4.0 v4.1 v1b.0 v1b.1
##  1:    0     1     0     0     0    1    0     0     1    0    0     0     0
##  2:    0     1     0     0     0    0    0     0     1    0    0     0     0
##  3:    0     1     0     0     0    0    0     0     1    0    0     0     0
##  4:    1     1     0     0     0    0    0     0     0    0    1     0     0
##  5:    1     1     0     0     0    1    0     0     0    0    0     0     0
##  6:    0     0     0     0     0    0    0     0     1    0    0     0     0
##  7:    1     1     0     0     0    0    0     0     0    0    0     0     0
##  8:    1     0     0     0     0    1    0     0     0    0    0     0     1
##  9:    1     1     0     0     0    0    0     0     0    0    0     0     0
## 10:    0     0     0     1     0    0    0     0     1    0    0     0     0
## Consistency 0.95
## Splitting method, # of splits= Random 1000
## Model, % Consistency Met= v1a.0 v5b.0 v6.0 v5c.1 0.95
## Consistency 0.631
## Consistency 0.913
## Splitting method, # of splits= Random 1000
## Model, % Consistency Met= v1a.0 v5b.0 v5c.1 0.913
## Consistency 0.941
## Splitting method, # of splits= Random 1000
## Model, % Consistency Met= v1a.0 v3.1 v5b.0 v4.1 0.941
## Consistency 0.641
## Consistency 0.333
```

```
## Consistency 0.533
## Consistency 0.542
## Consistency 0.493
## Consistency 0.404
## Consistency 0.501
## Consistency 0.835
## Consistency 0.835
## Consistency 0.17
## Number of subgroups meeting consistency criteria= 3
##    p.consistency Nsg group.id m.index K   M.1   M.2   M.3   M.4
## 1:          0.95 106       25       1 4 v1a.0 v5b.0  v6.0 v5c.1
## 2:         0.913 124       30       3 3 v1a.0 v5b.0 v5c.1
## 3:         0.941  88       56       4 4 v1a.0  v3.1 v5b.0  v4.1
##    p.consistency Nsg group.id m.index K   M.1   M.2   M.3   M.4
## 1:         0.950 106       25       1 4 v1a.0 v5b.0  v6.0 v5c.1
## 2:         0.941  88       56       4 4 v1a.0  v3.1 v5b.0  v4.1
## 3:         0.913 124       30       3 3 v1a.0 v5b.0 v5c.1
```

```
xx <- fs.est$find.grps$out.found$hr.subgroups
covs.found <- xx[, -c(1:10)]
covs.most <- apply(covs.found, 2, sum)
covs.most <- covs.most[covs.most > 0]
print(covs.most)
```

```
##  v2.0  v7.0  v7.1 v1a.0 v1a.1  v3.0  v3.1 v5b.0 v5b.1 v5a.0 v5a.1  v6.0 v5c.0
##    36    17     1    23    16     7    54    23    17    17     2    33    24
## v5c.1  v4.0  v4.1 v1b.0 v1b.1
##    14    12    17     5    34
```

```
print(fs.est$grp.consistency$result)
```

```
##    p.consistency Nsg group.id m.index K   M.1   M.2   M.3   M.4
## 1:         0.950 106       25       1 4 v1a.0 v5b.0  v6.0 v5c.1
## 2:         0.941  88       56       4 4 v1a.0  v3.1 v5b.0  v4.1
## 3:         0.913 124       30       3 3 v1a.0 v5b.0 v5c.1
```

```
df0.fs <- subset(fs.est$df.pred, treat.recommend == 0)
df1.fs <- subset(fs.est$df.pred, treat.recommend == 1)

if (!is.null(outfs)) save(fs.est, df.analysis, FSconfounders.name, file = outfs)
```

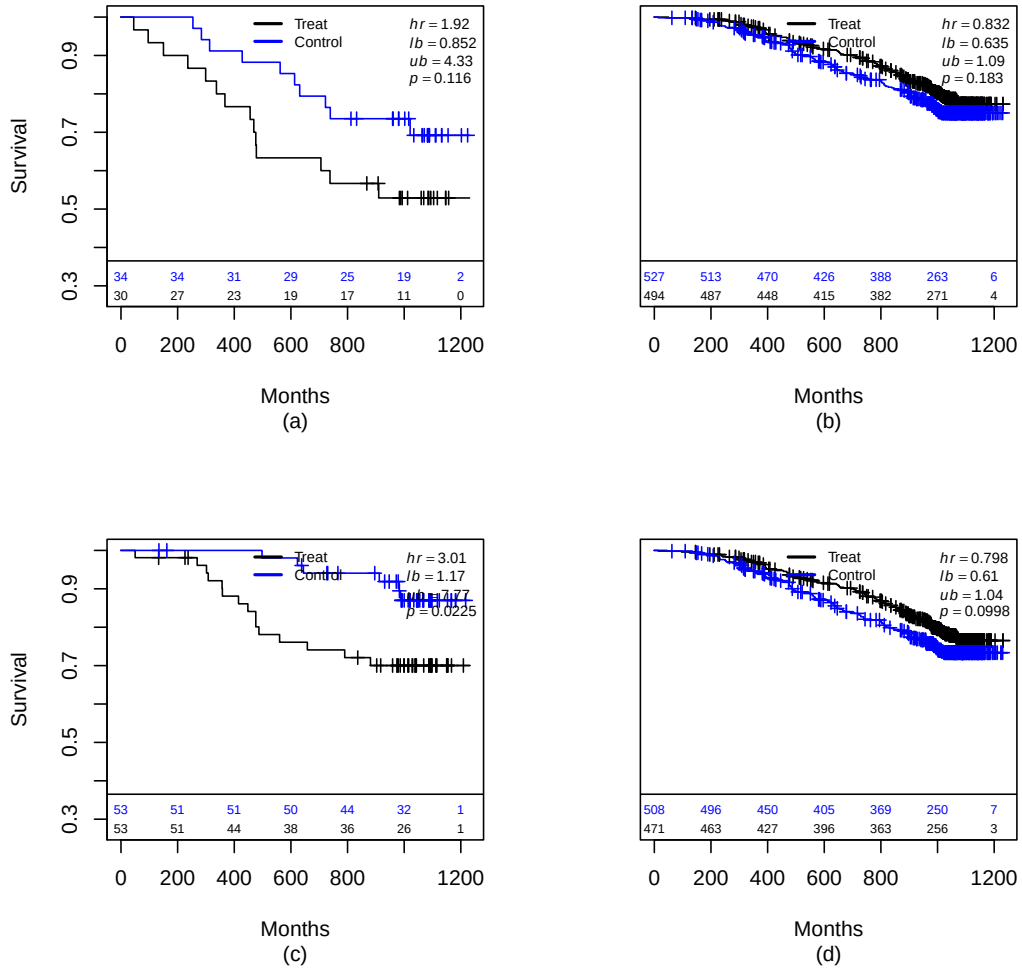
```
t.done <- proc.time()[3]
t.min <- (t.done - t.start)/60
cat("Minutes and hours for FS estimation", c(t.min, t.min/60), "\n")
```

```
## Minutes and hours for FS estimation 0.4088167 0.006813611
```

```
t.start <- proc.time()[3]
# Note, the elements above will need to be re-initiated if running separate
# from above E.g., outcome.names, event.name, ... hr.threshold, etc.

library(doParallel)
registerDoParallel(parallel::detectCores(logical = FALSE))
```

```
# Compare with GRF
layout(matrix(c(1, 2, 3, 4), 2, 2, byrow = TRUE))
plot.subgroup(sub1 = df0.grf, sub1C = df1.grf, tte.name = "time_days", event.name = "cens",
    treat.name = "treat", fix.rows = FALSE, byrisk = 200, show.med = FALSE, ymin = 0.4,
    subtitle1 = "(a)", subtitle2 = "(b)")
plot.subgroup(sub1 = df0.fs, sub1C = df1.fs, tte.name = "time_days", event.name = "cens",
    treat.name = "treat", fix.rows = FALSE, byrisk = 200, show.med = FALSE, ymin = 0.4,
    subtitle1 = "(c)", subtitle2 = "(d)")
```




```
cox.formula.boot <- as.formula(paste("Surv(time_days,cens)~treat"))
split_method <- "Random"
est.loghr <- TRUE

confounders.name <- FSconfounders.name
stop.threshold <- 0.99
fs.splits <- 1000
max.minutes <- 6

# Suggest running 50, first ... to get timing estimate
```

```r
NB <- 2000

df_temp <- fs.est$df.pred[, c("id", "treat.recommend")]
dfa <- merge(df.analysis, df_temp, by = "id")
df_boot_analysis <- dfa

fitH <- get_Cox_sg(df_sg = subset(df_boot_analysis, treat.recommend == 0), cox.formula = cox.formula.boo
    est.loghr = est.loghr)
H_obs <- fitH$est_obs  # log(hr) scale
seH_obs <- fitH$se_obs
# Hc observed estimates
fitHc <- get_Cox_sg(df_sg = subset(df_boot_analysis, treat.recommend == 1), cox.formula = cox.formula.bo
    est.loghr = est.loghr)
Hc_obs <- fitHc$est_obs
seHc_obs <- fitHc$se_obs
rm("fitH", "fitHc")

Ystar_mat <- bootYstar({
    ystar <- get_Ystar(boot)
}, boots = NB, seed = 8316951, counter = "boot", export = fun_arg_list_boot)
# Check dimension
if (dim(Ystar_mat)[1] != NB | dim(Ystar_mat)[2] != nrow(df_boot_analysis)) stop("Dimension of Ystar_mat

tB.start <- proc.time()[3]
# Bootstraps
resB <- bootPar({
    ans <- fsboot_forparallel(boot)
}, boots = NB, seed = 8316951, counter = "boot", export = fun_arg_list_boot)
tB.now <- proc.time()[3]
tB.min <- (tB.now - tB.start)/60

doParallel::stopImplicitCluster()

cat("Minutes for Boots", c(NB, tB.min), "\n")

## Minutes for Boots 2000 83.63755

cat("Projection per 100", c(tB.min * (100/NB)), "\n")

## Projection per 100 4.181878

cat("Propn bootstrap subgroups found =", c(sum(!is.na(resB$H_biasadj_1))/NB), "\n")

## Propn bootstrap subgroups found = 0.981

# How many timmed out
cat("Number timmed out=", c(sum(is.na(resB$H_biasadj_1) & resB$tmins_search > max.minutes)),
    "\n")

## Number timmed out= 0

H_estimates <- get_dfRes(Hobs = H_obs, seHobs = seH_obs, H1_adj = resB$H_biasadj_1,
    ystar = Ystar_mat, cov_method = "standard", cov_trim = 0.05)

Hc_estimates <- get_dfRes(Hobs = Hc_obs, seHobs = seHc_obs, H1_adj = resB$Hc_biasadj_1,
    ystar = Ystar_mat, cov_method = "standard", cov_trim = 0.05)

print(H_estimates)
```

```
##         H0     sdH0 H0_lower H0_upper       H1      sdH1 H1_lower H1_upper
## 1: 3.01229 1.456134  1.16796 7.769007 1.535497 0.2949964 1.053702 2.237587

print(Hc_estimates)

##           H0   sdH0  H0_lower H0_upper      H1      sdH1  H1_lower H1_upper
## 1: 0.7977977 0.1095 0.6096253 1.044053 0.86181 0.1204216 0.6553483 1.133315

if (!is.null(outfsboot)) save(fs.est, Ystar_mat, resB, H_estimates, Hc_estimates,
    df_boot_analysis, file = outfsboot)
```

```
t.done <- proc.time()[3]
t.min <- (t.done - t.start)/60
cat("Minutes and hours for FS bootstrap", c(t.min, t.min/60), "\n")

## Minutes and hours for FS bootstrap 83.67047 1.394508
```

```
df0.fs <- subset(fs.est$df.pred, treat.recommend == 0)
df1.fs <- subset(fs.est$df.pred, treat.recommend == 1)

# ITT analysis
cox_itt <- summary(coxph(Surv(time_days, cens) ~ treat, data = fs.est$df.pred))$conf.int

# ITT estimates
resITT <- c(round(cox_itt[c(1, 3, 4)], 2), nrow(fs.est$df.pred))

# Forest Search Un-adjusted
Hstat <- c(unlist(H_estimates))[c(1, 3, 4)]
resH_obs <- c(c(Hstat), nrow(df0.fs))
# Bias-corrected
Hstat <- c(unlist(H_estimates))[c(5, 7, 8)]
resH_bc <- c(c(Hstat), nrow(df0.fs))

Hstat2 <- c(unlist(H_estimates))[c(5, 7, 8)]
Hstat2 <- round(Hstat2, 2)
a <- paste0(Hstat2[1], " [")
a <- paste0(a, Hstat2[2])
a <- paste0(a, ",")
a <- paste0(a, Hstat2[3])
a <- paste0(a, "]")
H_bc2 <- c(a)

# Un-adjusted
Hcstat <- c(unlist(Hc_estimates))[c(1, 3, 4)]
resHc_obs <- c(c(Hcstat), nrow(df1.fs))
# Bias-corrected
Hcstat <- c(unlist(Hc_estimates))[c(5, 7, 8)]
resHc_bc <- c(c(Hcstat), nrow(df1.fs))



Hcstat2 <- c(unlist(Hc_estimates))[c(5, 7, 8)]
Hcstat2 <- round(Hcstat2, 2)
a <- paste0(Hcstat2[1], " [")
a <- paste0(a, Hcstat2[2])
```

```
a <- paste0(a, ",")
a <- paste0(a, Hcstat2[3])
a <- paste0(a, "]")
Hc_bc2 <- c(a)

res <- rbind(resITT, resH_obs, resH_bc, resHc_obs, resHc_bc)

resf <- as.data.frame(res)

colnames(resf) <- c("HR Estimate", "Lower", "Upper", "$\\#$ Subjects")

rnH <- c("$\\hat{H}$", "$\\hat{H}_{bc}$")
rnHc <- c("$\\hat{H}^{c}$", "$\\hat{H}^{c}_{bc}$")
rnItt <- c("ITT")
rownames(resf) <- c(rnItt, rnH, rnHc)
```

```
# Resolve conflict with dplyr
library(conflicted)
group_rows <- kableExtra::group_rows

options(knitr.kable.NA = ".", format = "latex")
tab_actg <- kbl(resf, longtable = FALSE, align = "c", format = "latex", booktabs = TRUE,
    escape = F, digits = 3, caption = "\\label{tab:actg} ACTG-175 FS Analysis: Cox hazard ratio (HR) est
Cox model estimates are based on subgroups: The estimated subgroup $\\hat{H}$; and
the bootstrap ($B=2000$) bias-correction to $\\hat{H}$ estimates, denoted $\\hat{H}_{bc}$.  Estimates fo
The number of subjects in each population ($\\#$ Subjects) are listed.") %>%
    kable_styling(full_width = FALSE, font_size = 9, latex_options = "hold_position") %>%
    group_rows("ITT", 1, 1) %>%
    group_rows("H subgroup estimates", 2, 3) %>%
    group_rows("H-complement subgroup estimates", 4, 5)
```

Table 1: ACTG-175 FS Analysis: Cox hazard ratio (HR) estimates for the ITT population and subgroups H and H^c. Cox model estimates are based on subgroups: The estimated subgroup $\hat{H}$; and the bootstrap ($B = 2000$) bias-correction to $\hat{H}$ estimates, denoted $\hat{H}_{bc}$. Estimates for the complement H^c are defined analogously. The number of subjects in each population (# Subjects) are listed.

	HR Estimate	Lower	Upper	# Subjects
ITT				
ITT	0.900	0.690	1.160	1085
H subgroup estimates				
$\hat{H}$	3.012	1.168	7.769	106
$\hat{H}_{bc}$	1.535	1.054	2.238	106
H-complement subgroup estimates				
$\hat{H}^c$	0.798	0.610	1.044	979
$\hat{H}^c_{bc}$	0.862	0.655	1.133	979

```
t.done <- proc.time()[3]
t.min <- (t.done - t.start.all)/60
cat("Minutes and hours to finish", c(t.min, t.min/60), "\n")

## Minutes and hours to finish 87.01392 1.450232
```

```r
cat("Machine=", c(Sys.info()[[4]]), "\n")
```

```
## Machine= Mac-Studio-3.local
```

```r
cat("Number of cores=", c(detectCores(logical = FALSE)), "\n")
```

```
## Number of cores= 10
```