Prediction of AML in healthy individuals

Grace Collord & Moritz Gerstung

Fri Feb 16 16:34:37 2018

1 Preliminaries

1.1 Libraries

```
library(CoxHD)
```

```
## Loading required package: survival
```

```
## Loading required package: parallel
```

```
## Loading required package: RColorBrewer
```

```
library(survAUC)
library(survivalROC)
library(glmnet)
```

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

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-13
```

```
library(RColorBrewer)

set1 <- RColorBrewer::brewer.pal(8, "Set1")
```

Helper functions

```
superSet <- function(x, s, fill=NA){
    i <- intersect(colnames(x), s)
    n <- setdiff(s, colnames(x))
    y <- x[,i]
    if(length(n) > 0)
        y <- cbind(y,  matrix(fill, ncol=length(n), dimnames=list(NULL, n)) )[,s]
    return(y)
}
```

2 AML incidence data

Use known AML incidence to correct bias using weighted controls. The expected incidence of AML was calculated from the UK office of national statistics, available at http://www.cancerresearchuk.org/health-professional/cancer-statistics/statistics-by-cancer-type/leukaemia-aml/incidence (http://www.cancerresearchuk.org/health-professional/cancer-statistics/statistics-by-cancer-type/leukaemia-aml/incidence). Spline function to interpolate Male denoted by 1 and female by 0

```
age_incidence <- read.table("aml_age_incidence.txt", header=TRUE, sep="\t")
head(age_incidence)
```

	Age.Range <fctr>	Male.Cases <int>	Female.Cases <int>	Male.Rates <dbl>	Female.Rates <dbl>
1	0 to 04	18	12	0.9	0.6
2	05 to 09	10	10	0.5	0.5
3	10 to 14	8	10	0.4	0.6
4	15 to 19	15	14	0.7	0.8
5	20 to 24	21	18	1.0	0.8
6	25 to 29	22	20	1.0	0.9

6 rows

```
aml_inc <- function(gender, x){
    if(gender==1)
        splinefun(x=c(seq(0,90,5)), y=c(cumsum(age_incidence$Male.Rates/100000)*5)
, method="mono")(x)
    else
        splinefun(x=c(seq(0,90,5)), y=c(cumsum(age_incidence$Female.Rates/100000)*
5), method="mono")(x)
}
```

All cause mortality from the office of national statistics (https://www.ons.gov.uk/ (https://www.ons.gov.uk/)).

```
all_cause_mortality <- read.table("all_cause_mortality.txt", sep="\t", skip=1, hea
der=TRUE)
head(all_cause_mortality)
```

	x <int>	mx <dbl>	qx <dbl>	lx <dbl>	dx <dbl>	ex <dbl>	X <lgl>	mx.1 <dbl>	qx.1 <dbl>
1	0	0.004234	0.004225	100000.0	422.5	79.17	NA	0.003521	0.003515
2	1	0.000306	0.000306	99577.5	30.5	78.51	NA	0.000246	0.000246
3	2	0.000163	0.000163	99547.1	16.2	77.53	NA	0.000137	0.000137
4	3	0.000127	0.000127	99530.8	12.6	76.54	NA	0.000105	0.000105
5	4	0.000090	0.000090	99518.2	8.9	75.55	NA	0.000081	0.000081
6	5	0.000092	0.000092	99509.3	9.2	74.56	NA	0.000067	0.000067

6 rows | 1-10 of 13 columns

```
all_surv <- function(gender, age1, age2){
    if(gender==1)
        s <- all_cause_mortality$lx
    else
        s <- all_cause_mortality$lx.1
    f <- function(x) exp(splinefun(all_cause_mortality$x, log(s), method="mono")(x
))
    f(age2) / f(age1)
}
```

Function combining both

```
aml_inc_cr <- Vectorize(function(gender, age1, age2) sum(diff(aml_inc(gender, seq(
age1,age2,1) ))*all_surv(gender, age1, seq(age1,age2-1,1)) ), c("gender","age1","a
ge2"))
```

3 Discovery cohort

3.1 Data

4 (of 95) cases that were sampled within 6 months of AML diagnosis are excluded to avoid skewing model towards significance

```
f = "./arch_data/DC_vaf_matrix_414ctrl_91aml.csv"
torontoData <- read.csv(f)

torontoData$gender <- ifelse(torontoData$Sex == "male", 1,
        ifelse(torontoData$Sex == "female", 0, torontoData$Sex))

torontoData$gender <- as.numeric(torontoData$gender)
colnames(torontoData)
```

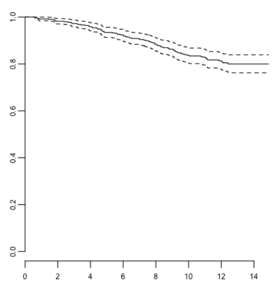
```
##  [1] "Sample"     "ASXL1"      "BCOR"       "CALR"       "CBL"        "DNMT3A"
"IDH1"       "IDH2"
##  [9] "JAK2"       "KDM6A"      "KIT"        "KMT2C"      "KRAS"       "NF1"
"NRAS"       "PHF6"
## [17] "PTPN11"     "RUNX1"      "SF3B1"      "SRSF2"      "TET2"       "TP53"
"U2AF1"      "Diagnosis"
## [25] "fu_years"   "age"        "Sex"        "no_drivers" "gender"
```

Manually standardize

```
torontoData <- torontoData[!duplicated(torontoData),]

gene_vars <- c("CALR", "NRAS", "DNMT3A", "SF3B1", "IDH1", "KIT", "TET2", "RAD21",
"JAK2", "CBL", "KRAS", "PTPN11", "IDH2", "TP53", "NF1", "SRSF2", "CEBPA", "ASXL1",
"RUNX1", "U2AF1", "BCOR", "KDM6A", "PHF6", "KMT2C", "KMT2D")

torontoX <- torontoData[, colnames(torontoData) %in% c(gene_vars, "age", "gender")
]

torontoX <- as.data.frame(torontoX)
```

Only include genes in model if mutated in >2 samples

```
thr <- 2
torontoX <- torontoX[,colSums(torontoX != 0)>=thr]

torontoGroups <- factor(names(torontoX) %in% c("age","gender")+1, level=1:2, label
s=c("Genes","Demographics"))

torontoX$age <- torontoX$age/10
names(torontoX)[which(names(torontoX)=="age")] <- "age_10"
g <- torontoGroups == "Genes"
torontoX[,g] <- torontoX[,g]*10
names(torontoX)[g] <- paste(names(torontoX)[g], "0.1",sep="_")

torontoSurv <- Surv(time = torontoData$fu_years, event = torontoData$Diagnosis=="A
ML")
plot(survfit(torontoSurv~ 1))
```



4 Validation cohort

4.1 Data

```
f = "./arch_data/VC_vaf_matrix_no_duplicates_262ctrl_29aml.csv"
sangerData <- read.csv(f)
colnames(sangerData)
```

```
##  [1] "Sample"      "ASXL1"       "BCOR"        "CBL"         "CEBPA"       "DNM
T3A"      "IDH1"        "IDH2"
##  [9] "JAK2"        "KMT2C"       "KMT2D"       "KRAS"        "NF1"         "NRA
S"        "PTPN11"      "RAD21"
## [17] "SF3B1"       "SRSF2"       "TET2"        "TP53"        "U2AF1"       "Ind
ividual"  "DOBfuzz"     "hcdate"
## [25] "endpt_age"   "Age_at_dx"   "Diagnosis"   "ever_smoked" "age"         "gen
der"      "systol"      "diastol"
## [33] "bmi"         "cholestl"    "triglyc"     "hdl"         "ldl"         "lym
"         "mcv"         "rdw"
## [41] "wbc"         "rbc"         "hct"         "plt"         "hgb"         "dod
"         "dead"        "dodx"
## [49] "prev_dm"     "prev_mi"     "prev_cva"    "no_drivers"
```

```
head(sangerData[, c("Sample", "gender")]) #male=1, female=0
```

	Sample <fctr>	gender <int>
1	PD29762b	0
2	PD29764b	0
3	PD29792b	0
4	PD29804c	0
5	PD29810c	1
6	PD29836b	0

6 rows

NB all dates are jittered

```
sangerData$hcdate <- as.Date(sangerData$hcdate)
sangerData$dodx <- as.Date(sangerData$dodx)

sangerPatients <- sub("[a-z]+$","", sangerData$Sample)
o <- order(sangerPatients, as.numeric(sangerData$hcdate))

sangerData <- sangerData[o,]
sangerPatients <- sangerPatients[o]

clinical_vars <- c("systol", "diastol", "bmi", "cholestl", "triglyc", "hdl", "ldl"
, "lym", "mcv", "rdw", "wbc", "plt", "hgb")
sangerX <- sangerData[, colnames(sangerData) %in% c(gene_vars, "age","gender",clin
ical_vars)]
sangerX <- as.data.frame(sangerX)

sangerX <- sangerX[,colSums(sangerX != 0,na.rm=TRUE)>=thr]
sangerGroups <- factor(grepl("^[a-z]", colnames(sangerX))*2, levels=0:2, labels=c(
"Genes", "Demographics", "Blood"))
sangerGroups[names(sangerX) %in% c("age","gender")] <- "Demographics"
table(sangerGroups)
```

```
## sangerGroups
##        Genes Demographics        Blood
##           15            2           13
```

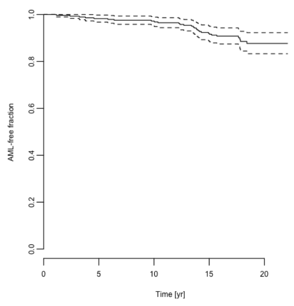
```
g <- sangerGroups=="Genes"
sangerX[g] <- sangerX[g] * 10
names(sangerX)[g] <- paste(names(sangerX[g]),"0.1", sep="_")
y <- StandardizeMagnitude(sangerX[!g])
sangerX <- cbind(sangerX[g],y)

poorMansImpute <- function(x) {x[is.na(x)] <- mean(x, na.rm=TRUE); return(x)}
sangerX <- as.data.frame(sapply(sangerX, poorMansImpute))

foo <- split(sangerData[,c("Diagnosis","hcdate","dodx")], sangerPatients)

bar <- do.call("rbind",lapply(foo, function(x){
                    y <- x
                    n <- nrow(y)
                    y[-n,"Diagnosis"] <- "Control"
                    start <- as.numeric(y$hcdate - y$hcdate[1])/365.25
                    end <- c(as.numeric(y$hcdate - y$hcdate[1])[-1]/365.25, as.num
eric(y$dodx[n] - y$hcdate[1])/365.25)
                    return(data.frame(Diagnosis=y[,"Diagnosis"], start=start, end=
end))
                }))

bar[1:6, ]
```

	Diagnosis <fctr>	start <dbl>	end <dbl>

PD29762	AML	0	9.754962
PD29764	AML	0	10.360027
PD29792	AML	0	14.108145
PD29804	Control	0	5.138946
PD29810	Control	0	18.573580
PD29836.1	Control	0	2.414784

6 rows

```
sangerPatientsSplit <- unlist(sapply(names(foo), function(n) rep(n, nrow(foo[[n]])
)))

sangerSurv <- Surv(time = bar$start, time2 = bar$end, event = bar$Diagnosis!="Cont
rol", origin = 0)
plot(survfit(sangerSurv ~ 1), ylab="AML-free fraction", xlab="Time [yr]")
```



5 Expected AML incidence

5.1 Validation cohort

```
head(sangerSurv)
```

```
## [1] (0, 9.754962]  (0,10.360027]  (0,14.108145]  (0, 5.138946+] (0,18.573580+]
(0, 2.414784+]
```

```
w <- c(which(sangerSurv[,1]==0)[-1]-1, nrow(sangerSurv))
head(sangerSurv[w,])
```

```
## [1] (0.000000, 9.754962]  (0.000000,10.360027]  (0.000000,14.108145]  (0.000000
, 5.138946+] (0.000000,18.573580+]
## [6] (2.414784,10.023272]
```

```
sangerSurv2 <- Surv(sangerSurv[w,2], sangerSurv[w,3])

expected_rate_sanger_cr <- mean(aml_inc_cr(sangerX[w,"gender"],sangerX[w,"age_10"]
*10, sangerX[w,"age_10"]*10+ pmax(1,sangerSurv2[,1]))[!sangerSurv2[,2]])

n_total_sanger <- sum(sangerSurv2[,2])/expected_rate_sanger_cr
n_total_sanger
```

```
## [1] 10406.64
```

5.2 Discovery cohort

```
expected_rate_toronto_cr <- mean(aml_inc_cr(torontoX[,"gender"],torontoX[,"age_10"
]*10, torontoX[,"age_10"]*10+ pmax(1,torontoSurv[,1]))[!torontoSurv[,2]])

n_total_toronto <- sum(torontoSurv[,2])/expected_rate_toronto_cr
n_total_toronto
```

```
## [1] 72377.73
```

6 Combined data

Survival

```
allSurv <- rbind(sangerSurv, Surv(rep(0, nrow(torontoSurv)), torontoSurv[,1], toro
ntoSurv[,2]))
allSurv <- Surv(allSurv[,1], allSurv[,2], allSurv[,3])
```

Data matrix

```
cohort <- c(rep("Sanger", nrow(sangerX)), rep("Toronto", nrow(torontoX)))
i <- c(sort(setdiff(gene_vars,"CALR")),"age","gender")
allX <- rbind(superSet(sangerData,i,fill=0), superSet(torontoData,i,fill=0))
allX <- allX[,colSums(allX>0)>=thr]
allX <- cbind(allX, cohort=cohort=="Sanger") + 0
allGroups <- factor(grepl("^[A-Z]",colnames(allX))+0, levels=1:0, labels=c("Genes"
,"Demographics"))

g <- allGroups=="Genes"
allX <- cbind(10*allX[,g], StandardizeMagnitude(allX[,!g]))
colnames(allX)[g] <- paste(colnames(allX)[g],"0.1",sep="_")
control <- c(sangerData$Diagnosis=="Control", torontoData$Diagnosis=="Control")
```

Weights

```
weights <- rep(1, nrow(allX))
weights[cohort=="Sanger" & control] <- n_total_sanger/sum(cohort=="Sanger" & contr
ol & allSurv[,1]==0)
weights[cohort=="Toronto" & control] <- n_total_toronto/sum(cohort=="Toronto" & co
ntrol)

n_total <- n_total_sanger + n_total_toronto
n_total
```

```
## [1] 82784.38
```

7 Coxph model fits

```
sigma0 <- 0.1
nu <- 1
which.mu <- "Genes"
```

7.1 Discovery cohort

7.1.1 Non-adjusted

```
fitToronto <- CoxRFX(torontoX, torontoSurv, groups=torontoGroups, which.mu=which.m
u, nu=nu, sigma0=sigma0)
waldToronto <- WaldTest(fitToronto)
```

```
##                    group    coef   coef-mu     sd      z df  p.value sig
## ASXL1_0.1          Genes  0.6715  3.40e-02 0.1169  5.745  1 9.19e-09 ***
## CALR_0.1           Genes  0.6168 -2.07e-02 0.0717  8.603  1 7.76e-18 ***
## CBL_0.1            Genes  0.5158 -1.22e-01 0.1311  3.935  1 8.30e-05 ***
## DNMT3A_0.1         Genes  0.5860 -5.15e-02 0.1017  5.761  1 8.36e-09 ***
## IDH1_0.1           Genes  0.6818  4.43e-02 0.1269  5.373  1 7.74e-08 ***
## IDH2_0.1           Genes  0.5153 -1.22e-01 0.1159  4.446  1 8.74e-06 ***
## JAK2_0.1           Genes  0.6967  5.92e-02 0.1249  5.580  1 2.40e-08 ***
## KDM6A_0.1          Genes  0.6375  2.36e-05 0.0581 10.982  1 4.67e-28 ***
## KMT2C_0.1          Genes  0.6602  2.27e-02 0.0618 10.689  1 1.14e-26 ***
## KRAS_0.1           Genes  0.6350 -2.46e-03 0.0581 10.932  1 8.12e-28 ***
## NF1_0.1            Genes  0.6359 -1.61e-03 0.0581 10.947  1 6.86e-28 ***
## PHF6_0.1           Genes  0.6429  5.40e-03 0.0586 10.978  1 4.87e-28 ***
## PTPN11_0.1         Genes  0.6546  1.71e-02 0.0583 11.224  1 3.11e-29 ***
## RUNX1_0.1          Genes  0.3926 -2.45e-01 0.0927  4.236  1 2.27e-05 ***
## SF3B1_0.1          Genes  0.7605  1.23e-01 0.1045  7.274  1 3.49e-13 ***
## SRSF2_0.1          Genes  0.4847 -1.53e-01 0.0944  5.134  1 2.83e-07 ***
## TET2_0.1           Genes  0.6127 -2.48e-02 0.1300  4.712  1 2.46e-06 ***
## TP53_0.1           Genes  0.8595  2.22e-01 0.0875  9.823  1 8.99e-23 ***
## U2AF1_0.1          Genes  0.8524  2.15e-01 0.0785 10.860  1 1.79e-27 ***
## age_10     Demographics -0.0387 -3.87e-02 0.0943 -0.410  1 6.82e-01
## gender     Demographics -0.0434 -4.34e-02 0.1069 -0.406  1 6.85e-01
```

```
survConcordance(fitToronto$surv ~ fitToronto$linear.predictors)
```

```
## Call:
## survConcordance(formula = fitToronto$surv ~ fitToronto$linear.predictors)
##
##   n= 505
## Concordance= 0.7426378 se= 0.03079247
## concordant discordant  tied.risk  tied.time   std(c-d)
##  28925.000  10024.000      0.000      1.000   2398.672
```

7.1.2 Adjusted

```
fitWeightedToronto <- CoxRFX(torontoX, torontoSurv, torontoGroups, which.mu=which.
mu, sigma0=sigma0, nu=nu, weights=weights[cohort=="Toronto"])
waldWeightedToronto <- WaldTest(fitWeightedToronto)
```

```
##                    group    coef coef-mu     sd      z df  p.value sig
## ASXL1_0.1         Genes  1.9481  0.0184 0.1452 13.415  1 4.92e-41 ***
## CALR_0.1          Genes  0.8664 -1.0633 0.7205  1.202  1 2.29e-01
## CBL_0.1           Genes  0.3846 -1.5451 0.3618  1.063  1 2.88e-01
## DNMT3A_0.1        Genes  0.7091 -1.2206 0.1236  5.736  1 9.70e-09 ***
## IDH1_0.1          Genes  2.3976  0.4679 0.3353  7.151  1 8.63e-13 ***
## IDH2_0.1          Genes  0.8112 -1.1185 0.2286  3.548  1 3.88e-04 ***
## JAK2_0.1          Genes  1.9253 -0.0044 0.1819 10.586  1 3.45e-26 ***
## KDM6A_0.1         Genes  1.9404  0.0107 0.1355 14.323  1 1.56e-46 ***
## KMT2C_0.1         Genes  2.4139  0.4841 0.6457  3.739  1 1.85e-04 ***
## KRAS_0.1          Genes  1.8253 -0.1044 0.1565 11.665  1 1.93e-31 ***
## NF1_0.1           Genes  1.8627 -0.0670 0.1522 12.238  1 1.94e-34 ***
## PHF6_0.1          Genes  2.1738  0.2441 0.1301 16.706  1 1.19e-62 ***
## PTPN11_0.1        Genes  2.5509  0.6212 0.2150 11.867  1 1.76e-32 ***
## RUNX1_0.1         Genes  0.7839 -1.1458 0.1361  5.761  1 8.38e-09 ***
## SF3B1_0.1         Genes  3.1354  1.2057 0.3087 10.156  1 3.11e-24 ***
## SRSF2_0.1         Genes  1.3985 -0.5312 0.1706  8.196  1 2.49e-16 ***
## TET2_0.1          Genes  0.6793 -1.2504 0.2014  3.373  1 7.43e-04 ***
## TP53_0.1          Genes  4.8882  2.9585 0.4224 11.572  1 5.69e-31 ***
## U2AF1_0.1         Genes  3.9699  2.0402 0.3601 11.024  1 2.94e-28 ***
## age_10     Demographics -0.0869 -0.0869 0.0996 -0.872  1 3.83e-01
## gender     Demographics -0.0443 -0.0443 0.1112 -0.399  1 6.90e-01
```

```
survConcordance(fitWeightedToronto$surv ~ fitWeightedToronto$linear.predictors, we
ights=weights[cohort=="Toronto"])
```

```
## Call:
## survConcordance(formula = fitWeightedToronto$surv ~ fitWeightedToronto$linear.p
redictors,
##     weights = weights[cohort == "Toronto"])
##
##   n= 505
## Concordance= 0.7739557 se= 0.03055735
## concordant discordant  tied.risk  tied.time   std(c-d)
##  4719299.0  1378335.7        0.0        1.0   372655.1
```

Uno's estimator of cumulative/dynamic AUC

```
a <- AUC.uno(torontoSurv, torontoSurv, fitWeightedToronto$linear.predictors, times
= seq(0,12, 0.1))
round(a$iauc, digits = 3)
```

```
## [1] 0.761
```

```
png("./figures/DC.adj.coxph.auc.uno.png", width = 14, height = 14, units = "cm", r
es = 800)
par(mar = c(4, 4, 4, 2) + 0.1, mgp=c(2.7,0.7,0), bty="L",  tcl =-0.2, las = 1, cex
.lab = 1.1)
plot(a$times, a$auc, xlab="Time (years)", ylab="AUC", pch=16, col="grey80", ylim =
c(0,1.0))
lines(a$times, predict(loess(a$auc ~ a$times, span=0.25)))
abline(h=a$iauc, lty = 3, lwd = 1)
mtext("Adjusted Cox PH model DC", font= 2, side = 3, cex = 1, line = 0.5)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(a$iauc,2
)))
dev.off()
```

```
## pdf
##   2
```

Time-dependent ROC AUC

```
r <- survivalROC(Stime = torontoSurv[,1], status=torontoSurv[,2], marker=fitWeight
edToronto$linear.predictors-colMeans(fitWeightedToronto$Z) %*% fitWeightedToronto$
coefficients, predict.time = 10, method="NNE", span=0.001)
round(r$AUC, digits = 3)
```

```
## [1] 0.783
```

```
png("./figures/DC.adj.coxph.roct.png", width = 14, height = 14, units = "cm", res
= 800)
par(mar = c(4, 4, 4, 2) + 0.1, mgp=c(2.7,0.7,0), bty="L",  tcl =-0.2, las = 1, cex
.lab = 1.1)
plot(r$FP, r$TP, type='s',
     xlab="False Positive Rate", ylab="True Positive Rate",
     col = "black")
mtext("Adjusted Cox PH model DC", font= 2, side = 3, cex = 1, line = 0.5)
abline(a = 0, b = 1, col = "grey70", lty = 1, lwd = 1)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(r$AUC,2)
))
dev.off()
```

```
## pdf
##   2
```

7.2 Validation cohort

7.2.1 Non-adjusted

```
fitSanger <- CoxRFX(sangerX, sangerSurv, groups=sangerGroups, which.mu=which.mu, n
u=nu, sigma0=sigma0)
waldSanger <- WaldTest(fitSanger)
```

```
##                    group     coef   coef-mu      sd       z df  p.value sig
## ASXL1_0.1          Genes  0.76929  0.138331 0.11468  6.7084  1 1.97e-11 ***
## CBL_0.1            Genes  0.62044 -0.010519 0.09149  6.7814  1 1.19e-11 ***
## DNMT3A_0.1         Genes  0.51590 -0.115058 0.11678  4.4176  1 9.98e-06 ***
## JAK2_0.1           Genes  0.58502 -0.045941 0.10315  5.6716  1 1.42e-08 ***
## KMT2C_0.1          Genes  0.64589  0.014930 0.08616  7.4961  1 6.57e-14 ***
## KMT2D_0.1          Genes  0.50507 -0.125896 0.15209  3.3209  1 8.97e-04 ***
## KRAS_0.1           Genes  0.63604  0.005083 0.08495  7.4876  1 7.02e-14 ***
## NF1_0.1            Genes  0.62556 -0.005397 0.08610  7.2657  1 3.71e-13 ***
## NRAS_0.1           Genes  0.63025 -0.000712 0.08492  7.4214  1 1.16e-13 ***
## RAD21_0.1          Genes  0.62875 -0.002212 0.08524  7.3763  1 1.63e-13 ***
## SF3B1_0.1          Genes  0.62728 -0.003678 0.08572  7.3181  1 2.52e-13 ***
## SRSF2_0.1          Genes  0.58180 -0.049163 0.12680  4.5883  1 4.47e-06 ***
## TET2_0.1           Genes  0.69969  0.068723 0.11185  6.2555  1 3.96e-10 ***
## TP53_0.1           Genes  0.69326  0.062294 0.08559  8.0998  1 5.51e-16 ***
## U2AF1_0.1          Genes  0.70018  0.069214 0.08556  8.1832  1 2.76e-16 ***
## age_10      Demographics  0.10777  0.107774 0.12063  0.8934  1 3.72e-01
## gender      Demographics  0.00589  0.005894 0.10667  0.0553  1 9.56e-01
## systol_100         Blood  0.03002  0.030016 0.04429  0.6777  1 4.98e-01
## diastol_100        Blood  0.04718  0.047181 0.02863  1.6478  1 9.94e-02   .
## bmi_10             Blood  0.14183  0.141832 0.07973  1.7790  1 7.52e-02   .
## cholestl_10        Blood  0.00525  0.005246 0.01501  0.3496  1 7.27e-01
## triglyc            Blood  0.00450  0.004496 0.10599  0.0424  1 9.66e-01
## hdl                Blood -0.09452 -0.094522 0.08059 -1.1729  1 2.41e-01
## ldl                Blood  0.11424  0.114236 0.11019  1.0367  1 3.00e-01
## lym                Blood  0.10961  0.109610 0.10081  1.0872  1 2.77e-01
## mcv_100            Blood -0.01645 -0.016447 0.00817 -2.0136  1 4.41e-02   *
## rdw_10             Blood  0.06116  0.061157 0.01972  3.1015  1 1.93e-03  **
## wbc_10             Blood  0.01499  0.014994 0.04138  0.3623  1 7.17e-01
## plt_100            Blood  0.06837  0.068369 0.09739  0.7020  1 4.83e-01
## hgb_10             Blood  0.04890  0.048900 0.02466  1.9826  1 4.74e-02   *
```

```
survConcordance(sangerSurv ~ fitSanger$linear.predictors)
```

```
## Call:
## survConcordance(formula = sangerSurv ~ fitSanger$linear.predictors)
##
##   n= 445
## Concordance= 0.793915 se= 0.05514512
## concordant discordant  tied.risk  tied.time   std(c-d)
##  5532.0000  1436.0000     0.0000     0.0000   768.5024
```

7.2.2 Adjusted

```
fitWeightedSanger <- CoxRFX(sangerX, sangerSurv, sangerGroups, which.mu=which.mu,
sigma0=sigma0, nu=nu, weights=weights[cohort=="Sanger"])
waldWeightedSanger <- WaldTest(fitWeightedSanger)
```

```
##                    group     coef  coef-mu      sd       z df  p.value sig
## ASXL1_0.1          Genes  2.93589  0.95179 0.45155  6.5018  1 7.93e-11 ***
## CBL_0.1            Genes  0.89451 -1.08959 1.25454  0.7130  1 4.76e-01
## DNMT3A_0.1         Genes  0.80635 -1.17775 0.22686  3.5544  1 3.79e-04 ***
## JAK2_0.1           Genes -0.33650 -2.32060 0.95076 -0.3539  1 7.23e-01
## KMT2C_0.1          Genes  2.07422  0.09012 1.10633  1.8749  1 6.08e-02   .
## KMT2D_0.1          Genes  0.05067 -1.93343 0.81191  0.0624  1 9.50e-01
## KRAS_0.1           Genes  2.45194  0.46784 0.41069  5.9702  1 2.37e-09 ***
## NF1_0.1            Genes  1.54402 -0.44008 0.90581  1.7046  1 8.83e-02   .
## NRAS_0.1           Genes  1.92976 -0.05434 0.37569  5.1366  1 2.80e-07 ***
## RAD21_0.1          Genes  1.75445 -0.22966 0.66215  2.6496  1 8.06e-03  **
## SF3B1_0.1          Genes  1.56640 -0.41770 0.99531  1.5738  1 1.16e-01
## SRSF2_0.1          Genes  1.51230 -0.47181 0.27893  5.4217  1 5.90e-08 ***
## TET2_0.1           Genes  1.31638 -0.66772 0.13659  9.6374  1 5.56e-22 ***
## TP53_0.1           Genes  4.92658  2.94248 0.92037  5.3528  1 8.66e-08 ***
## U2AF1_0.1          Genes  6.33456  4.35046 0.76145  8.3191  1 8.86e-17 ***
## age_10      Demographics  0.03788  0.03788 0.11866  0.3193  1 7.50e-01
## gender      Demographics -0.01411 -0.01411 0.10079 -0.1400  1 8.89e-01
## systol_100         Blood  0.01712  0.01712 0.04486  0.3816  1 7.03e-01
## diastol_100        Blood  0.03900  0.03900 0.02964  1.3156  1 1.88e-01
## bmi_10             Blood  0.15297  0.15297 0.08406  1.8198  1 6.88e-02   .
## cholestl_10        Blood  0.00238  0.00238 0.01544  0.1542  1 8.77e-01
## triglyc            Blood -0.03451 -0.03451 0.11758 -0.2935  1 7.69e-01
## hdl                Blood -0.12128 -0.12128 0.08447 -1.4357  1 1.51e-01
## ldl                Blood  0.13215  0.13215 0.11436  1.1555  1 2.48e-01
## lym                Blood  0.07976  0.07976 0.10326  0.7724  1 4.40e-01
## mcv_100            Blood -0.02401 -0.02401 0.00786 -3.0529  1 2.27e-03  **
## rdw_10             Blood  0.06721  0.06721 0.01666  4.0355  1 5.45e-05 ***
## wbc_10             Blood  0.00757  0.00757 0.04834  0.1567  1 8.76e-01
## plt_100            Blood  0.08415  0.08415 0.09986  0.8427  1 3.99e-01
## hgb_10             Blood  0.03718  0.03718 0.02437  1.5255  1 1.27e-01
```

```
survConcordance(sangerSurv ~ fitWeightedSanger$linear.predictors, weights=weights[
cohort=="Sanger"])
```

```
## Call:
## survConcordance(formula = sangerSurv ~ fitWeightedSanger$linear.predictors,
##     weights = weights[cohort == "Sanger"])
##
##   n= 445
## Concordance= 0.8351691 se= 0.05475847
## concordant discordant  tied.risk  tied.time   std(c-d)
##  218019.86   43028.90       0.00       0.00   28589.26
```

Uno’s estimator of cumulative/dynamic AUC

```
w <- c(which(sangerSurv[,1]==0)[-1]-1, nrow(sangerSurv))  #get right censored surv
ival data for each individual
s <- Surv(sangerSurv[w,2], sangerSurv[w,3])  ##Adjust according to dimensions of s
urvival object
a <- AUC.uno(s, s, fitWeightedSanger$linear.predictors[w], times= c(0, 22, 0.1))
round(a$iauc, digits = 3)
```

```
## [1] 0.811
```

Time-dependent ROC AUC

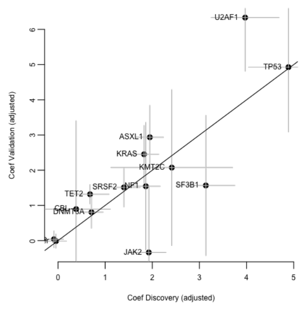
```
r <- survivalROC(Stime = s[,1], status=s[,2], marker=fitWeightedSanger$linear.pred
ictors[w]-colMeans(fitWeightedSanger$Z[w,]) %*% fitWeightedSanger$coefficients, pr
edict.time = 10, method="NNE", span=0.001)
round(r$AUC, digits = 3)
```

```
## [1] 0.737
```

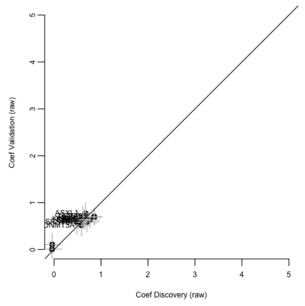
```
png("./figures/VC.ajd.coxph.roct.png", width = 14, height = 14, units = "cm", res
= 500)
par(mar = c(4, 4, 4, 2) + 0.1, mgp=c(2.7,0.7,0), bty="L",  tcl =-0.2, las = 1, cex
.lab = 1.1)
plot(r$FP, r$TP, type='s',
     xlab="False Positive Rate", ylab="True Positive Rate",
     col = "black")
mtext("Adjusted Cox PH VC", font= 2, side = 3, cex = 1, line = 0.5)
abline(a = 0, b = 1, col = "grey70", lty = 1, lwd = 1)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(r$AUC,2)
))
dev.off()
```

```
## pdf
##   2
```

```
i <- intersect(rownames(waldWeightedSanger), rownames(waldWeightedToronto))
plot( waldWeightedToronto[i,"coef"], waldWeightedSanger[i, "coef"], xlab="Coef Dis
covery (adjusted)", ylab="Coef Validation (adjusted)", pch=19, cex=1)
segments(waldWeightedToronto[i,"coef"]  - 2*waldWeightedToronto[i,"sd"], waldWeigh
tedSanger[i, "coef"], waldWeightedToronto[i,"coef"]  + 2*waldWeightedToronto[i,"sd
"], waldWeightedSanger[i, "coef"], col="grey" )
segments(waldWeightedToronto[i,"coef"]  , waldWeightedSanger[i, "coef"]-  2*waldWe
ightedSanger[i,"sd"], waldWeightedToronto[i,"coef"] , waldWeightedSanger[i, "coef"
] +2*waldWeightedSanger[i,"sd"], col="grey")
text(labels=sub("_.+","", i), waldWeightedToronto[i,"coef"], waldWeightedSanger[i,
"coef"], pos=2, adj=c(0,1))
abline(0,1)
```



```
plot( waldToronto[i,"coef"], waldSanger[i, "coef"], xlab="Coef Discovery (raw)", y
lab="Coef Validation (raw)", pch=19, cex=1, ylim=c(0,5),xlim=c(0,5))
segments(waldToronto[i,"coef"]  - 2*waldToronto[i,"sd"], waldSanger[i, "coef"], wa
ldToronto[i,"coef"]  + 2*waldToronto[i,"sd"], waldSanger[i, "coef"], col="grey" )
segments(waldToronto[i,"coef"]  , waldSanger[i, "coef"]-  2*waldSanger[i,"sd"], wa
ldToronto[i,"coef"] , waldSanger[i, "coef"] +2*waldSanger[i,"sd"], col="grey")
text(labels=sub("_.+","", i), waldToronto[i,"coef"], waldSanger[i, "coef"], pos=2,
adj=c(0,1))
abline(0,1)
```



7.3 Cross-validation

7.3.1 Non-adjusted

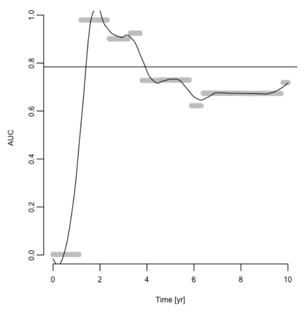
```
sangerImp <- torontoX[1:nrow(sangerX),]
sangerImp[,] <- NA
i <- intersect(names(sangerX),colnames(torontoX))
sangerImp[,i] <- sangerX[,i]
j <- setdiff(names(torontoX)[torontoGroups=="Genes"], names(sangerX))
sangerImp[,j] <- 0
```

DC fit, VC data

```
pS <- PredictRiskMissing(fitToronto, sangerImp)
survConcordance(sangerSurv ~ pS[,1])
```

```
## Call:
## survConcordance(formula = sangerSurv ~ pS[, 1])
##
##   n= 445
## Concordance= 0.7963548 se= 0.05514445
## concordant discordant  tied.risk  tied.time   std(c-d)
##   5545.000   1415.000      8.000      0.000    768.493
```

```
w <- c(which(sangerSurv[,1]==0)[-1]-1, nrow(sangerSurv))
s <- Surv(sangerSurv[w,2], sangerSurv[w,3])
t <- seq(0,10,0.1)
a <- AUC.uno(torontoSurv, s, pS[w,1], times=t)
plot(a$times, a$auc, xlab="Time [yr]", ylab="AUC", pch=16, col='grey')
lines(a$times, predict(loess(a$auc ~ a$times, span=0.25)))
abline(h=a$iauc)
```



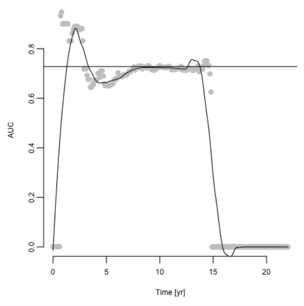
```
torontoImp <- sangerX[1:nrow(torontoX),]
torontoImp[,] <- NA
i <- intersect(names(sangerX),colnames(torontoX))
torontoImp[,i] <- torontoX[,i]
j <- setdiff(names(sangerX)[sangerGroups=="Genes"], names(torontoX))
torontoImp[,j] <- 0
```

VC fit, DC data

```
pT <- PredictRiskMissing(fitSanger, torontoImp)
survConcordance(torontoSurv ~ pT[,1])
```

```
## Call:
## survConcordance(formula = torontoSurv ~ pT[, 1])
##
##   n= 505
## Concordance= 0.6992477 se= 0.03079247
## concordant discordant  tied.risk  tied.time   std(c-d)
##  27235.000  11714.000      0.000      1.000   2398.672
```

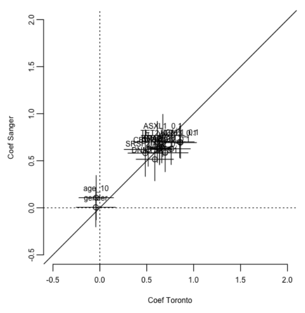
```
t <- seq(0,22,0.1)
a <- AUC.uno(s, torontoSurv, pT[,1], times=t)
plot(a$times, a$auc, xlab="Time [yr]", ylab="AUC", pch=16, col='grey')
lines(a$times, predict(loess(a$auc ~ a$times, span=0.25)))
abline(h=a$iauc)
```



```
sangerM <- sangerX
sangerM[,sangerGroups=="Blood"] <- NA
p <- PredictRiskMissing(fitSanger, sangerM)
survConcordance(sangerSurv ~ p[,1])
```

```
## Call:
## survConcordance(formula = sangerSurv ~ p[, 1])
##
##   n= 445
## Concordance= 0.8069747 se= 0.05514449
## concordant discordant  tied.risk  tied.time   std(c-d)
##  5619.0000  1341.0000     8.0000     0.0000   768.4936
```

```
plot(waldToronto[i,"coef"], waldSanger[i,"coef"], xlab="Coef Toronto", ylab="Coef
Sanger", xlim=c(-0.5,2), ylim=c(-0.5,2))
text(labels=i,waldToronto[i,"coef"], waldSanger[i,"coef"], pos=3)
segments(x0=waldToronto[i,"coef"], x1=waldToronto[i,"coef"], y0= waldSanger[i,"coe
f"]-1.96*waldSanger[i,"sd"], y1=waldSanger[i,"coef"]+1.96*waldSanger[i,"sd"])
segments(x0=waldToronto[i,"coef"]-1.96*waldToronto[i,"sd"], x1=waldToronto[i,"coef
"]+1.96*waldToronto[i,"sd"], y0= waldSanger[i,"coef"], y1=waldSanger[i,"coef"])
abline(0,1)
abline(h=0, lty=3)
abline(v=0, lty=3)
```



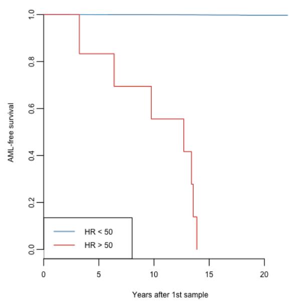
7.3.2 Adjusted

DC fit, VC data

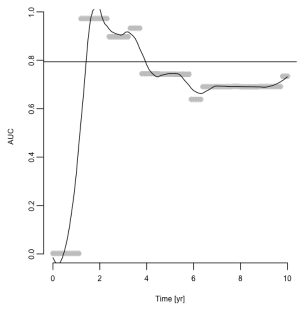
```
pS <- PredictRiskMissing(fitWeightedToronto, sangerImp)
survConcordance(sangerSurv ~ pS[,1], weights=weights[cohort=="Sanger"])
```

```
## Call:
## survConcordance(formula = sangerSurv ~ pS[, 1], weights = weights[cohort ==
##     "Sanger"])
##
##   n= 445
## Concordance= 0.821456 se= 0.05475772
##  concordant  discordant   tied.risk   tied.time    std(c-d)
## 214281.1753  46449.8206    317.7601      0.0000  28588.8682
```

```
m <- as.numeric(colSums(fitWeightedToronto$Z * weights[cohort=="Toronto"])/sum(wei
ghts[cohort=="Toronto"])) %*% coef(fitWeightedToronto)
plot(survfit(sangerSurv ~ exp(pS[,1]-as.numeric(m))>50, weights=weights[cohort=="S
anger"]), col=set1[2:1], ylab="AML-free survival", xlab='Years after 1st sample')
legend("bottomleft", c("HR < 50", "HR > 50"), lty=1, col=set1[2:1])
```



```
w <- c(which(sangerSurv[,1]==0)[-1]-1, nrow(sangerSurv))
s <- Surv(sangerSurv[w,2], sangerSurv[w,3])
t <- seq(0,10,0.1)
a <- AUC.uno(torontoSurv, s, pS[w,1], times=t)
plot(a$times, a$auc, xlab="Time [yr]", ylab="AUC", pch=16, col='grey')
lines(a$times, predict(loess(a$auc ~ a$times, span=0.25)))
abline(h=a$iauc)
```



```
png("./figures/DCfit.VCdata.adj.coxph.auc.uno.png", width = 14, height = 14, units
= "cm", res = 500)
par(mar = c(4, 4, 4, 2) + 0.1, mgp=c(2.7,0.7,0), bty="L",  tcl =-0.2, las = 1, cex
.lab = 1.1)
plot(a$times, a$auc, xlab="Time (years)", ylab="AUC", pch=16, col="grey80", ylim =
c(0,1.0))
lines(a$times, predict(loess(a$auc ~ a$times, span=0.25)))
abline(h=a$iauc, lty = 3, lwd = 1)
mtext("DC fit, VC data", font= 2, side = 3, cex = 1, line = 0.5)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(a$iauc,2
)))
dev.off()
```

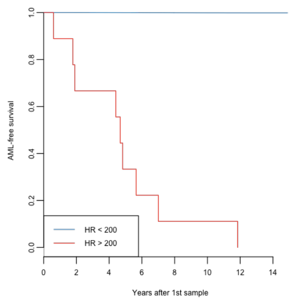
```
## pdf
##   2
```

VC fit, DC data

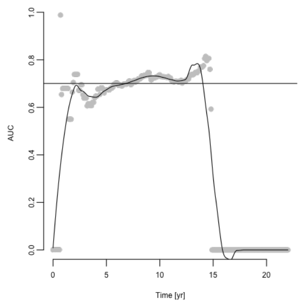
```
pT <- PredictRiskMissing(fitWeightedSanger, torontoImp)
survConcordance(torontoSurv ~ pT[,1], weights=weights[cohort=="Toronto"])
```

```
## Call:
## survConcordance(formula = torontoSurv ~ pT[, 1], weights = weights[cohort ==
##     "Toronto"])
##
##   n= 505
## Concordance= 0.7202544 se= 0.03055735
## concordant discordant  tied.risk  tied.time   std(c-d)
##  4391848.0  1705786.7        0.0        1.0   372655.1
```

```
m <- as.numeric(colSums(fitWeightedSanger$Z * weights[cohort=="Sanger"])/sum(weigh
ts[cohort=="Sanger"])) %*% coef(fitWeightedSanger)
plot(survfit(torontoSurv ~ exp(pT[,1]-as.numeric(m))>200, weights=weights[cohort==
"Toronto"]), col=set1[2:1], ylab="AML-free survival", xlab='Years after 1st sample
')
legend("bottomleft", c("HR < 200", "HR > 200"), lty=1, col=set1[2:1])
```



```
t <- seq(0,22,0.1)
a <- AUC.uno(s, torontoSurv, pT[,1], times=t)
plot(a$times, a$auc, xlab="Time [yr]", ylab="AUC", pch=16, col='grey')
lines(a$times, predict(loess(a$auc ~ a$times, span=0.25)))
abline(h=a$iauc)
```



```
png("./figures/VCfit.DCdata.adj.coxph.auc.uno.png", width = 14, height = 14, units
= "cm", res = 500)
par(mar = c(4, 4, 4, 2) + 0.1, mgp=c(2.7,0.7,0), bty="L",  tcl =-0.2, las = 1, cex
.lab = 1.1)
plot(a$times, a$auc, xlab="Time (years)", ylab="AUC", pch=16, col="grey80", ylim =
c(0,1.0))
lines(a$times, predict(loess(a$auc ~ a$times, span=0.25)))
abline(h=a$iauc, lty = 3, lwd = 1)
mtext("VC fit, DC data", font= 2, side = 3, cex = 1, line = 0.5)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(a$iauc,2
)))#dev.off()
dev.off()
```

```
## pdf
##   2
```

7.4 Combined

7.4.1 Non-adjusted

```
fitAll <- CoxRFX(allX, allSurv, allGroups, which.mu=which.mu, sigma0=sigma0, nu=nu
)
fitAll
```

```
## Means:
##              mean    sd  z   p.val sig
## Genes        0.79 0.068 12 3.9e-31 ***
## Demographics 0.00 0.000  0      NA
##
## Variances - p-values only indicative:
##              sigma2 chisq  df   p.val sig
## Genes          0.19    25 9.2 2.7e-03  **
## Demographics   0.48    25 2.7 1.2e-05 ***
##
## Partial log hazard:
##              Cov[g,g] Sum(Cov[,g])   MSE
## Genes            0.40         0.41 0.012
## Demographics     0.45         0.46 0.032
## TOTAL             NaN         0.88 0.044
```

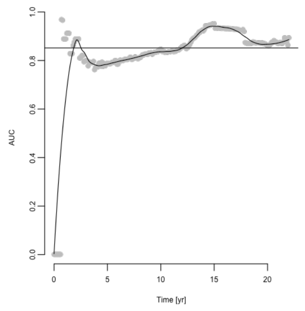
```
WaldTest(fitAll, uncentered=FALSE)
```

```
##                         group      coef coef-mu      sd       z df  p.value sig
## ASXL1_0.1               Genes -0.042129 -0.8326 0.12580 -0.3349  1 7.38e-01
## BCOR_0.1                Genes  0.018602 -0.7719 0.00792  2.3484  1 1.89e-02   *
## CBL_0.1                 Genes -0.313214 -1.1037 0.20346 -1.5394  1 1.24e-01
## DNMT3A_0.1              Genes -0.233727 -1.0242 0.10840 -2.1561  1 3.11e-02   *
## IDH1_0.1                Genes  0.021937 -0.7685 0.20020  0.1096  1 9.13e-01
## IDH2_0.1                Genes -0.278283 -1.0687 0.15309 -1.8177  1 6.91e-02   .
## JAK2_0.1                Genes -0.030573 -0.8210 0.14841 -0.2060  1 8.37e-01
## KDM6A_0.1               Genes  0.000538 -0.7899 0.00638  0.0843  1 9.33e-01
## KMT2C_0.1               Genes  0.068877 -0.7216 0.08598  0.8011  1 4.23e-01
## KMT2D_0.1               Genes -0.391241 -1.1817 0.20457 -1.9125  1 5.58e-02   .
## KRAS_0.1                Genes  0.006235 -0.7842 0.01271  0.4907  1 6.24e-01
## NF1_0.1                 Genes -0.020208 -0.8107 0.03223 -0.6270  1 5.31e-01
## NRAS_0.1                Genes  0.034555 -0.7559 0.01285  2.6887  1 7.17e-03  **
## PHF6_0.1                Genes  0.016466 -0.7740 0.01532  1.0749  1 2.82e-01
## PTPN11_0.1              Genes  0.360022 -0.4304 0.20817  1.7295  1 8.37e-02   .
## RAD21_0.1               Genes -0.006662 -0.7971 0.01823 -0.3654  1 7.15e-01
## RUNX1_0.1               Genes -0.399568 -1.1900 0.11410 -3.5019  1 4.62e-04 ***
## SF3B1_0.1               Genes  0.239576 -0.5509 0.20922  1.1451  1 2.52e-01
## SRSF2_0.1               Genes -0.290822 -1.0813 0.13577 -2.1420  1 3.22e-02   *
## TET2_0.1                Genes -0.158347 -0.9488 0.10442 -1.5165  1 1.29e-01
## TP53_0.1                Genes  0.686128 -0.1043 0.19933  3.4423  1 5.77e-04 ***
## U2AF1_0.1               Genes  0.711837 -0.0786 0.19998  3.5595  1 3.72e-04 ***
## age_10           Demographics -0.034319 -0.0343 0.10560 -0.3250  1 7.45e-01
## gender           Demographics -0.096757 -0.0968 0.18251 -0.5302  1 5.96e-01
## cohort           Demographics -1.297202 -1.2972 0.24120 -5.3781  1 7.53e-08 ***
## mu.Genes                   NA  0.790457      NA      NA      NA  1       NA
## mu.Demographics            NA  0.000000      NA      NA      NA  1       NA
```

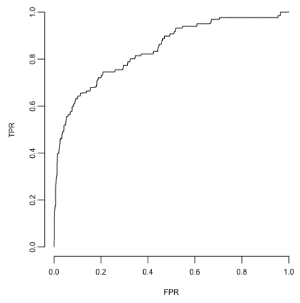
```
survConcordance(allSurv ~ fitAll$linear.predictors)
```

```
## Call:
## survConcordance(formula = allSurv ~ fitAll$linear.predictors)
##
##   n= 950
## Concordance= 0.8059859 se= 0.02746324
## concordant discordant  tied.risk  tied.time   std(c-d)
##  61799.000  14873.000      8.000      1.000   4211.763
```

```
w <- c(which(allSurv[,1]==0)[-1]-1, nrow(allSurv))
s <- Surv(allSurv[w,2], allSurv[w,3])
t <- seq(0,22,0.1)
a <- AUC.uno(s, s, fitAll$linear.predictors[w], times=t)
plot(a$times, a$auc, xlab="Time [yr]", ylab="AUC", pch=16, col='grey')
lines(a$times, predict(loess(a$auc ~ a$times, span=0.25)))
abline(h=a$iauc)
```



```
r <- survivalROC(Stime = s[,1], status=s[,2], marker=fitAll$linear.predictors[w]-c
olMeans(fitAll$Z[w,]) %*% fitAll$coefficients, predict.time = 10, method="NNE", sp
an=0.001)
plot(r$FP, r$TP, type='s', xlab="FPR", ylab="TPR")
```



```
round(r$AUC, 3)
```

```
## [1] 0.84
```

7.4.2 Adjusted

```
fitWeighted <- CoxRFX(allX, allSurv, allGroups, which.mu=which.mu, sigma0=sigma0,
nu=nu, weights=weights)
waldWeighted <- WaldTest(fitWeighted)
```

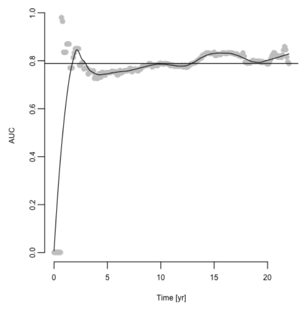
```
##                    group    coef coef-mu     sd      z df  p.value sig
## ASXL1_0.1          Genes  1.9907  0.0666 0.1328 14.985  1 9.18e-51 ***
## BCOR_0.1           Genes  2.1375  0.2134 0.1144 18.677  1 7.57e-78 ***
## CBL_0.1            Genes  0.3984 -1.5256 0.3634  1.096  1 2.73e-01
## DNMT3A_0.1         Genes  0.6589 -1.2652 0.1112  5.926  1 3.10e-09 ***
## IDH1_0.1           Genes  2.4306  0.5065 0.3313  7.337  1 2.18e-13 ***
## IDH2_0.1           Genes  0.8422 -1.0818 0.2181  3.862  1 1.13e-04 ***
## JAK2_0.1           Genes  1.8770 -0.0471 0.1954  9.607  1 7.44e-22 ***
## KDM6A_0.1          Genes  1.9370  0.0129 0.1241 15.607  1 6.51e-55 ***
## KMT2C_0.1          Genes  2.3674  0.4434 0.7114  3.328  1 8.75e-04 ***
## KMT2D_0.1          Genes  0.1632 -1.7609 0.4835  0.338  1 7.36e-01
## KRAS_0.1           Genes  1.9831  0.0590 0.1706 11.622  1 3.20e-31 ***
## NF1_0.1            Genes  1.5839 -0.3402 0.4410  3.592  1 3.29e-04 ***
## NRAS_0.1           Genes  2.3167  0.3926 0.1248 18.569  1 5.76e-77 ***
## PHF6_0.1           Genes  2.2266  0.3025 0.1241 17.937  1 6.04e-72 ***
## PTPN11_0.1         Genes  2.1631  0.2390 0.3107  6.962  1 3.35e-12 ***
## RAD21_0.1          Genes  1.8365 -0.0876 0.2512  7.311  1 2.65e-13 ***
## RUNX1_0.1          Genes  0.8106 -1.1134 0.1329  6.098  1 1.08e-09 ***
## SF3B1_0.1          Genes  3.1070  1.1829 0.3114  9.977  1 1.92e-23 ***
## SRSF2_0.1          Genes  1.3684 -0.5557 0.1491  9.176  1 4.47e-20 ***
## TET2_0.1           Genes  0.9527 -0.9714 0.1172  8.126  1 4.45e-16 ***
## TP53_0.1           Genes  5.0534  3.1293 0.3907 12.934  1 2.88e-38 ***
## U2AF1_0.1          Genes  4.1247  2.2006 0.3300 12.498  1 7.67e-36 ***
## age_10      Demographics -0.0962 -0.0962 0.0863 -1.114  1 2.65e-01
## gender      Demographics -0.0522 -0.0522 0.1044 -0.499  1 6.17e-01
## cohort      Demographics  0.0499  0.0499 0.0973  0.512  1 6.08e-01
```

```
survConcordance(fitWeighted$surv ~ fitWeighted$linear.predictor, weights=weights)
```

```
## Call:
## survConcordance(formula = fitWeighted$surv ~ fitWeighted$linear.predictor,
##     weights = weights)
##
##   n= 950
## Concordance= 0.7778849 se= 0.02802535
##   concordant   discordant    tied.risk    tied.time     std(c-d)
## 6313552.2348 1802641.1313     317.7601       1.0000  454936.0746
```

Dynamic/cumulative AUC

```
w <- c(which(allSurv[,1]==0)[-1]-1, nrow(allSurv))
survAll2 <- Surv(allSurv[w,2], allSurv[w,3])
t <- seq(0,22,0.1)
a <- AUC.uno(survAll2, survAll2, fitWeighted$linear.predictor[w], times=t)
plot(a$times, a$auc, xlab="Time [yr]", ylab="AUC", pch=16, col='grey')
lines(a$times, predict(loess(a$auc ~ a$times, span=0.25)))
abline(h=a$iauc)
```



```
round(a$iauc, 3)
```

```
## [1] 0.789
```

```
png("./figures/combined.ajd.coxph.auc.uno.png", width = 14, height = 14, units = "
cm", res = 500)
par(mar = c(4, 4, 4, 2) + 0.1, mgp=c(2.7,0.7,0), bty="L",  tcl =-0.2, las = 1, cex
.lab = 1.1)
plot(a$times, a$auc, xlab="Time (years)", ylab="AUC", pch=16, col="grey80", ylim =
c(0,1.0))
lines(a$times, predict(loess(a$auc ~ a$times, span=0.25)))
abline(h=a$iauc, lty = 3, lwd = 1)
mtext("Combined adjusted Cox PH", font= 2, side = 3, cex = 1, line = 0.5)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(a$iauc,2
)))
dev.off()
```

```
## pdf
##   2
```

Time-depenent ROC

```
r <- survivalROC(Stime = survAll2[,1], status=survAll2[,2], marker=fitWeighted$lin
ear.predictors[w]-colMeans(fitWeighted$Z[w,]) %*% fitWeighted$coefficients, predic
t.time = 10, method="NNE", span=0.001)
round(r$AUC, 3)
```

```
## [1] 0.791
```

```
png("./figures/Combined.adj.coxph.roct.png", width = 14, height = 14, units = "cm"
, res = 500)
par(mar = c(4, 4, 4, 2) + 0.1, mgp=c(2.7,0.7,0), bty="L",  tcl =-0.2, las = 1, cex
.lab = 1.1)
plot(r$FP, r$TP, type='s',
     xlab="False Positive Rate", ylab="True Positive Rate",
     col = "black")
mtext("Combined adjusted Cox PH", font= 2, side = 3, cex = 1, line = 0.5)
abline(a = 0, b = 1, col = "grey70", lty = 1, lwd = 1)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(r$AUC,2)
))
dev.off()
```

```
## pdf
##   2
```

7.4.3 Bootstrap

```
coefWeightedBoot <- sapply(1:100, function(foo){
            set.seed(foo)
            b <- unique(sample(1:nrow(allX), replace=TRUE))
            fitWeighted <- CoxRFX(allX[b,], allSurv[b,], allGroups, which.mu=which
.mu, sigma0=sigma0, nu=5, weights=weights[b])
            c(coef(fitWeighted), 'mu.Genes'=fitWeighted$mu["Genes"])
        })
```

```
concBoots <- sapply(1:100, function(foo){
            set.seed(foo)
            b <- unique(sample(1:nrow(allX), replace=TRUE))
            oob <- !1:nrow(allX) %in% b
            c(inb=as.numeric(survConcordance(allSurv[b,]~ as.matrix(allX)[b,] %*%
coefWeightedBoot[-26,foo], weights=weights[b])$concordance),
                    oob=as.numeric(survConcordance(allSurv[oob,]~ as.matrix(allX)[
oob,] %*% coefWeightedBoot[-26,foo],weights=weights[oob])$concordance),
                    auc = AUC.uno(survAll2[oob[w],], survAll2[oob[w],], as.matrix(
allX)[w,][oob[w],] %*% coefWeightedBoot[-26,foo], times=t)$iauc
            )
        })

apply(concBoots,1,quantile)
```

```
##            inb       oob       auc
## 0%   0.7127155 0.6414249 0.6163769
## 25%  0.7623231 0.7282023 0.7333587
## 50%  0.7757864 0.7643297 0.7833229
## 75%  0.7985773 0.7875492 0.8223659
## 100% 0.8519811 0.8713292 0.8805585
```

7.4.4 Forest plot

Figure 2

```
png("./figures/Combined.adj.coxph.boostrapped.forest.png", width = 18.5, height =
19, units = "cm", res = 800)
par(bty="n", mar=c(3,6,3,15)+.5, mgp=c(2,0.5,0), xpd=FALSE, tcl=-.25)
c <- c(waldWeighted[-25,"coef"], "mu"=fitWeighted$mu["Genes"]); names(c)[1:24] <-
rownames(waldWeighted)[-25]
o <- c(23:24,1:22,25)
s <- c(rep(1,2), rep(.5, 23))
c <- exp(c*c(rep(0.5,22), c(1,1),0.5))

ci <- apply(coefWeightedBoot,1,quantile, c(0.025,0.975))[,-25] * rep(c(rep(0.5,22)
, c(1,1),0.5), each=2)
y <- rev(seq_along(c))
plot(c[o], y, xlab="Hazard ratio", log='x', ylab='', xaxt = "n", yaxt="n", pch=NA,
xlim=c(0.5,50))
atx <- axTicks(1)
axis(1,at=atx,labels=atx)
abline(h=y, col="#EEEEEE", lty=1)
abline(v=1, lty=1, col="grey")
abline(v=c["mu.Genes"], col=mg14::colTrans("#57B2AB"), lty=1)
segments(exp(ci[1,o]), y, exp(ci[2,o]),y)
points(c[o], y, xlab="Relative risk",  bg=set1[3], cex=2, pch=c(rep(21,24), 23))
m <- match(names(c)[o],rownames(waldWeightedToronto))[-25]
points(exp(c(waldWeightedToronto$coef[m], fitWeightedToronto$mu["Genes"])*s), y,bg
=set1[4], pch=c(rep(21,24), 23), cex=1)
m <- match(names(c)[o],rownames(waldWeightedSanger))[-25]
points(exp(c(waldWeightedSanger$coef[m], fitWeightedSanger$mu["Genes"])*s), y,bg=s
et1[5], pch=c(rep(21,24), 23), cex=1)
mtext(side=2, sub("mu.Genes","Av. gene", sub("_.+","", sub("age", "Age", sub("gend
er", "Gender", names(c)[o])))), at=y, las=2, font=c(1,1,rep(3,22),1))
r <- sapply(split(as.data.frame(allX>0), control), colMeans)
f <- sapply(split(allX, control), apply, 2, function(x) mean(x[x>0]))
par(xpd=NA)
points(rep(100,22),y[3:24], cex=sqrt(r[o[3:24],2]*10), pch=21, bg=set1[2])
points(rep(100*1.5,22), y[3:24], cex=sqrt(r[o[3:24],1]*10), pch=21, bg=set1[1])
points(rep(360,22),y[3:24], cex=sqrt(f[o[3:24],2]), pch=21, bg=set1[2])
points(rep(360*1.5,22), y[3:24], cex=sqrt(f[o[3:24],1]), pch=21, bg=set1[1])
legend(x=0.5, y=28, pch=21, pt.bg=set1[c(4,5,3)], c("DC","VC","Combined"), bty="n"
, ncol=3, text.width=0.35)
text(y=24, x=100, "           Recurrence")
text(y=24, x=360*1.5, "VAF")
axis(1, at=c(100,100*1.5), c("Control ","Pre-AML "), las=2, line=-1)
axis(1, at=c(360,360*1.5), c("Control ","Pre-AML "), las=2, line=-1)
dev.off()
```

```
## pdf
##   2
```

7.4.5 Dichotomous variables

```
allXDich <- allX
allXDich[allGroups=="Genes"] <- (allXDich[allGroups=="Genes"] > 0) + 0
fitWeightedDich <- CoxRFX(allXDich, allSurv, allGroups, which.mu=which.mu, sigma0=
sigma0, nu=nu, weights=weights)

WaldTest(fitWeightedDich)
```

```
##                    group    coef coef-mu     sd       z df  p.value sig
## ASXL1_0.1          Genes  1.3797 -0.3942 0.3175  4.3456  1 1.39e-05 ***
## BCOR_0.1           Genes  2.5308  0.7570 0.8406  3.0106  1 2.61e-03  **
## CBL_0.1            Genes  0.3932 -1.3806 0.4991  0.7879  1 4.31e-01
## DNMT3A_0.1         Genes  0.7794 -0.9944 0.2049  3.8048  1 1.42e-04 ***
## IDH1_0.1           Genes  2.0403  0.2665 0.5817  3.5073  1 4.53e-04 ***
## IDH2_0.1           Genes  3.9907  2.2169 0.5363  7.4414  1 9.96e-14 ***
## JAK2_0.1           Genes  3.2315  1.4577 0.3911  8.2629  1 1.42e-16 ***
## KDM6A_0.1          Genes  0.7396 -1.0343 0.7822  0.9456  1 3.44e-01
## KMT2C_0.1          Genes -0.4630 -2.2368 0.5910 -0.7834  1 4.33e-01
## KMT2D_0.1          Genes  0.8142 -0.9597 0.9409  0.8653  1 3.87e-01
## KRAS_0.1           Genes -0.0209 -1.7948 0.7030 -0.0298  1 9.76e-01
## NF1_0.1            Genes -1.1385 -2.9124 0.8236 -1.3824  1 1.67e-01
## NRAS_0.1           Genes  1.6320 -0.1419 0.7812  2.0891  1 3.67e-02   *
## PHF6_0.1           Genes  4.0915  2.3176 0.7069  5.7883  1 7.11e-09 ***
## PTPN11_0.1         Genes  2.2597  0.4859 0.6548  3.4510  1 5.59e-04 ***
## RAD21_0.1          Genes  1.0923 -0.6816 0.9283  1.1767  1 2.39e-01
## RUNX1_0.1          Genes  2.6557  0.8818 0.5738  4.6284  1 3.69e-06 ***
## SF3B1_0.1          Genes  0.0815 -1.6924 0.6027  0.1352  1 8.92e-01
## SRSF2_0.1          Genes  4.2431  2.4693 0.3084 13.7566  1 4.65e-43 ***
## TET2_0.1           Genes  0.9715 -0.8023 0.2351  4.1328  1 3.58e-05 ***
## TP53_0.1           Genes  2.0033  0.2295 0.4168  4.8067  1 1.53e-06 ***
## U2AF1_0.1          Genes  5.7172  3.9433 0.4178 13.6831  1 1.28e-42 ***
## age_10     Demographics -0.3024 -0.3024 0.0958 -3.1571  1 1.59e-03  **
## gender     Demographics -0.0512 -0.0512 0.1362 -0.3759  1 7.07e-01
## cohort     Demographics  0.2569  0.2569 0.1435  1.7896  1 7.35e-02   .
```

```
survConcordance(allSurv ~ fitWeightedDich$linear.predictors, weights=weights)
```

```
## Call:
## survConcordance(formula = allSurv ~ fitWeightedDich$linear.predictors,
##     weights = weights)
##
##   n= 950
## Concordance= 0.764251 se= 0.02802535
##   concordant   discordant    tied.risk    tied.time     std(c-d)
## 6202805.3608 1913213.1798     492.5856       1.0000  454936.0734
```

7.4.6 Bootstrap adjustment

To compare to the weighted CoxRFX models

```
set.seed(42)

p <- c(rep(n_total_sanger, sum(cohort=="Sanger" & control)), rep(n_total_toronto,
sum(cohort=="Toronto" & control)))
b42 <- c(sample(which(control), size=round(n_total) - sum(!control), prob=p, repla
ce=TRUE), which(!control))

fitBoot <- CoxRFX(allX[b42,], allSurv[b42,], allGroups, which.mu=which.mu, sigma0=
sigma0, nu=nu)

set.seed(42)
b <- c(sample(which( sangerData$Diagnosis=="Control"), size=round(n_total_sanger)
- sum(sangerData$Diagnosis!="Control"), replace=TRUE), which(sangerData$Diagnosis!
="Control"))

fitBootSanger <- CoxRFX(sangerX[b,], sangerSurv[b,], sangerGroups, which.mu=which.
mu, sigma0=sigma0, nu=nu)

survConcordance(fitBootSanger$surv ~ fitBootSanger$linear.predictors)
```

```
## Call:
## survConcordance(formula = fitBootSanger$surv ~ fitBootSanger$linear.predictors)
##
##   n= 10407
## Concordance= 0.8334695 se= 0.05475909
## concordant discordant  tied.risk  tied.time   std(c-d)
##   140833.0    28139.0        0.0        0.0    18505.5
```

```
waldBootSanger <- WaldTest(fitBootSanger)
```

```
##                     group     coef  coef-mu      sd       z df  p.value sig
## ASXL1_0.1           Genes  2.75130  0.85036 0.44987  6.1157  1 9.61e-10 ***
## CBL_0.1             Genes  0.90179 -0.99914 1.17452  0.7678  1 4.43e-01
## DNMT3A_0.1          Genes  0.75840 -1.14254 0.22408  3.3845  1 7.13e-04 ***
## JAK2_0.1            Genes -0.20568 -2.10662 0.92220 -0.2230  1 8.24e-01
## KMT2C_0.1           Genes  2.16912  0.26819 0.96833  2.2401  1 2.51e-02   *
## KMT2D_0.1           Genes  0.06618 -1.83475 0.76576  0.0864  1 9.31e-01
## KRAS_0.1            Genes  2.31066  0.40972 0.38106  6.0638  1 1.33e-09 ***
## NF1_0.1             Genes  1.57512 -0.32581 0.77819  2.0241  1 4.30e-02   *
## NRAS_0.1            Genes  1.84937 -0.05157 0.35761  5.1715  1 2.32e-07 ***
## RAD21_0.1           Genes  1.70593 -0.19501 0.58727  2.9049  1 3.67e-03  **
## SF3B1_0.1           Genes  1.54550 -0.35544 0.87032  1.7758  1 7.58e-02   .
## SRSF2_0.1           Genes  1.40565 -0.49529 0.27962  5.0271  1 4.98e-07 ***
## TET2_0.1            Genes  1.25279 -0.64815 0.13571  9.2317  1 2.66e-20 ***
## TP53_0.1            Genes  4.63845  2.73751 0.89272  5.1959  1 2.04e-07 ***
## U2AF1_0.1           Genes  5.78946  3.88853 0.73724  7.8528  1 4.07e-15 ***
## age_10      Demographics  0.04278  0.04278 0.11873  0.3603  1 7.19e-01
## gender      Demographics -0.01852 -0.01852 0.10088 -0.1836  1 8.54e-01
## systol_100          Blood  0.02344  0.02344 0.04556  0.5145  1 6.07e-01
## diastol_100         Blood  0.04133  0.04133 0.03020  1.3686  1 1.71e-01
## bmi_10              Blood  0.14916  0.14916 0.08426  1.7702  1 7.67e-02   .
## cholestl_10         Blood  0.00303  0.00303 0.01547  0.1958  1 8.45e-01
## triglyc             Blood -0.02770 -0.02770 0.11803 -0.2347  1 8.14e-01
## hdl                 Blood -0.12117 -0.12117 0.08479 -1.4291  1 1.53e-01
## ldl                 Blood  0.13479  0.13479 0.11448  1.1775  1 2.39e-01
## lym                 Blood  0.08408  0.08408 0.10435  0.8057  1 4.20e-01
## mcv_100             Blood -0.02485 -0.02485 0.00798 -3.1160  1 1.83e-03  **
## rdw_10              Blood  0.06629  0.06629 0.01703  3.8934  1 9.88e-05 ***
## wbc_10              Blood  0.01199  0.01199 0.04735  0.2532  1 8.00e-01
## plt_100             Blood  0.09163  0.09163 0.10006  0.9158  1 3.60e-01
## hgb_10              Blood  0.03986  0.03986 0.02497  1.5960  1 1.10e-01
```

```
set.seed(42)
b <- c(sample(which( torontoData$Diagnosis=="Control"), size=round(n_total_toronto
) - sum(torontoData$Diagnosis!="Control"), replace=TRUE), which(torontoData$Diagno
sis!="Control"))

fitBootToronto <- CoxRFX(torontoX[b,], torontoSurv[b,], torontoGroups, which.mu=wh
ich.mu, sigma0=sigma0, nu=nu)
survConcordance(fitBootToronto$surv ~ fitBootToronto$linear.predictors)
```

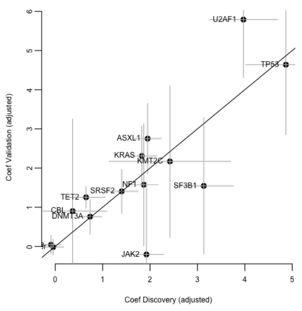
```
## Call:
## survConcordance(formula = fitBootToronto$surv ~ fitBootToronto$linear.predictor
s)
##
##   n= 72378
## Concordance= 0.7750173 se= 0.03055346
## concordant discordant  tied.risk  tied.time   std(c-d)
##  4722585.0  1370937.0        0.0        1.0   372356.4
```

```
waldWeightedToronto <- WaldTest(fitBootToronto)
```

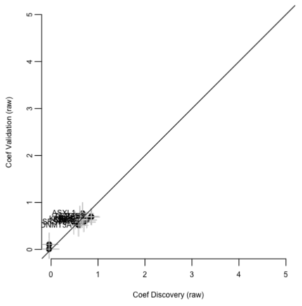
```
##                   group    coef  coef-mu     sd      z df  p.value sig
## ASXL1_0.1         Genes  1.9494  0.01801 0.1451 13.430  1 4.03e-41 ***
## CALR_0.1          Genes  0.9415 -0.98990 0.7233  1.302  1 1.93e-01
## CBL_0.1           Genes  0.3663 -1.56509 0.3604  1.016  1 3.09e-01
## DNMT3A_0.1        Genes  0.7358 -1.19559 0.1243  5.921  1 3.20e-09 ***
## IDH1_0.1          Genes  2.3973  0.46594 0.3355  7.145  1 8.98e-13 ***
## IDH2_0.1          Genes  0.8078 -1.12360 0.2283  3.538  1 4.03e-04 ***
## JAK2_0.1          Genes  1.9240 -0.00738 0.1822 10.562  1 4.49e-26 ***
## KDM6A_0.1         Genes  1.9436  0.01219 0.1340 14.506  1 1.12e-47 ***
## KMT2C_0.1         Genes  2.4194  0.48806 0.6410  3.774  1 1.60e-04 ***
## KRAS_0.1          Genes  1.8282 -0.10316 0.1559 11.725  1 9.46e-32 ***
## NF1_0.1           Genes  1.8677 -0.06366 0.1512 12.353  1 4.69e-35 ***
## PHF6_0.1          Genes  2.1755  0.24415 0.1302 16.711  1 1.08e-62 ***
## PTPN11_0.1        Genes  2.5369  0.60555 0.2217 11.445  1 2.49e-30 ***
## RUNX1_0.1         Genes  0.7795 -1.15181 0.1359  5.738  1 9.57e-09 ***
## SF3B1_0.1         Genes  3.1337  1.20231 0.3091 10.138  1 3.76e-24 ***
## SRSF2_0.1         Genes  1.4023 -0.52910 0.1703  8.235  1 1.80e-16 ***
## TET2_0.1          Genes  0.6503 -1.28104 0.2012  3.232  1 1.23e-03  **
## TP53_0.1          Genes  4.8664  2.93502 0.4220 11.532  1 9.14e-31 ***
## U2AF1_0.1         Genes  3.9705  2.03910 0.3601 11.025  1 2.89e-28 ***
## age_10     Demographics -0.0891 -0.08907 0.0998 -0.892  1 3.72e-01
## gender     Demographics -0.0449 -0.04493 0.1114 -0.403  1 6.87e-01
```

Compare results

```
i <- intersect(rownames(waldBootSanger), rownames(waldWeightedToronto))
plot( waldWeightedToronto[i,"coef"], waldBootSanger[i, "coef"], xlab="Coef Discove
ry (adjusted)", ylab="Coef Validation (adjusted)", pch=19, cex=1)#sqrt(colMeans(rb
ind(sangerX[,i], torontoX[,i])>0)*100))
segments(waldWeightedToronto[i,"coef"]  - 2*waldWeightedToronto[i,"sd"], waldBootS
anger[i, "coef"], waldWeightedToronto[i,"coef"]  + 2*waldWeightedToronto[i,"sd"],
waldBootSanger[i, "coef"], col="grey" )
segments(waldWeightedToronto[i,"coef"]  , waldBootSanger[i, "coef"]-  2*waldBootSa
nger[i,"sd"], waldWeightedToronto[i,"coef"] , waldBootSanger[i, "coef"] +2*waldBoo
tSanger[i,"sd"], col="grey")
text(labels=sub("_.+","", i), waldWeightedToronto[i,"coef"], waldBootSanger[i, "co
ef"], pos=2, adj=c(0,1))
abline(0,1)
```



```
plot( waldToronto[i,"coef"], waldSanger[i, "coef"], xlab="Coef Discovery (raw)", y
lab="Coef Validation (raw)", pch=19, cex=1, ylim=c(0,5),xlim=c(0,5))#sqrt(colMeans
(rbind(sangerX[,i], torontoX[,i])>0)*100))
segments(waldToronto[i,"coef"]  - 2*waldToronto[i,"sd"], waldSanger[i, "coef"], wa
ldToronto[i,"coef"]  + 2*waldToronto[i,"sd"], waldSanger[i, "coef"], col="grey" )
segments(waldToronto[i,"coef"]  , waldSanger[i, "coef"]-  2*waldSanger[i,"sd"], wa
ldToronto[i,"coef"] , waldSanger[i, "coef"] +2*waldSanger[i,"sd"], col="grey")
text(labels=sub("_.+","", i), waldToronto[i,"coef"], waldSanger[i, "coef"], pos=2,
adj=c(0,1))
abline(0,1)
```



7.4.7 LOOCV

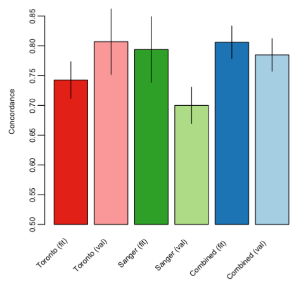
```
samples <- factor(c(as.character(sangerData$Individual), as.character(torontoData$
Sample)))
```

```
looAll <- do.call("rbind",mclapply(levels(samples), function(l){
                    i <- samples!=l
                    f <<- CoxRFX(allX[i,], allSurv[i,], allGroups, which.mu=which.
mu, sigma0=sigma0, nu=nu)
                    p <- as.matrix(allX[!i,,drop=FALSE]) %*% f$coefficients
                    r <- cbind(matrix(f$coefficients, nrow=length(p), ncol=length(
f$coefficients), byrow=TRUE), linear.predictor=p)
                    colnames(r) <- c(names(f$coefficients), "linear.predictor")
                    as.data.frame(r)
                }, mc.cores=4))
looAll <- looAll[order(order(samples)),]
pp <- looAll$linear.predictor

c <- rbind(
        `Toronto (fit)`=as.data.frame(survConcordance(torontoSurv ~ fitToronto$lin
ear.predictors)[c("concordance","std.err")]),
        `Toronto (val)`=as.data.frame(survConcordance(sangerSurv ~ pS[,1])[c("conc
ordance","std.err")]),
        `Sanger (fit)`=as.data.frame(survConcordance(sangerSurv ~ fitSanger$linear
.predictors)[c("concordance","std.err")]),
        `Sanger (val)`=as.data.frame(survConcordance(torontoSurv ~ pT[,1])[c("conc
ordance","std.err")]),
        `Combined (fit)`=as.data.frame(survConcordance(allSurv ~ fitAll$linear.pre
dictors)[c("concordance","std.err")]),
        `Combined (val)`=as.data.frame(survConcordance(allSurv ~ pp)[c("concordanc
e","std.err")]))

c
```

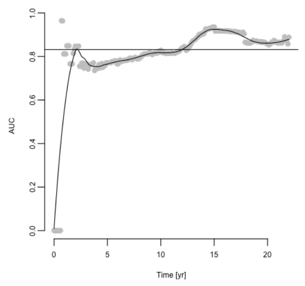
	concordance <dbl>	**std.err** <dbl>
Toronto (fit)	0.7426378	0.03079247
Toronto (val)	0.8069747	0.05514445
Sanger (fit)	0.7939150	0.05514512
Sanger (val)	0.7000180	0.03079247
Combined (fit)	0.8059859	0.02746324
Combined (val)	0.7847548	0.02746328

6 rows

```
par(mar=c(5,3,1,1), mgp=c(2,.5,0))
b <- barplot(c$concordance-0.5, ylab="Concordance", col=rev(RColorBrewer::brewer.p
al(6,"Paired")), ylim=c(0.5,0.88), offset=0.5)
mg14::rotatedLabel(x=b, labels=rownames(c))
segments(b,c$concordance+c$std.err,b,c$concordance-c$std.err)
```



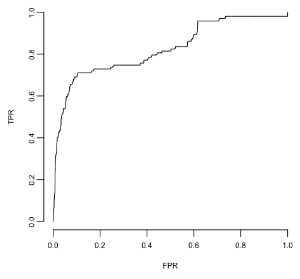
```
w <- c(which(allSurv[,1]==0)[-1]-1, nrow(allSurv))
survAll2 <- Surv(allSurv[w,2], allSurv[w,3])
t <- seq(0,22,0.1)
a <- AUC.uno(survAll2, survAll2, looAll$linear.predictor[w], times=t)
plot(a$times, a$auc, xlab="Time [yr]", ylab="AUC", pch=16, col='grey')
lines(a$times, predict(loess(a$auc ~ a$times, span=0.25)))
abline(h=a$iauc)
```



```
round(a$iauc, 3)
```

```
## [1] 0.832
```

```
r <- survivalROC(Stime = survAll2[,1], status=survAll2[,2], marker=looAll$linear.p
redictor[w], predict.time = 10, method="NNE", span=0.001)#0.25*nrow(s)^(-0.20))
plot(r$FP, r$TP, type='s', xlab="FPR", ylab="TPR")
```

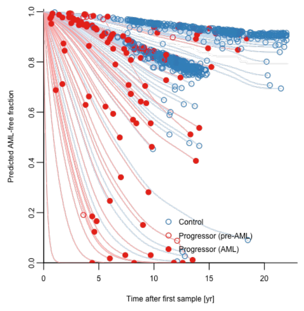


```
round(r$AUC, 3)
```

```
## [1] 0.825
```

7.4.7.1 Individual Predictions (non-adjusted)

```
plot(survfit(allSurv~1), conf.int=FALSE, xlab='Time after first sample [yr]', ylab
='Predicted AML-free fraction', col='white', bty='L', yaxs='i', ylim=c(0,1.01))
d <- data.frame(t=NULL, s=NULL, pch=NULL, col=character())
for(i in unique(samples)){
    km <- exp(predict(smooth.spline(log(summary(survfit(allSurv[samples!=i,]~1), t
imes=t)$surv), df=10))$y)
    l0 <- colMeans(fitAll$Z[samples!=i,,drop=FALSE]) %*% as.numeric(looAll[samples
==i,][1,colnames(fitAll$Z)])
    kmi <- function(km, s, lp, l0){
        .kmi <- function(km, sj, lpj, l0) km[t >= sj[,1] & t <= sj[,2]]^exp(lpj-l0
)
        k0 <- 1
        for(j in 1:nrow(s)) {
            k <- .kmi(km, s[j,], lp[j], l0)
            k <- k * k0/k[1]
            w <- t >= s[j,1] & t <= s[j,2]
            k0 <- k[length(k)]
            c <- if(s[nrow(s),3]==1) set1[1] else set1[2]
            #if(c==set1[1]) next
            lines(t[w], k, col=mg14:::colTrans(c), type='l')
            p <- if(s[j,3]==1) 19 else 1
            #points(t[w][length(k)], k[length(k)], col=c, pch=p)
            d <<- rbind(d, data.frame(t=t[w][length(k)], s=k[length(k)], pch=p, co
l=c))
        }
    }
    kmi(km, allSurv[samples==i,], looAll$linear.predictor[samples==i], l0)
}
points(d$t, d$s, pch=d$pch, col=as.character(d$col))
legend("bottomright", pch=c(1,1,19), col=c(set1[2], set1[1], set1[1]), legend=c("C
ontrol", "Progressor (pre-AML)", "Progressor (AML)"), bty='n')
```



7.4.7.2 Jackknife variance

```
i <- !duplicated(samples)
coef.jack <- colMeans(looAll[i,-ncol(looAll)])
var.jack <- rowSums((t(looAll[i,-ncol(looAll)]) - coef.jack)^2) * (sum(i)-1)/sum(i
)

p.jack <- pchisq(coef.jack^2/var.jack,1, lower.tail=FALSE)

data.frame(coef.jack, p.jack, sig=mg14::sig2star(p.jack), n=colSums(allX[i,]>0))
```

	coef.jack <dbl>	p.jack <dbl>	sig <fctr>	n <dbl>
ASXL1_0.1	0.74835623	1.277998e-05	***	26
BCOR_0.1	0.80859507	2.311062e-04	***	1
CBL_0.1	0.47795378	3.123703e-01		12
DNMT3A_0.1	0.55685260	7.358773e-06	***	194
IDH1_0.1	0.81211760	5.586147e-10	***	3
IDH2_0.1	0.51251777	1.351015e-01		6
JAK2_0.1	0.75979214	3.181470e-08	***	10
KDM6A_0.1	0.79059980	7.666406e-05	***	3
KMT2C_0.1	0.85878619	5.304616e-04	***	6
KMT2D_0.1	0.40005469	3.584861e-01		1

1-10 of 25 rows Previous **1** 2 3 Next

7.4.8 Multiple bootstraps

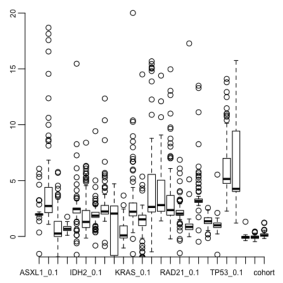
```
save(file="boot.RData", control, allX, allSurv, sigma0, nu, which.mu, allGroups, n
_total, cohort, p)
```

```
fitBoots <- simplify2array(mclapply(1:100, function(foo){
                    set.seed(foo)
                    w <- which(control)
                    s <- sample(seq_along(which(control)), replace=TRUE)
                    b <- c(sample(which(control)[s], size=round(n_total) - sum(!co
ntrol), prob=p[s], replace=TRUE), sample(which(!control), replace=TRUE))
                    fitBoot <- CoxRFX(allX[b,], allSurv[b,], allGroups, which.mu=w
hich.mu, sigma0=sigma0, nu=nu)
                    fitBoot$coefficients
                }, mc.cores=4))
save(fitBoots, file="fitBoots.RData")
```

```
load('fitBoots.RData')

WaldTest(fitBoot)
```

```
##                    group    coef coef-mu     sd      z df  p.value sig
## ASXL1_0.1          Genes  1.9782  0.0682 0.1330 14.873  1 4.90e-50 ***
## BCOR_0.1           Genes  2.1204  0.2104 0.1157 18.319  1 5.81e-75 ***
## CBL_0.1            Genes  0.3747 -1.5352 0.3614  1.037  1 3.00e-01
## DNMT3A_0.1         Genes  0.6499 -1.2600 0.1133  5.735  1 9.77e-09 ***
## IDH1_0.1           Genes  2.4215  0.5116 0.3299  7.341  1 2.12e-13 ***
## IDH2_0.1           Genes  0.8614 -1.0486 0.2191  3.931  1 8.47e-05 ***
## JAK2_0.1           Genes  1.8708 -0.0391 0.1956  9.562  1 1.15e-21 ***
## KDM6A_0.1          Genes  1.9211  0.0112 0.1251 15.363  1 2.92e-53 ***
## KMT2C_0.1          Genes  2.3935  0.4836 0.7067  3.387  1 7.07e-04 ***
## KMT2D_0.1          Genes  0.1309 -1.7790 0.4810  0.272  1 7.86e-01
## KRAS_0.1           Genes  1.9602  0.0503 0.1717 11.415  1 3.53e-30 ***
## NF1_0.1            Genes  1.5704 -0.3396 0.4386  3.580  1 3.43e-04 ***
## NRAS_0.1           Genes  2.3060  0.3960 0.1213 19.014  1 1.31e-80 ***
## PHF6_0.1           Genes  2.2127  0.3028 0.1241 17.835  1 3.80e-71 ***
## PTPN11_0.1         Genes  2.1333  0.2233 0.3110  6.860  1 6.86e-12 ***
## RAD21_0.1          Genes  1.8285 -0.0815 0.2524  7.244  1 4.36e-13 ***
## RUNX1_0.1          Genes  0.8075 -1.1025 0.1325  6.095  1 1.10e-09 ***
## SF3B1_0.1          Genes  3.0963  1.1863 0.3107  9.967  1 2.13e-23 ***
## SRSF2_0.1          Genes  1.3408 -0.5692 0.1503  8.923  1 4.55e-19 ***
## TET2_0.1           Genes  0.9202 -0.9897 0.1179  7.807  1 5.85e-15 ***
## TP53_0.1           Genes  5.0203  3.1104 0.3921 12.803  1 1.57e-37 ***
## U2AF1_0.1          Genes  4.0999  2.1900 0.3306 12.402  1 2.54e-35 ***
## age_10      Demographics -0.0761 -0.0761 0.0912 -0.835  1 4.04e-01
## gender      Demographics -0.0530 -0.0530 0.1157 -0.458  1 6.47e-01
## cohort      Demographics  0.1992  0.1992 0.1103  1.806  1 7.09e-02   .
```

```
boxplot(t(fitBoots), ylim=c(-1,20))
points(fitBoot$coefficiencts, pch="*", col='red')
```



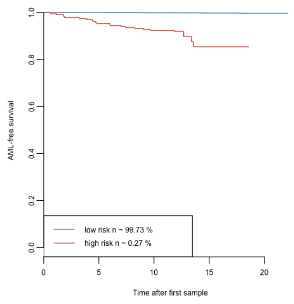
Concordance on out of bag samples

```
concBoots <- sapply(1:100, function(foo){
            set.seed(foo)
            w <- which(control)
            s <- sample(seq_along(which(control)), replace=TRUE)
            b <- c(sample(which(control)[s], size=round(n_total) - sum(!control),
prob=p[s], replace=TRUE), sample(which(!control), replace=TRUE))
            oob <- !1:nrow(allX) %in% b
            oos <- c(sample(which(oob & control), size=round(n_total) - sum(!contr
ol), replace=TRUE), sample(which(oob&!control), size=sum(!control), replace=TRUE))
            c(inb=as.numeric(survConcordance(allSurv[b,]~ as.matrix(allX)[b,] %*%
fitBoots[,foo])$concordance),
                    oob=as.numeric(survConcordance(allSurv[oob,]~ as.matrix(allX)[
oob,] %*% fitBoots[,foo])$concordance),
                    oos=as.numeric(survConcordance(allSurv[oos,]~ as.matrix(allX)[
oos,] %*% fitBoots[,foo])$concordance)
            )
        })
```

```
looAllWeighted <- do.call("rbind",mclapply(levels(samples), function(l){
                    i <- samples!=l
                    f <<- CoxRFX(allX[i,], allSurv[i,], allGroups, which.mu=which.
mu, sigma0=sigma0, nu=nu, weights=weights[i])
                    p <- as.matrix(allX[!i,,drop=FALSE]) %*% f$coefficients
                    r <- cbind(matrix(f$coefficients, nrow=length(p), ncol=length(
f$coefficients), byrow=TRUE), linear.predictor=p)
                    colnames(r) <- c(names(f$coefficients), "linear.predictor")
                    as.data.frame(r)
                }, mc.cores=4))
looAllWeighted <- looAllWeighted[order(order(samples)),]
pp <- looAllWeighted$linear.predictor
survConcordance(allSurv ~ pp, weights=weights)
```

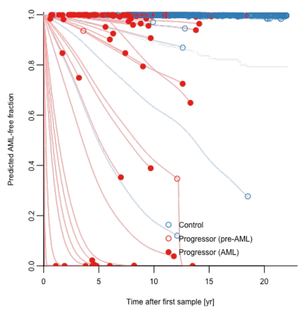
```
## Call:
## survConcordance(formula = allSurv ~ pp, weights = weights)
##
##   n= 950
## Concordance= 0.7561883 se= 0.02802535
## concordant discordant  tied.risk  tied.time   std(c-d)
##  6137610.4  1978900.7        0.0        1.0   454936.2
```

```
h <- exp(looAllWeighted$linear.predictor) > 100
plot(survfit(allSurv ~ h, weights=weights), col=set1[2:1], ylab="AML-free survival
", xlab="Time after first sample")
f <- sum(h*weights)/sum(weights) *100
legend("bottomleft", lty=1, col=set1[2:1], paste(c("low risk", "high risk"), "n ~"
, round(c( 100-f,f), 2),"%"))
```



7.4.9 Individual Predictions with corrected baseline

```
plot(survfit(allSurv~1), conf.int=FALSE, xlab='Time after first sample [yr]', ylab
='Predicted AML-free fraction', col='white', bty='L', yaxs='i', ylim=c(0,1.01))
d <- data.frame(t=NULL, s=NULL, pch=NULL, col=character())
for(i in unique(samples)){
    km <- exp(predict(smooth.spline(log(summary(survfit(allSurv[samples!=i,]~1, we
ights=weights[samples!=i]), times=t)$surv), df=10))$y)
    l0 <- colSums(fitAll$Z[samples!=i,,drop=FALSE] * weights[samples!=i]) %*% as.n
umeric(looAllWeighted[samples==i,][1,colnames(fitAll$Z)]) / sum(weights[samples!=i
])
    kmi <- function(km, s, lp, l0){
        .kmi <- function(km, sj, lpj, l0) km[t >= sj[,1] & t <= sj[,2]]^exp(lpj-l0
)
        k0 <- 1
        for(j in 1:nrow(s)) {
            k <- .kmi(km, s[j,], lp[j], l0)
            k <- k * k0/k[1]
            w <- t >= s[j,1] & t <= s[j,2]
            k0 <- k[length(k)]
            c <- if(s[nrow(s),3]==1) set1[1] else set1[2]
            lines(t[w], k, col=mg14:::colTrans(c), type='l')
            p <- if(s[j,3]==1) 19 else 1
            d <<- rbind(d, data.frame(t=t[w][length(k)], s=k[length(k)], pch=p, co
l=c))
        }
    }
    kmi(km, allSurv[samples==i,], looAllWeighted$linear.predictor[samples==i], l0)
}
points(d$t, d$s, pch=d$pch, col=as.character(d$col))
legend("bottomright", pch=c(1,1,19), col=c(set1[2], set1[1], set1[1]), legend=c("C
ontrol", "Progressor (pre-AML)", "Progressor (AML)"), bty='n')
```

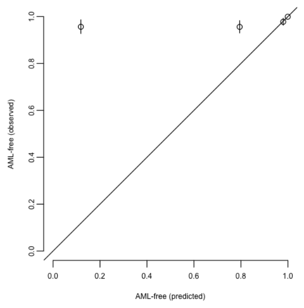


Callibration

```r
p10 <- km[t==10]^exp(looAllWeighted$linear.predictor)
c <- cut(p10, c(0,0.4,0.95,0.99,1))
table(c)
```

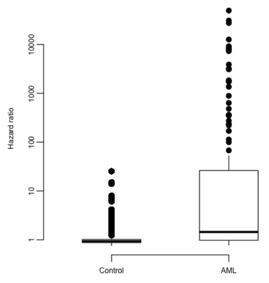
```
## c
##     (0,0.4]  (0.4,0.95] (0.95,0.99]    (0.99,1]
##          11          16          12         908
```

```r
s <- summary(survfit(allSurv~c, weights=weights), times=10)
m <- sapply(split(p10,c), mean)
plot(m, s$surv, xlab="AML-free (predicted)", ylab="AML-free (observed)", xlim=c(0,
1), ylim=c(0,1))
segments(m,s$lower,m,s$upper)
abline(0,1)
```



Hazard

```r
boxplot(exp(fitBoot$linear.predictors) ~ factor(1-control[b42], labels=c("Control"
,"AML")), log='y', ylab="Hazard ratio", pch=19, staplewex=0, lty=1, boxwex=0.5)
```



7.4.10 Some simulations

```r
bX <- sapply(1:50, function(foo){
            set.seed(foo)
            X <- rbind(apply(allX[control,], 2, sample, n_total-sum(!control), rep
lace=TRUE), apply(allX[!control,], 2, sample) )
            lambda0 <- 5e-4
            r <- X%*%coef(fitBoot)
            t <- rexp(n_total, lambda0 * exp(r))
            tmax <- 13 + runif(n_total, 0,1)
            s <- Surv(pmin(t,tmax), t < tmax)
            cases <- which(s[,2]==1)
            controls1 <- sample(which(s[,2]==0), size=1*length(cases))
            controls4 <- sample(which(s[,2]==0), size=sum(control))
            cbind(controls_inc=colMeans(X[controls4,allGroups=="Genes"]>0), AML_in
c=colMeans(X[cases,allGroups=="Genes"]>0), controls_vaf=apply(X[controls4,allGroup
s=="Genes"], 2, function(x) mean(x[x>0])),AML_vaf=apply(X[cases,allGroups=="Genes"
], 2, function(x) mean(x[x>0])))
        }, simplify='array')
```

Expected vs observed driver frequency

```
png("./figures/driver.freq.simulation.png", width = 15, height = 14, units = "cm",
res = 500)
par(mar = c(5, 4, 1.5, 0.5) + 0.1, mgp=c(2,0.4,0), las=1, tcl=-0.2)
plot(-rowMeans(bX[,'controls_inc',]), type='h', ylim=c(-.5,1)/2.5, lwd=8, xaxt='n'
, yaxt = 'n',  ylab="Control  -  Driver frequency (%)  -  Pre-AML", xlab="", col=s
et1[2])
atx <- axTicks(2)
axis(2,at=atx,labels= c(20, 10, 0, 10, 20, 30, 40))
points(x=1:22+.5,-colMeans(allX[control,allGroups=="Genes"]>0), type='h', lwd=8, c
ol=set1[1])
points(rowMeans(bX[,"AML_inc",]), type='h', lwd=8, col=set1[2])
points(x=1:22+.5,colMeans(allX[!control,allGroups=="Genes"]>0), type='h', lwd=8, c
ol=set1[1])
mtext(side=1, at=1:22,sub("_.+","",colnames(allX)[allGroups=="Genes"]), las=2, fon
t=3, line=0.7)
legend("topright", fill=set1[2:1], c("Expected","Observed"), cex = 0.8)
abline(h=0)
dev.off()
```

```
## pdf
##   2
```

Expected vs observed driver VAF

```
avgVaf <- function(x) mean(x[x>0])

png("./figures/driver.vaf.simulation.png", width = 15, height = 14, units = "cm",
res = 500)
par(mar = c(5, 4, 1.5, 0.5) + 0.1, mgp=c(2,0.4,0), las=1, tcl=-0.2)
plot(-apply(bX[,'controls_vaf',],1,avgVaf)*10, type='h', ylim=c(-40,50), lwd=8, xa
xt='n', yaxt = 'n', ylab="Control  -  Driver VAF (%)  -  Pre-AML", xlab="", col=se
t1[2])
atx <- axTicks(2)
axis(2,at=atx,labels= c(40, 20,0, 20, 40))
points(x=1:22+.5,-apply(allX[control,allGroups=="Genes"],2,avgVaf)*10, type='h', l
wd=8, col=set1[1])
points(apply(bX[,"AML_vaf",],1,avgVaf)*10, type='h', lwd=8, col=set1[2])
points(x=1:22+.5,apply(allX[!control,allGroups=="Genes"],2,avgVaf)*10, type='h', l
wd=8, col=set1[1])
mtext(side=1, at=1:22,sub("_.+","",colnames(allX)[allGroups=="Genes"]), las=2, fon
t=3, line = 0.6)
legend("bottomleft", fill=set1[2:1], c("Expected","Observed"), cex = 0.8)
abline(h=0)
dev.off()
```

```
## pdf
##   2
```

7.4.11 Simple models

```
samples <- factor(c(as.character(sangerData$Individual), as.character(torontoData$
Sample)))
```

max vaf:

```
v <- apply(allX[,allGroups=="Genes"], 1, max)*10
```

cumulative vaf

```
c <- apply(allX[,allGroups=="Genes"], 1, sum)*10
```

number of mutations

```
m <- rowSums(allX[,allGroups=="Genes"]>0)
```

any mutation

```
a <- as.integer(m>0)
```

7.4.11.1 Presence of any mutation

```
d <- data.frame(a)
summary(f <- coxph(allSurv ~ ., data=d ))
```

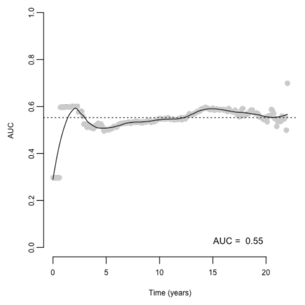
```
## Call:
## coxph(formula = allSurv ~ ., data = d)
##
##   n= 950, number of events= 120
##
##     coef exp(coef) se(coef)     z Pr(>|z|)
## a 1.5144    4.5468   0.2046 7.402 1.35e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##   exp(coef) exp(-coef) lower .95 upper .95
## a     4.547     0.2199     3.045      6.79
##
## Concordance= 0.672  (se = 0.023 )
## Rsquare= 0.064   (max possible= 0.801 )
## Likelihood ratio test= 63.31  on 1 df,   p=1.776e-15
## Wald test            = 54.78  on 1 df,   p=1.347e-13
## Score (logrank) test = 66.02  on 1 df,   p=4.441e-16
```

```
los <- do.call("rbind",mclapply(levels(samples), function(l){
  i <- samples!=l
  f <<- coxph(allSurv ~ ., data=d, subset=i)
  p <- as.matrix(d[!i,]) %*% f$coefficients
  r <- cbind(matrix(f$coefficients, nrow=length(p), ncol=length(f$coefficients), b
yrow=TRUE), linear.predictor=p)
  colnames(r) <- c(names(f$coefficients), "linear.predictor")
  as.data.frame(r)
}, mc.cores=4))
psAnyMt <- los[order(order(samples)),]

survConcordance(allSurv ~ psAnyMt$linear.predictor)
```

```
## Call:
## survConcordance(formula = allSurv ~ psAnyMt$linear.predictor)
##
##   n= 950
## Concordance= 0.5431925 se= 0.02388586
## concordant discordant  tied.risk  tied.time   std(c-d)
##  34829.000  28205.000  13646.000      1.000   3663.136
```

Dynamic/cumulative AUC

```
w <- c(which(allSurv[,1]==0)[-1]-1, nrow(allSurv))
survAll2 <- Surv(allSurv[w,2], allSurv[w,3])
t <- seq(0,22,0.1)
allX2 <- allX[w, ]

auc.uno <- AUC.uno(survAll2, survAll2, psAnyMt$linear.predictor[w], times=t)

plot(auc.uno$times, auc.uno$auc, xlab="Time (years)", ylab="AUC", pch=16, col="gre
y80", ylim = c(0,1.0))
lines(auc.uno$times, predict(loess(auc.uno$auc ~ auc.uno$times, span=0.25)))
abline(h=auc.uno$iauc, lty = 3, lwd = 1)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(auc.uno$
iauc,2)))
```



```
AnyMt.a <- auc.uno
```

Presence of any mutation + vaf

```
d <- data.frame(a,v)
summary(f <- coxph(allSurv ~ ., data=d ))
```

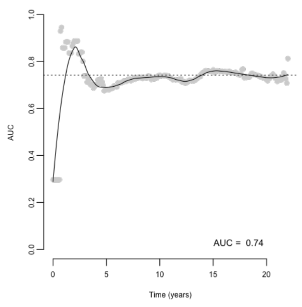
```
## Call:
## coxph(formula = allSurv ~ ., data = d)
##
##   n= 950, number of events= 120
##
##       coef exp(coef) se(coef)     z Pr(>|z|)
## a 1.025548  2.788622 0.223677 4.585 4.54e-06 ***
## v 0.050613  1.051915 0.005605 9.030  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##   exp(coef) exp(-coef) lower .95 upper .95
## a     2.789     0.3586     1.799     4.323
## v     1.052     0.9506     1.040     1.064
##
## Concordance= 0.737  (se = 0.024 )
## Rsquare= 0.119   (max possible= 0.801 )
## Likelihood ratio test= 120.5  on 2 df,   p=0
## Wald test            = 161.8  on 2 df,   p=0
## Score (logrank) test = 263.9  on 2 df,   p=0
```

```
los <- do.call("rbind",mclapply(levels(samples), function(l){
  i <- samples!=l
  f <<- coxph(allSurv ~ ., data=d, subset=i)
  p <- as.matrix(d[!i,]) %*% f$coefficients
  r <- cbind(matrix(f$coefficients, nrow=length(p), ncol=length(f$coefficients), b
yrow=TRUE), linear.predictor=p)
  colnames(r) <- c(names(f$coefficients), "linear.predictor")
  as.data.frame(r)
}, mc.cores=4))
psAnyMtVaf <- los[order(order(samples)),]

survConcordance(allSurv ~ psAnyMtVaf$linear.predictor)
```

```
## Call:
## survConcordance(formula = allSurv ~ psAnyMtVaf$linear.predictor)
##
##   n= 950
## Concordance= 0.7287559 se= 0.0238873
## concordant discordant  tied.risk  tied.time   std(c-d)
##  49091.000  14009.000  13580.000      1.000   3663.356
```

Dynamic/cumulative AUC

```
auc.uno <- AUC.uno(survAll2, survAll2, psAnyMtVaf$linear.predictor[w], times=t)

plot(auc.uno$times, auc.uno$auc, xlab="Time (years)", ylab="AUC", pch=16, col="gre
y80", ylim = c(0,1.0))
lines(auc.uno$times, predict(loess(auc.uno$auc ~ auc.uno$times, span=0.25)))
abline(h=auc.uno$iauc, lty = 3, lwd = 1)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(auc.uno$
iauc,2)))
```



```
AnyMtVaf.a <- auc.uno
```

7.4.11.2 Number of mutations + vaf

```
d <- data.frame(m,v)
summary(f <- coxph(allSurv ~ ., data=d ))
```

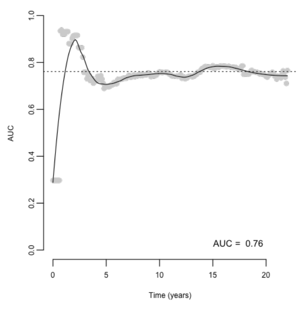
```
## Call:
## coxph(formula = allSurv ~ ., data = d)
##
##   n= 950, number of events= 120
##
##       coef exp(coef) se(coef)     z Pr(>|z|)
## m 0.653487  1.922231 0.088287 7.402 1.34e-13 ***
## v 0.040976  1.041827 0.006562 6.245 4.25e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##   exp(coef) exp(-coef) lower .95 upper .95
## m     1.922     0.5202     1.617     2.285
## v     1.042     0.9599     1.029     1.055
##
## Concordance= 0.744  (se = 0.024 )
## Rsquare= 0.142   (max possible= 0.801 )
## Likelihood ratio test= 145.3  on 2 df,   p=0
## Wald test            = 213.3  on 2 df,   p=0
## Score (logrank) test = 302.9  on 2 df,   p=0
```

```
los <- do.call("rbind",mclapply(levels(samples), function(l){
  i <- samples!=l
  f <<- coxph(allSurv ~ ., data=d, subset=i)
  p <- as.matrix(d[!i,]) %*% f$coefficients
  r <- cbind(matrix(f$coefficients, nrow=length(p), ncol=length(f$coefficients), b
yrow=TRUE), linear.predictor=p)
  colnames(r) <- c(names(f$coefficients), "linear.predictor")
  as.data.frame(r)
}, mc.cores=4))
psNMtVaf <- los[order(order(samples)),]

survConcordance(allSurv ~ psNMtVaf$linear.predictor)
```

```
## Call:
## survConcordance(formula = allSurv ~ psNMtVaf$linear.predictor)
##
##   n= 950
## Concordance= 0.7431403 se= 0.0238873
## concordant discordant  tied.risk  tied.time   std(c-d)
##  50194.000  12906.000  13580.000      1.000   3663.356
```

Dynamic/cumulative AUC

```
auc.uno <- AUC.uno(survAll2, survAll2, psNMtVaf$linear.predictor[w], times=t)

plot(auc.uno$times, auc.uno$auc, xlab="Time (years)", ylab="AUC", pch=16, col="gre
y80", ylim = c(0,1.0))
lines(auc.uno$times, predict(loess(auc.uno$auc ~ auc.uno$times, span=0.25)))
abline(h=auc.uno$iauc, lty = 3, lwd = 1)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(auc.uno$
iauc,2)))
```



```
NMtVaf.a <- auc.uno
```

7.4.11.3 Number of mutations + cumulative vaf

```
d <- data.frame(m,c)
summary(f <- coxph(allSurv ~ ., data=d ))
```

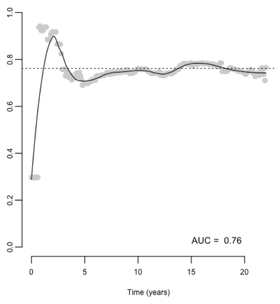
```
## Call:
## coxph(formula = allSurv ~ ., data = d)
##
##   n= 950, number of events= 120
##
##       coef exp(coef) se(coef)     z Pr(>|z|)
## m 0.613264  1.846449 0.090393 6.784 1.17e-11 ***
## c 0.033648  1.034220 0.005036 6.681 2.38e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##   exp(coef) exp(-coef) lower .95 upper .95
## m     1.846     0.5416     1.547     2.204
## c     1.034     0.9669     1.024     1.044
##
## Concordance= 0.744  (se = 0.024 )
## Rsquare= 0.144   (max possible= 0.801 )
## Likelihood ratio test= 148.2  on 2 df,   p=0
## Wald test            = 223.3  on 2 df,   p=0
## Score (logrank) test = 350.7  on 2 df,   p=0
```

```
los <- do.call("rbind",mclapply(levels(samples), function(l){
  i <- samples!=l
  f <<- coxph(allSurv ~ ., data=d, subset=i)
  p <- as.matrix(d[!i,]) %*% f$coefficients
  r <- cbind(matrix(f$coefficients, nrow=length(p), ncol=length(f$coefficients), b
yrow=TRUE), linear.predictor=p)
  colnames(r) <- c(names(f$coefficients), "linear.predictor")
  as.data.frame(r)
}, mc.cores=4))
psNMtCumVaf <- los[order(order(samples)),]

survConcordance(allSurv ~ psNMtCumVaf$linear.predictor)
```

```
## Call:
## survConcordance(formula = allSurv ~ psNMtCumVaf$linear.predictor)
##
##   n= 950
## Concordance= 0.743362 se= 0.0238873
## concordant discordant  tied.risk  tied.time   std(c-d)
##  50211.000  12889.000  13580.000      1.000   3663.356
```

Dynamic/cumulative AUC

```
auc.uno <- AUC.uno(survAll2, survAll2, psNMtCumVaf$linear.predictor[w], times=t)

plot(auc.uno$times, auc.uno$auc, xlab="Time (years)", ylab="AUC", pch=16, col="gre
y80", ylim = c(0,1.0))
lines(auc.uno$times, predict(loess(auc.uno$auc ~ auc.uno$times, span=0.25)))
abline(h=auc.uno$iauc, lty = 3, lwd = 1)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(auc.uno$
iauc,2)))
```



```
NMtCumVaf.a <- auc.uno
```

Gene-level risks

```
d <- allX
summary(f <- coxph(allSurv ~ ., data=d))
```

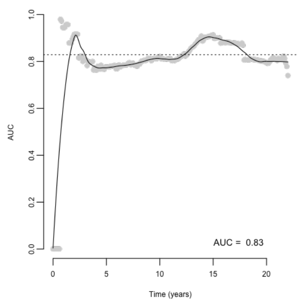
```
## Call:
## coxph(formula = allSurv ~ ., data = d)
##
##   n= 950, number of events= 120
##
##                  coef  exp(coef)   se(coef)      z Pr(>|z|)
## ASXL1_0.1     0.45410    1.57475    0.25483  1.782   0.0748 .
## BCOR_0.1      4.53517   93.23942   15.29850  0.296   0.7669
## CBL_0.1       0.02418    1.02448    0.74288  0.033   0.9740
## DNMT3A_0.1    0.13468    1.14417    0.18286  0.737   0.4614
## IDH1_0.1      0.39412    1.48307    0.63231  0.623   0.5331
## IDH2_0.1      0.51163    1.66800    0.29079  1.759   0.0785 .
## JAK2_0.1      0.59064    1.80514    0.39331  1.502   0.1332
## KDM6A_0.1     0.15988    1.17337   32.12704  0.005   0.9960
## KMT2C_0.1    -0.50258    0.60497    1.77003 -0.284   0.7765
## KMT2D_0.1    -0.01333    0.98676    0.58364 -0.023   0.9818
## KRAS_0.1      0.54336    1.72178   12.36468  0.044   0.9649
## NF1_0.1      -0.76668    0.46455    5.94275 -0.129   0.8973
## NRAS_0.1      7.40428 1643.00852    6.01855  1.230   0.2186
## PHF6_0.1      4.31340   74.69375   15.42773  0.280   0.7798
## PTPN11_0.1    4.49429   89.50474    6.18432  0.727   0.4674
## RAD21_0.1     0.07319    1.07594    6.89358  0.011   0.9915
## RUNX1_0.1     0.17980    1.19698    0.24611  0.731   0.4650
## SF3B1_0.1     1.10331    3.01414    0.52063  2.119   0.0341 *
## SRSF2_0.1     0.34535    1.41248    0.21771  1.586   0.1127
## TET2_0.1      0.17179    1.18743    0.20206  0.850   0.3952
## TP53_0.1      2.17381    8.79176    0.55321  3.929 8.51e-05 ***
## U2AF1_0.1     2.74012   15.48884    0.35246  7.774 7.55e-15 ***
## age_10       -0.01189    0.98818    0.10907 -0.109   0.9132
## gender       -0.01138    0.98868    0.19862 -0.057   0.9543
## cohort       -0.13561    0.87318    0.23791 -0.570   0.5687
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##            exp(coef) exp(-coef) lower .95 upper .95
## ASXL1_0.1     1.5747  0.6350222 9.557e-01 2.595e+00
## BCOR_0.1     93.2394  0.0107251 8.861e-12 9.811e+14
## CBL_0.1       1.0245  0.9761095 2.389e-01 4.394e+00
## DNMT3A_0.1    1.1442  0.8739972 7.995e-01 1.637e+00
## IDH1_0.1      1.4831  0.6742750 4.295e-01 5.121e+00
## IDH2_0.1      1.6680  0.5995195 9.434e-01 2.949e+00
## JAK2_0.1      1.8051  0.5539734 8.351e-01 3.902e+00
## KDM6A_0.1     1.1734  0.8522477 5.283e-28 2.606e+27
## KMT2C_0.1     0.6050  1.6529815 1.884e-02 1.943e+01
## KMT2D_0.1     0.9868  1.0134221 3.144e-01 3.097e+00
## KRAS_0.1      1.7218  0.5807959 5.142e-11 5.765e+10
## NF1_0.1       0.4646  2.1526020 4.060e-06 5.315e+04
## NRAS_0.1   1643.0085  0.0006086 1.238e-02 2.181e+08
## PHF6_0.1     74.6937  0.0133880 5.510e-12 1.012e+15
## PTPN11_0.1   89.5047  0.0111726 4.872e-04 1.644e+07
## RAD21_0.1     1.0759  0.9294227 1.459e-06 7.936e+05
## RUNX1_0.1     1.1970  0.8354364 7.389e-01 1.939e+00
## SF3B1_0.1     3.0141  0.3317696 1.086e+00 8.362e+00
## SRSF2_0.1     1.4125  0.7079756 9.219e-01 2.164e+00
## TET2_0.1      1.1874  0.8421566 7.991e-01 1.764e+00
## TP53_0.1      8.7918  0.1137429 2.973e+00 2.600e+01
## U2AF1_0.1    15.4888  0.0645626 7.763e+00 3.091e+01
## age_10        0.9882  1.0119578 7.980e-01 1.224e+00
## gender        0.9887  1.0114489 6.699e-01 1.459e+00
## cohort        0.8732  1.1452345 5.478e-01 1.392e+00
##
## Concordance= 0.81  (se = 0.027 )
## Rsquare= 0.069   (max possible= 0.801 )
## Likelihood ratio test= 67.53  on 25 df,   p=8.884e-06
## Wald test            = 110.8  on 25 df,   p=9.049e-13
## Score (logrank) test = 782.6  on 25 df,   p=0
```

```
los <- do.call("rbind",mclapply(levels(samples), function(l){
  i <- samples!=l
  f <<- coxph(allSurv ~ ., data=d, subset=i)
  p <- as.matrix(d[!i,]) %*% f$coefficients
  r <- cbind(matrix(f$coefficients, nrow=length(p), ncol=length(f$coefficients), b
yrow=TRUE), linear.predictor=p)
  colnames(r) <- c(names(f$coefficients), "linear.predictor")
  as.data.frame(r)
}, mc.cores=4))
psGenes <- los[order(order(samples)),]

survConcordance(allSurv ~ psGenes$linear.predictor)
```

```
## Call:
## survConcordance(formula = allSurv ~ psGenes$linear.predictor)
##
##   n= 950
## Concordance= 0.7799296 se= 0.02746328
## concordant discordant  tied.risk  tied.time   std(c-d)
##  59805.000  16875.000      0.000      1.000   4211.768
```

Dynamic/cumulative AUC

```
auc.uno <- AUC.uno(survAll2, survAll2, psGenes$linear.predictor[w], times=t)

plot(auc.uno$times, auc.uno$auc, xlab="Time (years)", ylab="AUC", pch=16, col="gre
y80", ylim = c(0,1.0))
lines(auc.uno$times, predict(loess(auc.uno$auc ~ auc.uno$times, span=0.25)))
abline(h=auc.uno$iauc, lty = 3, lwd = 1)
legend("bottomright", bty = "n", cex = 1.2, legend = paste("AUC = ",round(auc.uno$
iauc,2)))
```

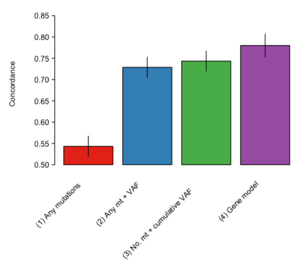


```
Genes.a <- auc.uno

# Concordance summary
c <- rbind(
  `(1) Any mutations`=as.data.frame(survConcordance(allSurv ~ psAnyMt$linear.predi
ctor)[c("concordance","std.err")]),
  `(2) Any mt + VAF`=as.data.frame(survConcordance(allSurv ~ psAnyMtVaf$linear.pre
dictor)[c("concordance","std.err")]),
  `(3) No. mt + cumulative VAF`=as.data.frame(survConcordance(allSurv ~ psNMtCumVa
f$linear.predictor)[c("concordance","std.err")]),
  `(4) Gene model`=as.data.frame(survConcordance(allSurv ~ psGenes$linear.predicto
r)[c("concordance","std.err")]))

c
```

	concordance <dbl>	std.err <dbl>
(1) Any mutations	0.5431925	0.02388586
(2) Any mt + VAF	0.7287559	0.02388730
(3) No. mt + cumulative VAF	0.7433620	0.02388730
(4) Gene model	0.7799296	0.02746328

4 rows

```
set1 <- RColorBrewer::brewer.pal(6,"Set1")

par(mar = c(9, 4, 1.5, 0.5) + 0.1, mgp=c(2.7,0.4,0), las=1, tcl=-0.2)
b <- barplot(c$concordance-0.5, ylab="Concordance", col=set1, ylim=c(0.5,0.88), of
fset=0.5)
mg14::rotatedLabel(x=b, labels=rownames(c))
segments(b,c$concordance+c$std.err,b,c$concordance-c$std.err)
```



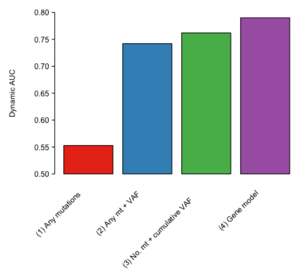
Dynamic/cumulative AUC summary

```
d.auc <- data.frame(iauc = c(AnyMt.a$iauc, AnyMtVaf.a$iauc, NMtCumVaf.a$iauc, 0.79
))
rownames(d.auc) <- c("(1) Any mutations", "(2) Any mt + VAF", "(3) No. mt + cumula
tive VAF", "(4) Gene model")

d.auc
```

	iauc <dbl>
(1) Any mutations	0.5528776
(2) Any mt + VAF	0.7420613
(3) No. mt + cumulative VAF	0.7618961
(4) Gene model	0.7900000

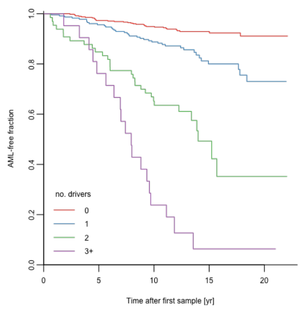
4 rows

```
par(mar = c(9, 4, 1.5, 0.5) + 0.1, mgp=c(2.7,0.4,0), las=1, tcl=-0.2)
b <- barplot(d.auc$iauc-0.5, ylab="Dynamic AUC", col=set1, ylim=c(0.5,0.80), offse
t=0.5)
mg14::rotatedLabel(x=b, labels=rownames(d.auc))
```



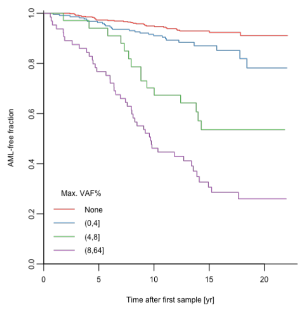
AML-free survival by number of drivers

```
nonc <- rowSums(allX[,allGroups=="Genes"]>0)
nonc <- cut(nonc, c(-1,0,1,2,max(nonc)))
plot(survfit(allSurv~nonc), col=set1, xlab='Time after first sample [yr]', ylab='A
ML-free fraction', bty='L', yaxs='i', ylim=c(0,1.01))
legend("bottomleft", c(0,1,2,"3+"), col=set1, lty=1, bty='n', title="no. drivers")
```



AML-free survival by max VAF

```
mvaf <- apply(allX[,allGroups=="Genes"], 1, max)*10
mvaf <- cut(mvaf, c(-1,0,4,8,max(mvaf)))
plot(survfit(allSurv~mvaf), col=set1, xlab='Time after first sample [yr]', ylab='A
ML-free fraction', bty='L', yaxs='i', ylim=c(0,1.01))
levels(mvaf)[1] <- "None"
legend("bottomleft", levels(mvaf), col=set1, lty=1, bty='n', title="Max. VAF%")
```



8 Logistic regression

```
library(glmnet)
library(ROCR)
```

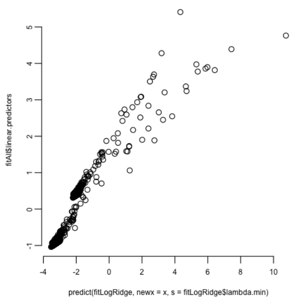
```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

8.1 Combined

```
set.seed(42)
y <- allSurv[,3]
x <- allX
x <- as.matrix(cbind(x, mu.Genes=rowSums(x[,allGroups=="Genes"])))
fitLogRidge <- cv.glmnet(x, y, alpha=0, standardize=FALSE, penalty.factor=c(allGro
ups=="Genes",FALSE), family="binomial", lambda=10^seq(-5,5,0.1)/nrow(x))
fitLog <- glm(y ~ x[,-ncol(x)], family="binomial")
coefLogRidge <- coef(fitLogRidge, s=fitLogRidge$lambda.min)[-1,1]
w <- names(coefLogRidge) %in% colnames(allX)[allGroups=="Genes"]
coefLogRidge[w] <- coefLogRidge[w] + coefLogRidge["mu.Genes"]
names(coefLogRidge) <- colnames(x)
s <- summary(survfit(allSurv ~1))

plot(predict(fitLogRidge, newx=x, s=fitLogRidge$lambda.min),fitAll$linear.predicto
rs)
```



```
cor(predict(fitLogRidge, newx=x, s=fitLogRidge$lambda.min),fitAll$linear.predictor
s)
```

```
##        [,1]
## 1 0.9325608
```

8.2 Discovery cohort

```
set.seed(42)
x <- cbind(as.matrix(torontoX), mu.Genes=rowSums(torontoX[torontoGroups=="Genes"])
)
fitLogRidgeToronto <- cv.glmnet(x, torontoSurv[,2], alpha=0, standardize=FALSE, pe
nalty.factor=c(torontoGroups=="Genes",FALSE), family="binomial", lambda=10^seq(-5,
5,0.1)/nrow(x))
l <- max(which(abs(fitLogRidgeToronto$cvm- min(fitLogRidgeToronto$cvm)) < 0.01))
coefFitLogRidgeToronto <- coef(fitLogRidgeToronto, s=fitLogRidge$lambda.min *nrow(
allX)/nrow(torontoX))[-1,1]
w <- names(coefFitLogRidgeToronto) %in% colnames(torontoX)[torontoGroups=="Genes"]
coefFitLogRidgeToronto[w] <- coefFitLogRidgeToronto[w] + coefFitLogRidgeToronto["m
u.Genes"]
```

8.3 Validation cohort

```
set.seed(42)
x <- cbind(as.matrix(sangerX), mu.Genes=rowSums(sangerX[sangerGroups=="Genes"]))
y <- sangerSurv[,3]
fitLogRidgeSanger <- glmnet(x, y, alpha=0, standardize=FALSE, penalty.factor=c(san
gerGroups%in%c("Genes","Blood"),1e-2) , family="binomial",lambda=10^seq(-5,5,0.1)/
nrow(x))
coefFitLogRidgeSanger <- coef(fitLogRidgeSanger, s=fitLogRidge$lambda.min*nrow(all
X)/nrow(sangerX)/4)[-1,1]
w <- names(coefFitLogRidgeSanger) %in% colnames(sangerX)[sangerGroups=="Genes"]
coefFitLogRidgeSanger[w] <- coefFitLogRidgeSanger[w] + coefFitLogRidgeSanger["mu.G
enes"]
coefFitLogRidgeSanger
```

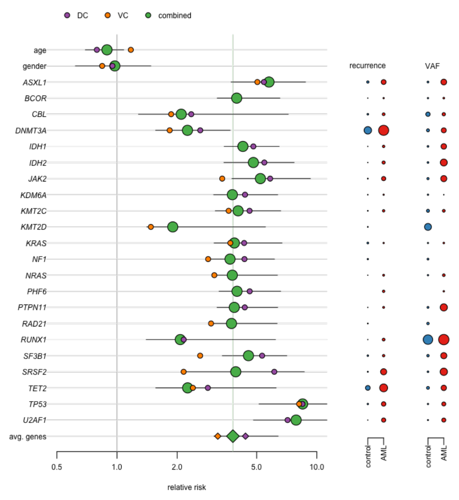
```
##   ASXL1_0.1     CBL_0.1  DNMT3A_0.1    JAK2_0.1   KMT2C_0.1   KMT2D_0.1    KRAS
_0.1     NF1_0.1    NRAS_0.1   RAD21_0.1
##  1.61735484  0.62402794  0.60690505  1.21223108  1.28664688  0.38990853  1.3057
9768  1.05008349  1.12131863  1.08384807
##   SF3B1_0.1   SRSF2_0.1    TET2_0.1    TP53_0.1   U2AF1_0.1      age_10      ge
nder  systol_100 diastol_100      bmi_10
##  0.95795153  0.76775960  0.87432787  2.09849607  2.46513749  0.15915519 -0.1710
4884 -0.26674155  0.40623412  0.78151214
## cholestl_10     triglyc         hdl         ldl         lym     mcv_100      rd
w_10      wbc_10     plt_100      hgb_10
##  0.02221735 -0.02231645 -0.60655423  0.08051073  0.02388812 -0.48424380  1.4392
5261 -0.13343432  0.28531137  0.80105113
##    mu.Genes
##  1.16143798
```

8.4 Bootstrap CIs

```
coefLogRidgeBoot <- sapply(1:100, function(foo){
            set.seed(foo)
            y <- allSurv[,3]
            x <- allX
            x <- as.matrix(cbind(x, mu.Genes=rowSums(x[,allGroups=="Genes"])))
            b <- sample(1:nrow(x), replace=TRUE)
            fitLogRidgeBoot <- glmnet(x[b,], y[b], alpha=0, standardize=FALSE, pen
alty.factor=c(allGroups=="Genes",FALSE, FALSE), family="binomial", lambda=10^seq(-
5,5,0.1)/nrow(x))
            coefLogRidgeBoot <- coef(fitLogRidgeBoot, s=fitLogRidge$lambda.min)[-1
,1]
            w <- names(coefLogRidgeBoot) %in% colnames(allX)[allGroups=="Genes"]
            coefLogRidgeBoot[w] <- coefLogRidgeBoot[w] + coefLogRidgeBoot["mu.Gene
s"]
            names(coefLogRidgeBoot) <- colnames(x)
            coefLogRidgeBoot
        })
```

8.5 Forest plot

```
par(bty="n", mar=c(3,6,3,10)+.5, mgp=c(2,0.5,0), xpd=FALSE)
c <- exp(coefLogRidge[-25])
o <- c(23:24,1:22,25)
ci <- apply(coefLogRidgeBoot,1,quantile, c(0.025,0.975))[,-25]
y <- rev(seq_along(c))
plot(c[o], y, xlab="relative risk", log='x', ylab='', yaxt="n", pch=NA, xlim=c(0.5
,10))
abline(h=y, col="#EEEEEE", lty=1)
abline(v=1, lty=1, col="grey")
abline(v=c["mu.Genes"], col=mg14::colTrans(set1[3]), lty=1)
segments(exp(ci[1,o]), y, exp(ci[2,o]),y)
points(c[o], y, xlab="relative risk",  bg=set1[3], cex=2, pch=c(rep(21,24), 23))
m <- match(names(c)[o],names(coefFitLogRidgeToronto))
points(exp(coefFitLogRidgeToronto[m]), y,bg=set1[4], pch=c(rep(21,24), 23), cex=1)
m <- match(names(c)[o],names(coefFitLogRidgeSanger))
points(exp(coefFitLogRidgeSanger[m]), y,bg=set1[5], pch=c(rep(21,24), 23), cex=1)
mtext(side=2, sub("mu.Genes","avg. genes",sub("_.+","",names(c)[o])), at=y, las=2,
font=c(1,1,rep(3,22),1))

r <- sapply(split(as.data.frame(allX>0), control), colMeans)
f <- sapply(split(allX, control), apply, 2, function(x) mean(x[x>0]))
par(xpd=NA)
points(rep(18,22),y[3:24], cex=sqrt(r[o[3:24],2]*10), pch=21, bg=set1[2])
points(rep(18*1.2,22), y[3:24], cex=sqrt(r[o[3:24],1]*10), pch=21, bg=set1[1])
points(rep(36,22),y[3:24], cex=sqrt(f[o[3:24],2]), pch=21, bg=set1[2])
points(rep(36*1.2,22), y[3:24], cex=sqrt(f[o[3:24],1]), pch=21, bg=set1[1])
legend(x=0.5, y=28, pch=21, pt.bg=set1[c(4,5,3)], c("DC","VC","combined"), bty="n"
, ncol=3, text.width=0.1)

text(y=24, x=18, "recurrence")
text(y=24, x=38, "VAF")

axis(1, at=c(18,18*1.2), c("control","AML"), las=2, line=-1)
axis(1, at=c(36,36*1.2), c("control","AML"), las=2, line=-1)
```



8.6 AUC

```
aucLogRidgeBoot <- t(sapply(1:100, function(foo){
                    set.seed(foo)
                    y <- allSurv[,3]
                    x <- allX
                    x <- as.matrix(cbind(x, mu.Genes=rowSums(x[,allGroups=="Genes"
])))
                    b <- sample(1:nrow(x), replace=TRUE)
                    oob <- setdiff(1:nrow(x),b)
                    c(inb=performance(prediction(x[b,] %*% coefLogRidgeBoot[,foo],
y[b]),"auc")@y.values[[1]],
                            oob=performance(prediction(x[oob,] %*% coefLogRidgeBoo
t[,foo], y[oob]),"auc")@y.values[[1]])
                }))

apply(aucLogRidgeBoot, 2, quantile)
```

```
##            inb       oob
## 0%   0.7600825 0.7331746
## 25%  0.7981192 0.7814137
## 50%  0.8107881 0.8058353
## 75%  0.8228798 0.8254089
## 100% 0.8616209 0.8650056
```

```
performance(prediction(as.matrix(torontoX) %*% coefFitLogRidgeToronto[-22], toront
oSurv[,2]),"auc")@y.values[[1]]
```

```
## [1] 0.7649573
```

```
performance(prediction(as.matrix(sangerImp) %*% coefFitLogRidgeToronto[-22], sange
rSurv[,3]),"auc")@y.values[[1]]
```

```
## [1] 0.806366
```

```
performance(prediction(as.matrix(sangerX) %*% coefFitLogRidgeSanger[-31], sangerSu
rv[,3]),"auc")@y.values[[1]]
```

```
## [1] 0.8479775
```

```
performance(prediction(ImputeMissing(sangerX, as.matrix(torontoImp)) %*% coefFitLo
gRidgeSanger[-31], torontoSurv[,2]),"auc")@y.values[[1]]
```

```
## [1] 0.6885916
```

9 Tabulate results

```
# library(xlsx)
# wb <- createWorkbook("xlsx")
# sheet  <- createSheet(wb, sheetName="Cox PH adjusted (combined)")
# addDataFrame(waldWeighted,
#       sheet,
#       colnamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE) + Border(),
#       rownamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE)
# )
# sheet  <- createSheet(wb, sheetName="Cox PH adjusted (DC)")
# addDataFrame(waldWeightedToronto,
#       sheet,
#       colnamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE) + Border(),
#       rownamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE)
# )
#
# sheet  <- createSheet(wb, sheetName="Cox PH adjusted (VC)")
# addDataFrame(waldWeightedSanger,
#       sheet,
#       colnamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE) + Border(),
#       rownamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE)
# )
#
# sheet  <- createSheet(wb, sheetName="Logistic regression (combined)")
# addDataFrame(data.frame(`Coef combined`=coefLogRidge, CI=t(apply(coefLogRidgeBoo
t, 1, quantile, c(0.025,0.975))),
#               check.names=FALSE),
#       sheet,
#       colnamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE) + Border(),
#       rownamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE)
# )
#
# sheet  <- createSheet(wb, sheetName="Logistic regression (DC)")
# addDataFrame(data.frame(`Coef combined`=coefFitLogRidgeToronto,
#               check.names=FALSE),
#       sheet,
#       colnamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE) + Border(),
#       rownamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE)
# )
# sheet  <- createSheet(wb, sheetName="Logistic regression (Sanger)")
# addDataFrame(data.frame(`Coef combined`=coefFitLogRidgeSanger,
#               check.names=FALSE),
#       sheet,
#       colnamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE) + Border(),
#       rownamesStyle = CellStyle(wb) + Font(wb, isBold=TRUE)
# )
# saveWorkbook(wb, file="SupplementaryTables.xlsx")
```

10 Clinical/Demographic model

Necessary to reconstruct matrices and survival objects to use data from VC for all 8 samples sequenced in both cohorts ## Discovery cohort Data 83 pre-AML (keeping duplicates with validation cohort)

```
f = "./arch_data/DC_vaf_matrix_no_duplicates_414ctrl_83aml.csv"
torontoData <- read.csv(f)

torontoData$gender <- ifelse(torontoData$Sex == "male", 1,
                             ifelse(torontoData$Sex == "female", 0, torontoData$Se
x))
table(torontoData$gender)
```

```
##
##   0   1
## 293 204
```

```
torontoData$gender <- as.numeric(torontoData$gender)
colnames(torontoData)
```

```
##  [1] "Sample"     "ASXL1"      "BCOR"       "CALR"       "CBL"        "DNMT3A"
"IDH1"       "IDH2"
##  [9] "JAK2"       "KDM6A"      "KIT"        "KMT2C"      "KRAS"       "NF1"
"NRAS"       "PHF6"
## [17] "PTPN11"     "RUNX1"      "SF3B1"      "SRSF2"      "TET2"       "TP53"
"U2AF1"      "Diagnosis"
## [25] "fu_years"   "age"        "Sex"        "no_drivers" "gender"
```

Manually standardize magnitudes

```
torontoData <- torontoData[!duplicated(torontoData),]

gene_vars <- c("CALR", "NRAS", "DNMT3A", "SF3B1", "IDH1", "KIT", "TET2", "RAD21",
"JAK2", "CBL", "KRAS", "PTPN11", "IDH2", "TP53", "NF1", "SRSF2", "CEBPA", "ASXL1",
"RUNX1", "U2AF1", "BCOR", "KDM6A", "PHF6", "KMT2C", "KMT2D")

torontoX <- torontoData[, colnames(torontoData) %in% c(gene_vars, "age", "gender")
]

torontoX <- as.data.frame(torontoX)
```

Only include genes in model if mutated in >2 samples

```
thr <- 2
torontoX <- torontoX[,colSums(torontoX != 0)>=thr]

torontoGroups <- factor(names(torontoX) %in% c("age","gender")+1, level=1:2, label
s=c("Genes","Demographics"))
colnames(torontoX)
```

```
##  [1] "ASXL1"  "CALR"   "CBL"    "DNMT3A" "IDH1"   "IDH2"   "JAK2"   "KDM6A"  "K
MT2C"  "KRAS"   "NF1"    "PHF6"
## [13] "PTPN11" "RUNX1"  "SF3B1"  "SRSF2"  "TET2"   "TP53"   "U2AF1"  "age"    "g
ender"
```

```
torontoGroups
```

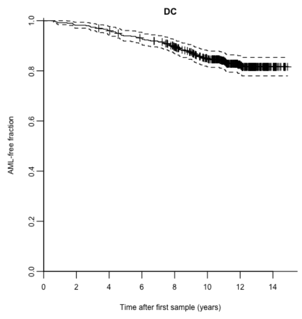
```
##  [1] Genes        Genes        Genes        Genes        Genes        Genes       
Genes        Genes       
##  [9] Genes        Genes        Genes        Genes        Genes        Genes       
Genes        Genes       
## [17] Genes        Genes        Genes        Demographics Demographics
## Levels: Genes Demographics
```

Manually standardize age and mutation VAFs

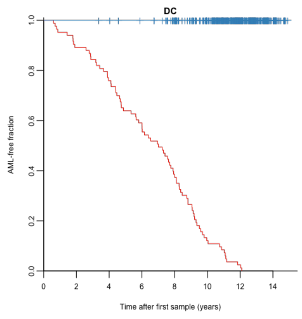
```
torontoX$age <- torontoX$age/10
names(torontoX)[which(names(torontoX)=="age")] <- "age_10"
g <- torontoGroups == "Genes"
torontoX[,g] <- torontoX[,g]*10
names(torontoX)[g] <- paste(names(torontoX)[g], "0.1",sep="_")
colnames(torontoX)
```

```
##  [1] "ASXL1_0.1"  "CALR_0.1"   "CBL_0.1"    "DNMT3A_0.1" "IDH1_0.1"   "IDH2_0.1
"   "JAK2_0.1"   "KDM6A_0.1" 
##  [9] "KMT2C_0.1"  "KRAS_0.1"   "NF1_0.1"    "PHF6_0.1"   "PTPN11_0.1" "RUNX1_0.
1"  "SF3B1_0.1"  "SRSF2_0.1" 
## [17] "TET2_0.1"   "TP53_0.1"   "U2AF1_0.1"  "age_10"     "gender"
```

```
torontoSurv <- Surv(torontoData$fu_years, torontoData$Diagnosis=="AML")
plot(survfit(torontoSurv~ 1), col= "black", main = "DC", xlab='Time after first sa
mple (years)', ylab='AML-free fraction', bty='L', yaxs='i', ylim=c(0,1.01), mark.t
ime = T)
```



```
plot(survfit(torontoSurv ~ torontoData$Diagnosis), xlab='Time after first sample (
years)', main = "DC", ylab='AML-free fraction', bty='L', yaxs='i', ylim=c(0,1.01),
mark.time = T, col = set1[1:2])
```



10.1 Validation cohort

all 37 pre-AML samples including overlap with DC

```
f = "./arch_data/VC_vaf_matrix_262ctrl_37aml.csv"
sangerData <- read.csv(f)

sangerData$hcdate <- as.Date(sangerData$hcdate)
sangerData$dodx <- as.Date(sangerData$dodx)

sangerPatients <- sub("[a-z]+$","", sangerData$Sample)
o <- order(sangerPatients, as.numeric(sangerData$hcdate))

sangerData <- sangerData[o,]
sangerPatients <- sangerPatients[o]

clinical_vars <- c("systol", "diastol", "bmi", "cholestl", "triglyc", "hdl", "ldl"
, "lym", "mcv", "rdw", "wbc", "plt", "hgb")
sangerX <- sangerData[, colnames(sangerData) %in% c(gene_vars, "age","gender",clin
ical_vars)]
sangerX <- as.data.frame(sangerX)

sangerX <- sangerX[,colSums(sangerX != 0,na.rm=TRUE)>=thr]
sangerGroups <- factor(grepl("^[a-z]", colnames(sangerX))*2, levels=0:2, labels=c(
"Genes", "Demographics", "Blood"))
sangerGroups[names(sangerX) %in% c("age","gender")] <- "Demographics"
table(sangerGroups)
```

```
## sangerGroups
##        Genes Demographics        Blood 
##           15            2           13
```

```
colnames(sangerX)
```

```
##  [1] "ASXL1"    "CBL"      "DNMT3A"   "JAK2"     "KMT2C"    "KMT2D"    "KRAS"
"NF1"      "NRAS"     "RAD21"
## [11] "SF3B1"    "SRSF2"    "TET2"     "TP53"     "U2AF1"    "age"      "gender"
"systol"   "diastol"  "bmi"
## [21] "cholestl" "triglyc"  "hdl"      "ldl"      "lym"      "mcv"      "rdw"
"wbc"      "plt"      "hgb"
```

```
sangerGroups
```

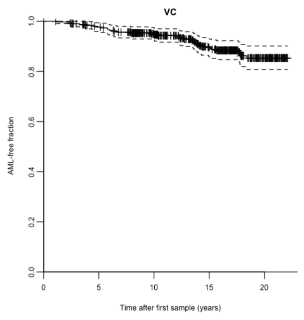
```
##  [1] Genes        Genes        Genes        Genes        Genes        Genes
Genes        Genes
##  [9] Genes        Genes        Genes        Genes        Genes        Genes
Genes        Demographics
## [17] Demographics Blood        Blood        Blood        Blood        Blood
Blood        Blood
## [25] Blood        Blood        Blood        Blood        Blood        Blood
## Levels: Genes Demographics Blood
```

```
g <- sangerGroups=="Genes"
sangerX[g] <- sangerX[g] * 10
names(sangerX)[g] <- paste(names(sangerX[g]),"0.1", sep="_")
y <- StandardizeMagnitude(sangerX[!g])
sangerX <- cbind(sangerX[g],y)

poorMansImpute <- function(x) {x[is.na(x)] <- mean(x, na.rm=TRUE); return(x)}
sangerX <- as.data.frame(sapply(sangerX, poorMansImpute))

foo <- split(sangerData[,c("Diagnosis","hcdate","dodx")], sangerPatients)

bar <- do.call("rbind",lapply(foo, function(x){
  y <- x
  n <- nrow(y)
  y[-n,"Diagnosis"] <- "Control"
  start <- as.numeric(y$hcdate - y$hcdate[1])/365.25
  end <- c(as.numeric(y$hcdate - y$hcdate[1])[-1]/365.25, as.numeric(y$dodx[n] - y
$hcdate[1])/365.25)
  return(data.frame(Diagnosis=y[,"Diagnosis"], start=start, end=end))
}))

bar[1:10, ]
```

	Diagnosis <fctr>	start <dbl>	end <dbl>
PD29762	AML	0.000000	9.754962
PD29764	AML	0.000000	10.360027
PD29792	AML	0.000000	14.108145
PD29804	Control	0.000000	5.138946
PD29810	Control	0.000000	18.573580
PD29836.1	Control	0.000000	2.414784
PD29836.2	AML	2.414784	10.023272
PD29851.1	Control	0.000000	4.599589
PD29851.2	AML	4.599589	12.205339
PD29856.1	Control	0.000000	4.331280

1-10 of 10 rows

```
sangerPatientsSplit <- unlist(sapply(names(foo), function(n) rep(n, nrow(foo[[n]])
)))

sangerSurv <- Surv(time = bar$start, time2 = bar$end, event = bar$Diagnosis!="Cont
rol", origin = 0)

plot(survfit(sangerSurv~ 1), col= "black", main = "VC", xlab='Time after first sam
ple (years)', ylab='AML-free fraction', bty='L', yaxs='i', ylim=c(0,1.01), mark.ti
me = T) #mark = 1
```



10.2 Expected AML incidence

Validation cohort

```
w <- c(which(sangerSurv[,1]==0)[-1]-1, nrow(sangerSurv))
sangerSurv2 <- Surv(sangerSurv[w,2], sangerSurv[w,3])

expected_rate_sanger_cr <- mean(aml_inc_cr(sangerX[w,"gender"],sangerX[w,"age_10"]
*10, sangerX[w,"age_10"]*10+ pmax(1,sangerSurv2[,1]))[!sangerSurv2[,2]])

n_total_sanger <- sum(sangerSurv2[,2])/expected_rate_sanger_cr
n_total_sanger
```

```
## [1] 13277.44
```

Discovery cohort only

```
expected_rate_toronto_cr <- mean(aml_inc_cr(torontoX[,"gender"],torontoX[,"age_10"
]*10, torontoX[,"age_10"]*10+ pmax(1,torontoSurv[,1]))[!torontoSurv[,2]])

n_total_toronto <- sum(torontoSurv[,2])/expected_rate_toronto_cr
n_total_toronto
```

```
## [1] 66014.85
```

10.3 Combined data

Survival

```
allSurv <- rbind(sangerSurv, Surv(rep(0, nrow(torontoSurv)), torontoSurv[,1], toro
ntoSurv[,2]))
allSurv <- Surv(allSurv[,1], allSurv[,2], allSurv[,3])
```

Data matrix

```
cohort <- c(rep("Sanger", nrow(sangerX)), rep("Toronto", nrow(torontoX)))
i <- c(sort(setdiff(gene_vars,"CALR")),"age","gender")
allX <- rbind(superSet(sangerData,i,fill=0), superSet(torontoData,i,fill=0))
allX <- allX[,colSums(allX>0)>=thr]
allX <- cbind(allX, cohort=cohort=="Sanger") + 0
allGroups <- factor(grepl("^[A-Z]",colnames(allX))+0, levels=1:0, labels=c("Genes"
,"Demographics"))

g <- allGroups=="Genes"
allX <- cbind(10*allX[,g], StandardizeMagnitude(allX[,!g]))
colnames(allX)[g] <- paste(colnames(allX)[g],"0.1",sep="_")
control <- c(sangerData$Diagnosis=="Control", torontoData$Diagnosis=="Control")
```

Weights

```
weights <- rep(1, nrow(allX))
weights[cohort=="Sanger" & control] <- n_total_sanger/sum(cohort=="Sanger" & contr
ol & allSurv[,1]==0)
weights[cohort=="Toronto" & control] <- n_total_toronto/sum(cohort=="Toronto" & co
ntrol)

n_total <- n_total_sanger + n_total_toronto
n_total
```

```
## [1] 79292.3
```

10.4 Coxph model fits

```
sigma0 <- 0.1
nu <- 1
which.mu <- "Genes"
```

10.4.1 Discovery cohort

10.4.1.1 Raw

```
fitToronto <- CoxRFX(torontoX, torontoSurv, groups=torontoGroups, which.mu=which.m
u, nu=nu, sigma0=sigma0)
waldToronto <- WaldTest(fitToronto)
```

```
##                    group    coef   coef-mu     sd      z df  p.value sig
## ASXL1_0.1          Genes  0.6922  0.049613 0.1172  5.908  1 3.47e-09 ***
## CALR_0.1           Genes  0.6239 -0.018696 0.0710  8.784  1 1.58e-18 ***
## CBL_0.1            Genes  0.5335 -0.109028 0.1293  4.126  1 3.70e-05 ***
## DNMT3A_0.1         Genes  0.5843 -0.058207 0.1059  5.517  1 3.44e-08 ***
## IDH1_0.1           Genes  0.6912  0.048657 0.1245  5.550  1 2.86e-08 ***
## IDH2_0.1           Genes  0.5136 -0.128999 0.1151  4.460  1 8.19e-06 ***
## JAK2_0.1           Genes  0.7120  0.069470 0.1243  5.730  1 1.00e-08 ***
## KDM6A_0.1          Genes  0.6419 -0.000647 0.0590 10.887  1 1.32e-27 ***
## KMT2C_0.1          Genes  0.6658  0.023265 0.0621 10.725  1 7.79e-27 ***
## KRAS_0.1           Genes  0.6403 -0.002210 0.0590 10.855  1 1.89e-27 ***
## NF1_0.1            Genes  0.6412 -0.001393 0.0590 10.870  1 1.61e-27 ***
## PHF6_0.1           Genes  0.6475  0.004993 0.0595 10.891  1 1.27e-27 ***
## PTPN11_0.1         Genes  0.6595  0.016950 0.0592 11.145  1 7.57e-29 ***
## RUNX1_0.1          Genes  0.4100 -0.232587 0.0923  4.443  1 8.89e-06 ***
## SF3B1_0.1          Genes  0.7728  0.130235 0.1019  7.585  1 3.33e-14 ***
## SRSF2_0.1          Genes  0.4783 -0.164235 0.0945  5.062  1 4.16e-07 ***
## TET2_0.1           Genes  0.6389 -0.003667 0.1295  4.932  1 8.13e-07 ***
## TP53_0.1           Genes  0.8079  0.165351 0.0673 12.009  1 3.19e-33 ***
## U2AF1_0.1          Genes  0.8537  0.211135 0.0773 11.048  1 2.23e-28 ***
## age_10      Demographics -0.0836 -0.083628 0.0975 -0.858  1 3.91e-01
## gender      Demographics  0.0113  0.011327 0.1091  0.104  1 9.17e-01
```

```
survConcordance(fitToronto$surv ~ fitToronto$linear.predictors)
```

```
## Call:
## survConcordance(formula = fitToronto$surv ~ fitToronto$linear.predictors)
##
##   n= 497
## Concordance= 0.7538671 se= 0.03218546
## concordant discordant  tied.risk  tied.time   std(c-d)
##   26561.00    8672.00       0.00       1.00    2267.98
```

10.4.2 Validation cohort

10.4.2.1 Raw

```
fitSanger <- CoxRFX(sangerX, sangerSurv, groups=sangerGroups, which.mu=which.mu, n
u=nu, sigma0=sigma0)
waldSanger <- WaldTest(fitSanger)
```

```
##                      group     coef   coef-mu      sd      z df  p.value sig
## ASXL1_0.1          Genes  0.64051  0.105357 0.11285  5.676  1 1.38e-08 ***
## CBL_0.1            Genes  0.52291 -0.012246 0.08720  5.997  1 2.01e-09 ***
## DNMT3A_0.1         Genes  0.43301 -0.102144 0.11026  3.927  1 8.60e-05 ***
## JAK2_0.1           Genes  0.52046 -0.014699 0.09655  5.391  1 7.02e-08 ***
## KMT2C_0.1          Genes  0.54634  0.011184 0.08151  6.703  1 2.05e-11 ***
## KMT2D_0.1          Genes  0.42573 -0.109421 0.14122  3.015  1 2.57e-03  **
## KRAS_0.1           Genes  0.53897  0.003816 0.08013  6.726  1 1.74e-11 ***
## NF1_0.1            Genes  0.52911 -0.006044 0.08135  6.504  1 7.80e-11 ***
## NRAS_0.1           Genes  0.53431 -0.000849 0.08011  6.670  1 2.56e-11 ***
## RAD21_0.1          Genes  0.53226 -0.002897 0.08049  6.613  1 3.77e-11 ***
## SF3B1_0.1          Genes  0.53076 -0.004391 0.08104  6.550  1 5.76e-11 ***
## SRSF2_0.1          Genes  0.50357 -0.031583 0.11851  4.249  1 2.14e-05 ***
## TET2_0.1           Genes  0.58716  0.052000 0.10482  5.602  1 2.12e-08 ***
## TP53_0.1           Genes  0.58827  0.053119 0.08077  7.283  1 3.25e-13 ***
## U2AF1_0.1          Genes  0.59395  0.058796 0.08084  7.347  1 2.03e-13 ***
## age_10      Demographics  0.08031  0.080306 0.11847  0.678  1 4.98e-01
## gender      Demographics -0.11803 -0.118029 0.11360 -1.039  1 2.99e-01
## systol_100         Blood  0.01074  0.010736 0.04230  0.254  1 8.00e-01
## diastol_100        Blood  0.02297  0.022974 0.02697  0.852  1 3.94e-01
## bmi_10             Blood  0.09128  0.091285 0.07510  1.215  1 2.24e-01
## cholestl_10        Blood  0.00934  0.009343 0.01381  0.676  1 4.99e-01
## triglyc            Blood  0.02435  0.024354 0.09637  0.253  1 8.00e-01
## hdl                Blood -0.07521 -0.075205 0.07691 -0.978  1 3.28e-01
## ldl                Blood  0.12764  0.127641 0.09931  1.285  1 1.99e-01
## lym                Blood  0.07714  0.077135 0.09427  0.818  1 4.13e-01
## mcv_100            Blood -0.00987 -0.009867 0.00826 -1.195  1 2.32e-01
## rdw_10             Blood  0.06196  0.061956 0.02072  2.990  1 2.79e-03  **
## wbc_10             Blood  0.01894  0.018939 0.03734  0.507  1 6.12e-01
## plt_100            Blood  0.05344  0.053435 0.09405  0.568  1 5.70e-01
## hgb_10             Blood  0.05198  0.051979 0.02446  2.125  1 3.36e-02   *
```

```
survConcordance(sangerSurv ~ fitSanger$linear.predictors)
```

```
## Call:
## survConcordance(formula = sangerSurv ~ fitSanger$linear.predictors)
##
##   n= 459
## Concordance= 0.7224015 se= 0.04865039
## concordant discordant  tied.risk  tied.time   std(c-d)
##  6714.0000  2580.0000     0.0000     0.0000   904.3134
```

10.4.2.2 Adjusted

```
fitWeightedSanger <- CoxRFX(sangerX, sangerSurv, sangerGroups, which.mu=which.mu,
sigma0=sigma0, nu=nu, weights=weights[cohort=="Sanger"])
waldWeightedSanger <- WaldTest(fitWeightedSanger)
```

```
##                      group      coef   coef-mu      sd        z df  p.value sig
## ASXL1_0.1          Genes  2.634306  0.838861 0.43502  6.05558  1 1.40e-09 ***
## CBL_0.1            Genes  0.630557 -1.164888 1.13502  0.55555  1 5.79e-01
## DNMT3A_0.1         Genes  0.698827 -1.096619 0.22597  3.09251  1 1.98e-03  **
## JAK2_0.1           Genes  0.049363 -1.746082 0.90486  0.05455  1 9.56e-01
## KMT2C_0.1          Genes  1.829655  0.034210 1.05055  1.74162  1 8.16e-02   .
## KMT2D_0.1          Genes -0.004783 -1.800228 0.75790 -0.00631  1 9.95e-01
## KRAS_0.1           Genes  2.139544  0.344099 0.40749  5.25049  1 1.52e-07 ***
## NF1_0.1            Genes  1.252510 -0.542935 0.89204  1.40410  1 1.60e-01
## NRAS_0.1           Genes  1.730987 -0.064459 0.36379  4.75820  1 1.95e-06 ***
## RAD21_0.1          Genes  1.487062 -0.308383 0.68933  2.15726  1 3.10e-02   *
## SF3B1_0.1          Genes  1.309652 -0.485793 0.96376  1.35890  1 1.74e-01
## SRSF2_0.1          Genes  1.451418 -0.344027 0.27015  5.37269  1 7.76e-08 ***
## TET2_0.1           Genes  1.222954 -0.572491 0.12864  9.50695  1 1.96e-21 ***
## TP53_0.1           Genes  4.699561  2.904116 0.91319  5.14632  1 2.66e-07 ***
## U2AF1_0.1          Genes  5.800067  4.004622 0.74776  7.75664  1 8.72e-15 ***
## age_10      Demographics  0.024711  0.024711 0.12062  0.20487  1 8.38e-01
## gender      Demographics -0.140352 -0.140352 0.11358 -1.23575  1 2.17e-01
## systol_100         Blood -0.000324 -0.000324 0.04456 -0.00726  1 9.94e-01
## diastol_100        Blood  0.019654  0.019654 0.02894  0.67907  1 4.97e-01
## bmi_10             Blood  0.101555  0.101555 0.08137  1.24811  1 2.12e-01
## cholestl_10        Blood  0.007469  0.007469 0.01457  0.51275  1 6.08e-01
## triglyc            Blood  0.007316  0.007316 0.10707  0.06832  1 9.46e-01
## hdl                Blood -0.108973 -0.108973 0.08295 -1.31365  1 1.89e-01
## ldl                Blood  0.149658  0.149658 0.10397  1.43938  1 1.50e-01
## lym                Blood  0.066987  0.066987 0.09901  0.67660  1 4.99e-01
## mcv_100            Blood -0.015964 -0.015964 0.00832 -1.91787  1 5.51e-02   .
## rdw_10             Blood  0.073201  0.073201 0.01789  4.09058  1 4.30e-05 ***
## wbc_10             Blood  0.020190  0.020190 0.04345  0.46465  1 6.42e-01
## plt_100            Blood  0.077199  0.077199 0.10027  0.76987  1 4.41e-01
## hgb_10             Blood  0.044376  0.044376 0.02513  1.76558  1 7.75e-02   .
```

```
survConcordance(sangerSurv ~ fitWeightedSanger$linear.predictors, weights=weights[
cohort=="Sanger"])
```

```
## Call:
## survConcordance(formula = sangerSurv ~ fitWeightedSanger$linear.predictors,
##     weights = weights[cohort == "Sanger"])
##
##   n= 459
## Concordance= 0.7639423 se= 0.04828991
## concordant discordant  tied.risk  tied.time   std(c-d)
##  334537.56  103371.88       0.00       0.00   42293.22
```

Uno’s estimator of cumulative/dynamic AUC

```
w <- c(which(sangerSurv[,1]==0)[-1]-1, nrow(sangerSurv))
s <- Surv(sangerSurv[w,2], sangerSurv[w,3])
a <- AUC.uno(s, s, fitWeightedSanger$linear.predictors[w], times= c(0, 22, 0.1))
round(a$iauc, digits = 3)
```

```
## [1] 0.761
```

11 Model excluding controls without mutations

Include only controls with ARCH & all pre-AML (regardless of mutation status) ## Discovery cohort (Toronto) Data

```
f = "./arch_data/DC_vaf_matrix_no_duplicates_414ctrl_83aml.csv"
torontoData <- read.csv(f)

gene_vars <- c("CALR", "NRAS", "DNMT3A", "SF3B1", "IDH1", "KIT", "TET2", "RAD21",
"JAK2", "CBL", "KRAS", "PTPN11", "IDH2", "TP53", "NF1", "SRSF2", "CEBPA", "ASXL1",
"RUNX1", "U2AF1", "BCOR", "KDM6A", "PHF6", "KMT2C", "KMT2D")

table(torontoData$Diagnosis)
```

```
## 
##     AML Control 
##      83     414
```

```
torontoData$gender <- ifelse(torontoData$Sex == "male", 1,
                             ifelse(torontoData$Sex == "female", 0, torontoData$Se
x))
dim(torontoData)
```

```
## [1] 497  29
```

```
torontoData <- torontoData[rowSums(torontoData[, colnames(torontoData) %in% gene_v
ars])>0 | torontoData$Diagnosis == "AML", ]
dim(torontoData)
```

```
## [1] 240  29
```

```
table(torontoData$gender)
```

```
## 
##   0   1 
## 135 105
```

```
torontoData$gender <- as.numeric(torontoData$gender)
colnames(torontoData)
```

```
##  [1] "Sample"     "ASXL1"      "BCOR"       "CALR"       "CBL"        "DNMT3A"    
"IDH1"       "IDH2"      
##  [9] "JAK2"       "KDM6A"      "KIT"        "KMT2C"      "KRAS"       "NF1"       
"NRAS"       "PHF6"      
## [17] "PTPN11"     "RUNX1"      "SF3B1"      "SRSF2"      "TET2"       "TP53"      
"U2AF1"      "Diagnosis" 
## [25] "fu_years"   "age"        "Sex"        "no_drivers" "gender"
```

Manually standardize magnitudes

```
torontoData <- torontoData[!duplicated(torontoData),]

torontoX <- torontoData[, colnames(torontoData) %in% c(gene_vars, "age", "gender")
]

torontoX <- as.data.frame(torontoX)
thr <- 2
torontoX <- torontoX[,colSums(torontoX != 0)>=thr]

torontoGroups <- factor(names(torontoX) %in% c("age","gender")+1, level=1:2, label
s=c("Genes","Demographics"))
colnames(torontoX)
```

```
##  [1] "ASXL1"  "CALR"   "CBL"    "DNMT3A" "IDH1"   "IDH2"   "JAK2"   "KDM6A"  "K
MT2C"  "KRAS"   "NF1"    "PHF6"  
## [13] "PTPN11" "RUNX1"  "SF3B1"  "SRSF2"  "TET2"   "TP53"   "U2AF1"  "age"    "g
ender"
```

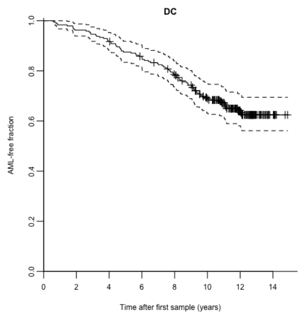
```
torontoGroups
```

```
##  [1] Genes        Genes        Genes        Genes        Genes        Genes       
Genes        Genes       
##  [9] Genes        Genes        Genes        Genes        Genes        Genes       
Genes        Genes       
## [17] Genes        Genes        Genes        Demographics Demographics
## Levels: Genes Demographics
```

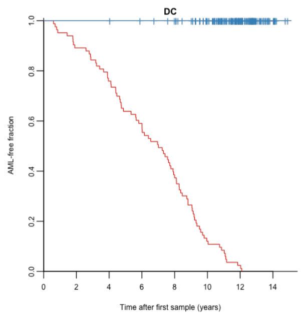
```
# Manually standardize age and mutation VAFs
torontoX$age <- torontoX$age/10
names(torontoX)[which(names(torontoX)=="age")] <- "age_10"
g <- torontoGroups == "Genes"
torontoX[,g] <- torontoX[,g]*10
names(torontoX)[g] <- paste(names(torontoX)[g], "0.1",sep="_")
colnames(torontoX)
```

```
##  [1] "ASXL1_0.1"   "CALR_0.1"    "CBL_0.1"     "DNMT3A_0.1" "IDH1_0.1"    "IDH2_0.1
"    "JAK2_0.1"    "KDM6A_0.1"
##  [9] "KMT2C_0.1"   "KRAS_0.1"    "NF1_0.1"     "PHF6_0.1"    "PTPN11_0.1" "RUNX1_0.
1"   "SF3B1_0.1"   "SRSF2_0.1"
## [17] "TET2_0.1"    "TP53_0.1"    "U2AF1_0.1"   "age_10"      "gender"
```

```
torontoSurv <- Surv(torontoData$fu_years, torontoData$Diagnosis=="AML")
plot(survfit(torontoSurv~ 1), col= "black", main = "DC", xlab='Time after first sa
mple (years)', ylab='AML-free fraction', bty='L', yaxs='i', ylim=c(0,1.01), mark.t
ime = T)
```



```
plot(survfit(torontoSurv ~ torontoData$Diagnosis), xlab='Time after first sample (
years)', main = "DC", ylab='AML-free fraction', bty='L', yaxs='i', ylim=c(0,1.01),
mark.time = T, col = set1[1:2])
```



11.1 Validation cohort

```
f = "./arch_data/VC_vaf_matrix_262ctrl_37aml.csv"
sangerData <- read.csv(f)
dim(sangerData)
```

```
## [1] 459  52
```

```
sangerData <- sangerData[rowSums(sangerData[, colnames(sangerData) %in% gene_vars]
)>0 | sangerData$Diagnosis == "AML", ]
dim(sangerData)
```

```
## [1] 173  52
```

```
sangerData$hcdate <- as.Date(sangerData$hcdate)
sangerData$dodx <- as.Date(sangerData$dodx)

sangerPatients <- sub("[a-z]+$","", sangerData$Sample)
o <- order(sangerPatients, as.numeric(sangerData$hcdate))

sangerData <- sangerData[o,]
sangerPatients <- sangerPatients[o]

clinical_vars <- c("systol", "diastol", "bmi", "cholestl", "triglyc", "hdl", "ldl"
, "lym", "mcv", "rdw", "wbc", "plt", "hgb")
sangerX <- sangerData[, colnames(sangerData) %in% c(gene_vars, "age","gender",clin
ical_vars)]
sangerX <- as.data.frame(sangerX)

sangerX <- sangerX[,colSums(sangerX != 0,na.rm=TRUE)>=thr]
sangerGroups <- factor(grepl("^[a-z]", colnames(sangerX))*2, levels=0:2, labels=c(
"Genes", "Demographics", "Blood"))
sangerGroups[names(sangerX) %in% c("age","gender")] <- "Demographics"
table(sangerGroups)
```

```
## sangerGroups
##        Genes Demographics        Blood 
##           15            2           13
```

```
colnames(sangerX)
```

```
##  [1] "ASXL1"    "CBL"      "DNMT3A"   "JAK2"     "KMT2C"    "KMT2D"    "KRAS"    
"NF1"      "NRAS"     "RAD21"   
## [11] "SF3B1"    "SRSF2"    "TET2"     "TP53"     "U2AF1"    "age"      "gender"  
"systol"   "diastol"  "bmi"     
## [21] "cholestl" "triglyc"  "hdl"      "ldl"      "lym"      "mcv"      "rdw"     
"wbc"      "plt"      "hgb"
```

```
sangerGroups
```

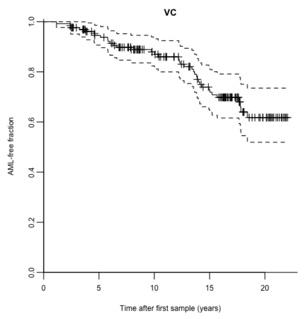
```
##  [1] Genes        Genes        Genes        Genes        Genes        Genes
Genes        Genes
##  [9] Genes        Genes        Genes        Genes        Genes        Genes
Genes        Demographics
## [17] Demographics Blood        Blood        Blood        Blood        Blood
Blood        Blood
## [25] Blood        Blood        Blood        Blood        Blood        Blood
## Levels: Genes Demographics Blood
```

```
g <- sangerGroups=="Genes"
sangerX[g] <- sangerX[g] * 10
names(sangerX)[g] <- paste(names(sangerX[g]),"0.1", sep="_")
y <- StandardizeMagnitude(sangerX[!g])
sangerX <- cbind(sangerX[g],y)

poorMansImpute <- function(x) {x[is.na(x)] <- mean(x, na.rm=TRUE); return(x)}
sangerX <- as.data.frame(sapply(sangerX, poorMansImpute))

foo <- split(sangerData[,c("Diagnosis","hcdate","dodx")], sangerPatients)

bar <- do.call("rbind",lapply(foo, function(x){
  y <- x
  n <- nrow(y)
  y[-n,"Diagnosis"] <- "Control"
  start <- as.numeric(y$hcdate - y$hcdate[1])/365.25
  end <- c(as.numeric(y$hcdate - y$hcdate[1])[-1]/365.25, as.numeric(y$dodx[n] - y
$hcdate[1])/365.25)
  return(data.frame(Diagnosis=y[,"Diagnosis"], start=start, end=end))
}))

bar[1:10, ]
```

	Diagnosis	start	end
	<fctr>	<dbl>	<dbl>
PD29762	AML	0.000000	9.754962
PD29764	AML	0.000000	10.360027
PD29792	AML	0.000000	14.108145
PD29810	Control	0.000000	18.573580
PD29836.1	Control	0.000000	2.414784
PD29836.2	AML	2.414784	10.023272
PD29851.1	Control	0.000000	4.599589
PD29851.2	AML	4.599589	12.205339
PD29856.1	Control	0.000000	4.331280
PD29856.2	AML	4.331280	17.828884

1-10 of 10 rows

```
sangerPatientsSplit <- unlist(sapply(names(foo), function(n) rep(n, nrow(foo[[n]])
)))

sangerSurv <- Surv(time = bar$start, time2 = bar$end, event = bar$Diagnosis!="Cont
rol", origin = 0)

plot(survfit(sangerSurv~ 1), col= "black", main = "VC", xlab='Time after first sam
ple (years)', ylab='AML-free fraction', bty='L', yaxs='i', ylim=c(0,1.01), mark.ti
me = T) #mark = 1
```



11.2 Expected AML incidence

Validation cohort

```
w <- c(which(sangerSurv[,1]==0)[-1]-1, nrow(sangerSurv))
sangerSurv2 <- Surv(sangerSurv[w,2], sangerSurv[w,3]) ## Unique individuals

expected_rate_sanger_cr <- mean(aml_inc_cr(sangerX[w,"gender"],sangerX[w,"age_10"]
*10, sangerX[w,"age_10"]*10+ pmax(1,sangerSurv2[,1]))[!sangerSurv2[,2]])

n_total_sanger <- sum(sangerSurv2[,2])/expected_rate_sanger_cr
n_total_sanger
```

```
## [1] 14208.3
```

Discovery cohort

```
expected_rate_toronto_cr <- mean(aml_inc_cr(torontoX[,"gender"],torontoX[,"age_10"
]*10, torontoX[,"age_10"]*10+ pmax(1,torontoSurv[,1]))[!torontoSurv[,2]])

n_total_toronto <- sum(torontoSurv[,2])/expected_rate_toronto_cr
n_total_toronto
```

```
## [1] 55688.66
```

11.3 Combined data

Survival

```
allSurv <- rbind(sangerSurv, Surv(rep(0, nrow(torontoSurv)), torontoSurv[,1], toro
ntoSurv[,2]))
allSurv <- Surv(allSurv[,1], allSurv[,2], allSurv[,3])
```

Data matrix

```
cohort <- c(rep("Sanger", nrow(sangerX)), rep("Toronto", nrow(torontoX)))
i <- c(sort(setdiff(gene_vars,"CALR")),"age","gender")
allX <- rbind(superSet(sangerData,i,fill=0), superSet(torontoData,i,fill=0))
allX <- allX[,colSums(allX>0)>=thr]
allX <- cbind(allX, cohort=cohort=="Sanger") + 0
allGroups <- factor(grepl("^[A-Z]",colnames(allX))+0, levels=1:0, labels=c("Genes"
,"Demographics"))

g <- allGroups=="Genes"
allX <- cbind(10*allX[,g], StandardizeMagnitude(allX[,!g]))
colnames(allX)[g] <- paste(colnames(allX)[g],"0.1",sep="_")
control <- c(sangerData$Diagnosis=="Control", torontoData$Diagnosis=="Control")
```

Weights

```
weights <- rep(1, nrow(allX))
weights[cohort=="Sanger" & control] <- n_total_sanger/sum(cohort=="Sanger" & contr
ol & allSurv[,1]==0)
weights[cohort=="Toronto" & control] <- n_total_toronto/sum(cohort=="Toronto" & co
ntrol)

n_total <- n_total_sanger + n_total_toronto
n_total
```

```
## [1] 69896.97
```

11.4 Coxph model fits

```
sigma0 <- 0.1
nu <- 1
which.mu <- "Genes"
```

11.4.1 DC

11.4.1.1 Raw

```
fitToronto <- CoxRFX(torontoX, torontoSurv, groups=torontoGroups, which.mu=which.m
u, nu=nu, sigma0=sigma0)
waldToronto <- WaldTest(fitToronto)
```

```
##                    group    coef   coef-mu     sd      z df  p.value sig
## ASXL1_0.1          Genes  0.4801  0.050389 0.1108  4.335  1 1.46e-05 ***
## CALR_0.1           Genes  0.4076 -0.022055 0.0700  5.824  1 5.76e-09 ***
## CBL_0.1            Genes  0.3119 -0.117817 0.1151  2.710  1 6.72e-03  **
## DNMT3A_0.1         Genes  0.3010 -0.128687 0.1054  2.857  1 4.28e-03  **
## IDH1_0.1           Genes  0.4535  0.023828 0.1092  4.152  1 3.29e-05 ***
## IDH2_0.1           Genes  0.3789 -0.050806 0.1052  3.602  1 3.15e-04 ***
## JAK2_0.1           Genes  0.4956  0.065922 0.1136  4.364  1 1.28e-05 ***
## KDM6A_0.1          Genes  0.4288 -0.000932 0.0594  7.214  1 5.45e-13 ***
## KMT2C_0.1          Genes  0.4450  0.015284 0.0619  7.194  1 6.28e-13 ***
## KRAS_0.1           Genes  0.4257 -0.004039 0.0595  7.156  1 8.31e-13 ***
## NF1_0.1            Genes  0.4272 -0.002451 0.0595  7.183  1 6.80e-13 ***
## PHF6_0.1           Genes  0.4321  0.002404 0.0598  7.230  1 4.83e-13 ***
## PTPN11_0.1         Genes  0.4414  0.011735 0.0596  7.407  1 1.29e-13 ***
## RUNX1_0.1          Genes  0.2761 -0.153642 0.0890  3.102  1 1.92e-03  **
## SF3B1_0.1          Genes  0.5346  0.104912 0.0892  5.993  1 2.06e-09 ***
## SRSF2_0.1          Genes  0.3772 -0.052539 0.0883  4.274  1 1.92e-05 ***
## TET2_0.1           Genes  0.4247 -0.005040 0.1174  3.617  1 2.98e-04 ***
## TP53_0.1           Genes  0.5441  0.114421 0.0665  8.181  1 2.81e-16 ***
## U2AF1_0.1          Genes  0.5788  0.149112 0.0722  8.015  1 1.10e-15 ***
## age_10      Demographics -0.3093 -0.309301 0.1116 -2.771  1 5.59e-03  **
## gender      Demographics -0.0253 -0.025329 0.1385 -0.183  1 8.55e-01
```

```
survConcordance(fitToronto$surv ~ fitToronto$linear.predictors, weights = weights[
cohort=="Toronto"])
```

```
## Call:
## survConcordance(formula = fitToronto$surv ~ fitToronto$linear.predictors,
##     weights = weights[cohort == "Toronto"])
##
##   n= 240
## Concordance= 0.7539084 se= 0.03193557
## concordant discordant  tied.risk  tied.time   std(c-d)
##  3255935.4  1062805.9        0.0        1.0   275842.9
```

11.4.1.2 Adjusted

```
fitWeightedToronto <- CoxRFX(torontoX, torontoSurv, torontoGroups, which.mu=which.
mu, sigma0=sigma0, nu=nu, weights=weights[cohort=="Toronto"])
waldWeightedToronto <- WaldTest(fitWeightedToronto)
```

```
##                    group    coef coef-mu    sd       z df  p.value sig
## ASXL1_0.1          Genes  1.9719  0.1365 0.150 13.1816  1 1.12e-39 ***
## CALR_0.1           Genes -0.0794 -1.9147 1.174 -0.0676  1 9.46e-01
## CBL_0.1            Genes  0.0165 -1.8188 0.426  0.0388  1 9.69e-01
## DNMT3A_0.1         Genes  0.3722 -1.4631 0.153  2.4301  1 1.51e-02   *
## IDH1_0.1           Genes  2.3375  0.5022 0.350  6.6815  1 2.36e-11 ***
## IDH2_0.1           Genes  0.5915 -1.2438 0.240  2.4621  1 1.38e-02   *
## JAK2_0.1           Genes  1.7762 -0.0592 0.193  9.2213  1 2.94e-20 ***
## KDM6A_0.1          Genes  1.6689 -0.1664 0.362  4.6081  1 4.06e-06 ***
## KMT2C_0.1          Genes -1.2330 -3.0683 1.191 -1.0356  1 3.00e-01
## KRAS_0.1           Genes  0.9875 -0.8478 0.555  1.7785  1 7.53e-02   .
## NF1_0.1            Genes  1.3623 -0.4730 0.501  2.7193  1 6.54e-03  **
## PHF6_0.1           Genes  2.6990  0.8636 0.255 10.5887  1 3.36e-26 ***
## PTPN11_0.1         Genes  3.6339  1.7986 0.723  5.0228  1 5.09e-07 ***
## RUNX1_0.1          Genes  0.6233 -1.2120 0.136  4.5906  1 4.42e-06 ***
## SF3B1_0.1          Genes  3.1088  1.2735 0.305 10.1981  1 2.02e-24 ***
## SRSF2_0.1          Genes  1.4956 -0.3397 0.172  8.6791  1 3.99e-18 ***
## TET2_0.1           Genes  0.5772 -1.2581 0.232  2.4920  1 1.27e-02   *
## TP53_0.1           Genes  8.9422  7.1069 0.823 10.8665  1 1.66e-27 ***
## U2AF1_0.1          Genes  4.0190  2.1836 0.384 10.4738  1 1.14e-25 ***
## age_10      Demographics -0.5274 -0.5274 0.135 -3.9171  1 8.96e-05 ***
## gender      Demographics  0.0323  0.0323 0.175  0.1842  1 8.54e-01
```

```
survConcordance(fitWeightedToronto$surv ~ fitWeightedToronto$linear.predictors, we
ights=weights[cohort=="Toronto"])
```

```
## Call:
## survConcordance(formula = fitWeightedToronto$surv ~ fitWeightedToronto$linear.p
redictors,
##     weights = weights[cohort == "Toronto"])
##
##   n= 240
## Concordance= 0.7701663 se= 0.03193557
## concordant discordant  tied.risk  tied.time   std(c-d)
##  3326148.9   992592.4        0.0        1.0   275842.9
```

```
#Uno's estimator of cumulative/dynamic AUC
a <- AUC.uno(torontoSurv, torontoSurv, fitWeightedToronto$linear.predictors, times
= seq(0,12, 0.1))
round(a$iauc, digits = 3)
```

```
## [1] 0.756
```

11.4.2 Validation cohort

11.4.2.1 Raw

```
fitSanger <- CoxRFX(sangerX, sangerSurv, groups=sangerGroups, which.mu=which.mu, n
u=nu, sigma0=sigma0)
waldSanger <- WaldTest(fitSanger)
```

```
##                     group     coef   coef-mu      sd       z df  p.value sig
## ASXL1_0.1           Genes  0.41389  1.04e-01 0.13253  3.1229  1 1.79e-03  **
## CBL_0.1             Genes  0.27978 -3.01e-02 0.10678  2.6202  1 8.79e-03  **
## DNMT3A_0.1          Genes  0.15476 -1.55e-01 0.12703  1.2183  1 2.23e-01
## JAK2_0.1            Genes  0.33012  2.02e-02 0.10874  3.0359  1 2.40e-03  **
## KMT2C_0.1           Genes  0.30175 -8.17e-03 0.09722  3.1037  1 1.91e-03  **
## KMT2D_0.1           Genes  0.14350 -1.66e-01 0.15722  0.9127  1 3.61e-01
## KRAS_0.1            Genes  0.30998  5.67e-05 0.09168  3.3811  1 7.22e-04 ***
## NF1_0.1             Genes  0.29225 -1.77e-02 0.09499  3.0768  1 2.09e-03  **
## NRAS_0.1            Genes  0.30685 -3.07e-03 0.09158  3.3507  1 8.06e-04 ***
## RAD21_0.1           Genes  0.29301 -1.69e-02 0.09373  3.1261  1 1.77e-03  **
## SF3B1_0.1           Genes  0.29894 -1.10e-02 0.09393  3.1825  1 1.46e-03  **
## SRSF2_0.1           Genes  0.40493  9.50e-02 0.13441  3.0125  1 2.59e-03  **
## TET2_0.1            Genes  0.37910  6.92e-02 0.11275  3.3624  1 7.73e-04 ***
## TP53_0.1            Genes  0.36746  5.75e-02 0.09308  3.9479  1 7.88e-05 ***
## U2AF1_0.1           Genes  0.37254  6.26e-02 0.09357  3.9813  1 6.85e-05 ***
## age_10       Demographics -0.01773 -1.77e-02 0.11451 -0.1548  1 8.77e-01
## gender       Demographics -0.03369 -3.37e-02 0.10501 -0.3208  1 7.48e-01
## systol_100          Blood  0.00145  1.45e-03 0.03839  0.0377  1 9.70e-01
## diastol_100         Blood  0.00773  7.73e-03 0.02329  0.3321  1 7.40e-01
## bmi_10              Blood  0.06828  6.83e-02 0.07091  0.9628  1 3.36e-01
## cholestl_10         Blood  0.01797  1.80e-02 0.01274  1.4109  1 1.58e-01
## triglyc             Blood  0.00471  4.71e-03 0.09569  0.0492  1 9.61e-01
## hdl                 Blood -0.00891 -8.91e-03 0.07257 -0.1227  1 9.02e-01
## ldl                 Blood  0.16056  1.61e-01 0.09725  1.6510  1 9.87e-02   .
## lym                 Blood -0.02015 -2.01e-02 0.08835 -0.2280  1 8.20e-01
## mcv_100             Blood -0.00369 -3.69e-03 0.00786 -0.4694  1 6.39e-01
## rdw_10              Blood  0.05420  5.42e-02 0.02080  2.6056  1 9.17e-03  **
## wbc_10              Blood  0.00379  3.79e-03 0.03521  0.1077  1 9.14e-01
## plt_100             Blood  0.03410  3.41e-02 0.09166  0.3720  1 7.10e-01
## hgb_10              Blood  0.03314  3.31e-02 0.02245  1.4763  1 1.40e-01
```

RDW p-val 9.171138e-03

```
survConcordance(sangerSurv ~ fitSanger$linear.predictors)
```

```
## Call:
## survConcordance(formula = sangerSurv ~ fitSanger$linear.predictors)
##
##   n= 173
## Concordance= 0.6611972 se= 0.05025086
## concordant discordant  tied.risk  tied.time   std(c-d)
##  2176.0000  1115.0000     0.0000     0.0000   330.7512
```

11.4.2.2 Adjusted

```
fitWeightedSanger <- CoxRFX(sangerX, sangerSurv, sangerGroups, which.mu=which.mu,
sigma0=sigma0, nu=nu, weights=weights[cohort=="Sanger"])
waldWeightedSanger <- WaldTest(fitWeightedSanger)
```

```
##                    group      coef   coef-mu      sd         z df  p.value sig
## ASXL1_0.1          Genes  2.580959  1.414558 0.47618  5.42008  1 5.96e-08 ***
## CBL_0.1            Genes -0.660213 -1.826614 1.39628 -0.47284  1 6.36e-01    
## DNMT3A_0.1         Genes  0.223151 -0.943251 0.24504  0.91066  1 3.62e-01    
## JAK2_0.1           Genes  0.705927 -0.460474 1.04486  0.67562  1 4.99e-01    
## KMT2C_0.1          Genes -0.385529 -1.551931 1.44435 -0.26692  1 7.90e-01    
## KMT2D_0.1          Genes -0.627231 -1.793633 1.03607 -0.60539  1 5.45e-01    
## KRAS_0.1           Genes  1.299133  0.132731 0.78999  1.64450  1 1.00e-01    
## NF1_0.1            Genes -0.815764 -1.982166 1.46470 -0.55695  1 5.78e-01    
## NRAS_0.1           Genes  0.728314 -0.438088 0.64251  1.13355  1 2.57e-01    
## RAD21_0.1          Genes -0.678392 -1.844793 1.44210 -0.47042  1 6.38e-01    
## SF3B1_0.1          Genes  0.072745 -1.093657 1.47708  0.04925  1 9.61e-01    
## SRSF2_0.1          Genes  1.726024  0.559622 0.23912  7.21826  1 5.27e-13 ***
## TET2_0.1           Genes  1.101278 -0.065124 0.15079  7.30320  1 2.81e-13 ***
## TP53_0.1           Genes  4.694801  3.528400 1.13074  4.15198  1 3.30e-05 ***
## U2AF1_0.1          Genes  7.530821  6.364419 1.06931  7.04270  1 1.89e-12 ***
## age_10      Demographics -0.190256 -0.190256 0.13151 -1.44666  1 1.48e-01    
## gender      Demographics -0.029742 -0.029742 0.12174 -0.24430  1 8.07e-01    
## systol_100         Blood -0.032537 -0.032537 0.04764 -0.68293  1 4.95e-01    
## diastol_100        Blood  0.000105  0.000105 0.02958  0.00356  1 9.97e-01    
## bmi_10             Blood  0.098774  0.098774 0.08970  1.10111  1 2.71e-01    
## cholestl_10        Blood  0.024226  0.024226 0.01553  1.55989  1 1.19e-01    
## triglyc            Blood  0.051097  0.051097 0.11392  0.44854  1 6.54e-01    
## hdl                Blood -0.082426 -0.082426 0.09326 -0.88380  1 3.77e-01    
## ldl                Blood  0.248075  0.248075 0.11127  2.22950  1 2.58e-02   *
## lym                Blood -0.054414 -0.054414 0.10621 -0.51234  1 6.08e-01    
## mcv_100            Blood -0.010783 -0.010783 0.00915 -1.17903  1 2.38e-01    
## rdw_10             Blood  0.095279  0.095279 0.01797  5.30078  1 1.15e-07 ***
## wbc_10             Blood  0.011314  0.011314 0.04898  0.23099  1 8.17e-01    
## plt_100            Blood  0.057755  0.057755 0.11248  0.51347  1 6.08e-01    
## hgb_10             Blood  0.016212  0.016212 0.02615  0.62004  1 5.35e-01    
```

```
survConcordance(sangerSurv ~ fitWeightedSanger$linear.predictors, weights=weights[
cohort=="Sanger"])
```

```
## Call:
## survConcordance(formula = sangerSurv ~ fitWeightedSanger$linear.predictors,
##     weights = weights[cohort == "Sanger"])
##
##   n= 173
## Concordance= 0.7231124 se= 0.0489519
## concordant discordant  tied.risk  tied.time   std(c-d)
##  296852.77  113668.16       0.00       0.00   40191.56
```

```
#Uno's estimator of cumulative/dynamic AUC
w <- c(which(sangerSurv[,1]==0)[-1]-1, nrow(sangerSurv))
s <- Surv(sangerSurv[w,2], sangerSurv[w,3])
a <- AUC.uno(s, s, fitWeightedSanger$linear.predictors[w], times= c(0, 22, 0.1))
round(a$iauc, digits = 3)
```

```
## [1] 0.403
```

12 CoxPH model excluding all samples without ARCH-PD

12.1 Discovery cohort

Data

```
f = "./arch_data/DC_vaf_matrix_414ctrl_91aml.csv"
torontoData <- read.csv(f)

gene_vars <- c("CALR", "NRAS", "DNMT3A", "SF3B1", "IDH1", "KIT", "TET2", "RAD21",
"JAK2", "CBL", "KRAS", "PTPN11", "IDH2", "TP53", "NF1", "SRSF2", "CEBPA", "ASXL1",
"RUNX1", "U2AF1", "BCOR", "KDM6A", "PHF6", "KMT2C", "KMT2D")

table(torontoData$Diagnosis)
```

```
##
##     AML Control
##      91     414
```

```
torontoData$gender <- ifelse(torontoData$Sex == "male", 1,
                             ifelse(torontoData$Sex == "female", 0, torontoData$Se
x))
dim(torontoData)
```

```
## [1] 505  29
```

```
torontoData <- torontoData[rowSums(torontoData[, colnames(torontoData) %in% gene_v
ars])>0, ]
dim(torontoData)
```

```
## [1] 221  29
```

```
table(torontoData$gender)
```

```
##
##   0   1
## 126  95
```

```
torontoData$gender <- as.numeric(torontoData$gender)
colnames(torontoData)
```

```
##  [1] "Sample"     "ASXL1"      "BCOR"       "CALR"       "CBL"        "DNMT3A"
"IDH1"       "IDH2"
##  [9] "JAK2"       "KDM6A"      "KIT"        "KMT2C"      "KRAS"       "NF1"
"NRAS"       "PHF6"
## [17] "PTPN11"     "RUNX1"      "SF3B1"      "SRSF2"      "TET2"       "TP53"
"U2AF1"      "Diagnosis"
## [25] "fu_years"   "age"        "Sex"        "no_drivers" "gender"
```

Manually standardize magnitudes

```
torontoData <- torontoData[!duplicated(torontoData),]

torontoX <- torontoData[, colnames(torontoData) %in% c(gene_vars, "age", "gender")
]

torontoX <- as.data.frame(torontoX)
thr <- 2
torontoX <- torontoX[,colSums(torontoX != 0)>=thr]

torontoGroups <- factor(names(torontoX) %in% c("age","gender")+1, level=1:2, label
s=c("Genes","Demographics"))
colnames(torontoX)
```

```
##  [1] "ASXL1"  "CALR"   "CBL"    "DNMT3A" "IDH1"   "IDH2"   "JAK2"   "KDM6A"  "K
MT2C"  "KRAS"   "NF1"    "PHF6"
## [13] "PTPN11" "RUNX1"  "SF3B1"  "SRSF2"  "TET2"   "TP53"   "U2AF1"  "age"    "g
ender"
```

```
torontoGroups
```

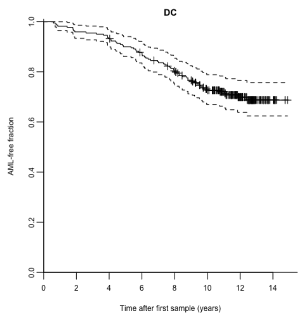
```
##  [1] Genes        Genes        Genes        Genes        Genes        Genes
Genes        Genes
##  [9] Genes        Genes        Genes        Genes        Genes        Genes
Genes        Genes
## [17] Genes        Genes        Genes        Demographics Demographics
## Levels: Genes Demographics
```

Manually standardize age and mutation VAFs

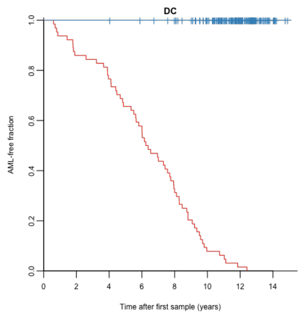
```
torontoX$age <- torontoX$age/10
names(torontoX)[which(names(torontoX)=="age")] <- "age_10"
g <- torontoGroups == "Genes"
torontoX[,g] <- torontoX[,g]*10
names(torontoX)[g] <- paste(names(torontoX)[g], "0.1",sep="_")
colnames(torontoX)
```

```
##  [1] "ASXL1_0.1"  "CALR_0.1"   "CBL_0.1"    "DNMT3A_0.1" "IDH1_0.1"   "IDH2_0.1
"   "JAK2_0.1"   "KDM6A_0.1"
##  [9] "KMT2C_0.1"  "KRAS_0.1"   "NF1_0.1"    "PHF6_0.1"   "PTPN11_0.1" "RUNX1_0.
1"  "SF3B1_0.1"  "SRSF2_0.1"
## [17] "TET2_0.1"   "TP53_0.1"   "U2AF1_0.1"  "age_10"     "gender"
```

```
torontoSurv <- Surv(torontoData$fu_years, torontoData$Diagnosis=="AML")
plot(survfit(torontoSurv~ 1), col= "black", main = "DC", xlab='Time after first sa
mple (years)', ylab='AML-free fraction', bty='L', yaxs='i', ylim=c(0,1.01), mark.t
ime = T)
```



```
plot(survfit(torontoSurv ~ torontoData$Diagnosis), xlab='Time after first sample (
years)', main = "DC", ylab='AML-free fraction', bty='L', yaxs='i', ylim=c(0,1.01),
mark.time = T, col = set1[1:2])
```



12.2 Validation cohort

```
f = "./arch_data/VC_vaf_matrix_no_duplicates_262ctrl_29aml.csv"
sangerData <- read.csv(f)
dim(sangerData)
```

```
## [1] 445  52
```

```
sangerData <- sangerData[rowSums(sangerData[, colnames(sangerData) %in% gene_vars]
)>0, ]
dim(sangerData)
```

```
## [1] 149  52
```

```
sangerData$hcdate <- as.Date(sangerData$hcdate)
sangerData$dodx <- as.Date(sangerData$dodx)

sangerPatients <- sub("[a-z]+$","", sangerData$Sample)
o <- order(sangerPatients, as.numeric(sangerData$hcdate))

sangerData <- sangerData[o,]
sangerPatients <- sangerPatients[o]

clinical_vars <- c("systol", "diastol", "bmi", "cholestl", "triglyc", "hdl", "ldl"
, "lym", "mcv", "rdw", "wbc", "plt", "hgb")
sangerX <- sangerData[, colnames(sangerData) %in% c(gene_vars, "age","gender",clin
ical_vars)]
sangerX <- as.data.frame(sangerX)

sangerX <- sangerX[,colSums(sangerX != 0,na.rm=TRUE)>=thr]
sangerGroups <- factor(grepl("^[a-z]", colnames(sangerX))*2, levels=0:2, labels=c(
"Genes", "Demographics", "Blood"))
sangerGroups[names(sangerX) %in% c("age","gender")] <- "Demographics"
table(sangerGroups)
```

```
## sangerGroups
##        Genes Demographics        Blood
##           15            2           13
```

```
colnames(sangerX)
```

```
##  [1] "ASXL1"    "CBL"      "DNMT3A"   "JAK2"     "KMT2C"    "KMT2D"    "KRAS"
"NF1"      "NRAS"     "RAD21"
## [11] "SF3B1"    "SRSF2"    "TET2"     "TP53"     "U2AF1"    "age"      "gender"
"systol"   "diastol"  "bmi"
## [21] "cholestl" "triglyc"  "hdl"      "ldl"      "lym"      "mcv"      "rdw"
"wbc"      "plt"      "hgb"
```

```
sangerGroups
```

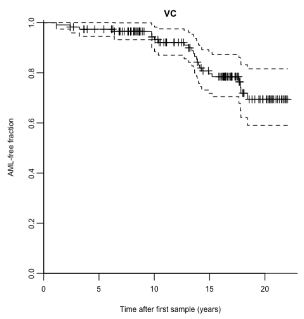
```
##  [1] Genes        Genes        Genes        Genes        Genes        Genes
Genes        Genes
##  [9] Genes        Genes        Genes        Genes        Genes        Genes
Genes        Demographics
## [17] Demographics Blood        Blood        Blood        Blood        Blood
Blood        Blood
## [25] Blood        Blood        Blood        Blood        Blood        Blood
## Levels: Genes Demographics Blood
```

```
g <- sangerGroups=="Genes"
sangerX[g] <- sangerX[g] * 10
names(sangerX)[g] <- paste(names(sangerX[g]),"0.1", sep="_")
y <- StandardizeMagnitude(sangerX[!g])
sangerX <- cbind(sangerX[g],y)

poorMansImpute <- function(x) {x[is.na(x)] <- mean(x, na.rm=TRUE); return(x)}
sangerX <- as.data.frame(sapply(sangerX, poorMansImpute))

foo <- split(sangerData[,c("Diagnosis","hcdate","dodx")], sangerPatients)

bar <- do.call("rbind",lapply(foo, function(x){
  y <- x
  n <- nrow(y)
  y[-n,"Diagnosis"] <- "Control"
  start <- as.numeric(y$hcdate - y$hcdate[1])/365.25
  end <- c(as.numeric(y$hcdate - y$hcdate[1])[-1]/365.25, as.numeric(y$dodx[n] - y
$hcdate[1])/365.25)
  return(data.frame(Diagnosis=y[,"Diagnosis"], start=start, end=end))
}))

bar[1:10, ]
```

	Diagnosis <fctr>	start <dbl>	end <dbl>
PD29762	AML	0.000000	9.754962
PD29764	AML	0.000000	10.360027
PD29792	AML	0.000000	14.108145
PD29810	Control	0.000000	18.573580
PD29836.1	Control	0.000000	2.414784
PD29836.2	AML	2.414784	10.023272
PD29856	AML	0.000000	17.828884
PD29896	AML	0.000000	6.387406
PD29918.1	Control	0.000000	5.442847
PD29918.2	AML	5.442847	13.396304

1-10 of 10 rows

```
sangerPatientsSplit <- unlist(sapply(names(foo), function(n) rep(n, nrow(foo[[n]])
)))

sangerSurv <- Surv(time = bar$start, time2 = bar$end, event = bar$Diagnosis!="Cont
rol", origin = 0)

plot(survfit(sangerSurv~ 1), col= "black", main = "VC", xlab='Time after first sam
ple (years)', ylab='AML-free fraction', bty='L', yaxs='i', ylim=c(0,1.01), mark.ti
me = T) #mark = 1
```



12.3 Expected AML incidence

Validation cohort

```
w <- c(which(sangerSurv[,1]==0)[-1]-1, nrow(sangerSurv))
sangerSurv2 <- Surv(sangerSurv[w,2], sangerSurv[w,3])

expected_rate_sanger_cr <- mean(aml_inc_cr(sangerX[w,"gender"],sangerX[w,"age_10"]
*10, sangerX[w,"age_10"]*10+ pmax(1,sangerSurv2[,1]))[!sangerSurv2[,2]])

n_total_sanger <- sum(sangerSurv2[,2])/expected_rate_sanger_cr
n_total_sanger
```

```
## [1] 9216.197
```

Discovery cohort

```
expected_rate_toronto_cr <- mean(aml_inc_cr(torontoX[,"gender"],torontoX[,"age_10"
]*10, torontoX[,"age_10"]*10+ pmax(1,torontoSurv[,1]))[!torontoSurv[,2]])

n_total_toronto <- sum(torontoSurv[,2])/expected_rate_toronto_cr
n_total_toronto
```

```
## [1] 42940.66
```

12.4 Combined data

Survival

```
allSurv <- rbind(sangerSurv, Surv(rep(0, nrow(torontoSurv)), torontoSurv[,1], toro
ntoSurv[,2]))
allSurv <- Surv(allSurv[,1], allSurv[,2], allSurv[,3])
```

Data matrix

```
cohort <- c(rep("Sanger", nrow(sangerX)), rep("Toronto", nrow(torontoX)))
i <- c(sort(setdiff(gene_vars,"CALR")),"age","gender")
allX <- rbind(superSet(sangerData,i,fill=0), superSet(torontoData,i,fill=0))
allX <- allX[,colSums(allX>0)>=thr]
allX <- cbind(allX, cohort=cohort=="Sanger") + 0
allGroups <- factor(grepl("^[A-Z]",colnames(allX))+0, levels=1:0, labels=c("Genes"
,"Demographics"))

g <- allGroups=="Genes"
allX <- cbind(10*allX[,g], StandardizeMagnitude(allX[,!g]))
colnames(allX)[g] <- paste(colnames(allX)[g],"0.1",sep="_")
control <- c(sangerData$Diagnosis=="Control", torontoData$Diagnosis=="Control")
```

Weights

```
weights <- rep(1, nrow(allX))
weights[cohort=="Sanger" & control] <- n_total_sanger/sum(cohort=="Sanger" & contr
ol & allSurv[,1]==0)
weights[cohort=="Toronto" & control] <- n_total_toronto/sum(cohort=="Toronto" & co
ntrol)

n_total <- n_total_sanger + n_total_toronto
n_total
```

```
## [1] 52156.85
```

12.5 Coxph model fits

```
sigma0 <- 0.1
nu <- 1
which.mu <- "Genes"
```

12.5.1 Toronto

12.5.1.1 Raw

```
fitToronto <- CoxRFX(torontoX, torontoSurv, groups=torontoGroups, which.mu=which.m
u, nu=nu, sigma0=sigma0)
waldToronto <- WaldTest(fitToronto)
```

```
##                    group    coef   coef-mu     sd      z df  p.value sig
## ASXL1_0.1          Genes  0.5750  0.032700 0.1158  4.964  1 6.91e-07 ***
## CALR_0.1           Genes  0.5200 -0.022339 0.0744  6.990  1 2.74e-12 ***
## CBL_0.1            Genes  0.4268 -0.115522 0.1231  3.469  1 5.23e-04 ***
## DNMT3A_0.1         Genes  0.4724 -0.069936 0.1062  4.448  1 8.66e-06 ***
## IDH1_0.1           Genes  0.5730  0.030722 0.1188  4.822  1 1.42e-06 ***
## IDH2_0.1           Genes  0.4711 -0.071177 0.1126  4.184  1 2.86e-05 ***
## JAK2_0.1           Genes  0.6084  0.066072 0.1214  5.011  1 5.43e-07 ***
## KDM6A_0.1          Genes  0.5420 -0.000284 0.0628  8.629  1 6.17e-18 ***
## KMT2C_0.1          Genes  0.5603  0.017953 0.0656  8.545  1 1.29e-17 ***
## KRAS_0.1           Genes  0.5394 -0.002952 0.0628  8.583  1 9.20e-18 ***
## NF1_0.1            Genes  0.5404 -0.001954 0.0628  8.599  1 8.07e-18 ***
## PHF6_0.1           Genes  0.5469  0.004542 0.0632  8.655  1 4.91e-18 ***
## PTPN11_0.1         Genes  0.5556  0.013243 0.0631  8.810  1 1.25e-18 ***
## RUNX1_0.1          Genes  0.3347 -0.207621 0.0917  3.650  1 2.62e-04 ***
## SF3B1_0.1          Genes  0.6532  0.110858 0.0963  6.781  1 1.19e-11 ***
## SRSF2_0.1          Genes  0.4370 -0.105330 0.0920  4.750  1 2.03e-06 ***
## TET2_0.1           Genes  0.5053 -0.037059 0.1248  4.050  1 5.12e-05 ***
## TP53_0.1           Genes  0.7280  0.185639 0.0825  8.828  1 1.07e-18 ***
## U2AF1_0.1          Genes  0.7148  0.172443 0.0805  8.879  1 6.76e-19 ***
## age_10      Demographics -0.0236 -0.023625 0.1092 -0.216  1 8.29e-01
## gender      Demographics -0.0832 -0.083228 0.1113 -0.748  1 4.55e-01
```

```
survConcordance(fitToronto$surv ~ fitToronto$linear.predictors)
```

```
## Call:
## survConcordance(formula = fitToronto$surv ~ fitToronto$linear.predictors)
##
##   n= 221
## Concordance= 0.7806171 se= 0.03687602
## concordant discordant  tied.risk  tied.time   std(c-d)
##  8981.0000  2524.0000     0.0000     1.0000   848.5173
```

12.5.1.2 Adjusted

```
fitWeightedToronto <- CoxRFX(torontoX, torontoSurv, torontoGroups, which.mu=which.
mu, sigma0=sigma0, nu=nu, weights=weights[cohort=="Toronto"])
waldWeightedToronto <- WaldTest(fitWeightedToronto)
```

```
##                   group    coef  coef-mu    sd      z df  p.value sig
## ASXL1_0.1         Genes  1.9878  0.06756 0.150 13.267  1 3.60e-40 ***
## CALR_0.1          Genes  0.6189 -1.30126 0.758  0.817  1 4.14e-01    
## CBL_0.1           Genes  0.2531 -1.66705 0.379  0.668  1 5.04e-01    
## DNMT3A_0.1        Genes  0.5859 -1.33434 0.136  4.313  1 1.61e-05 ***
## IDH1_0.1          Genes  2.4124  0.49218 0.341  7.083  1 1.41e-12 ***
## IDH2_0.1          Genes  0.8067 -1.11352 0.231  3.498  1 4.70e-04 ***
## JAK2_0.1          Genes  1.9535  0.03333 0.193 10.131  1 4.01e-24 ***
## KDM6A_0.1         Genes  1.9181 -0.00209 0.163 11.792  1 4.31e-32 ***
## KMT2C_0.1         Genes  2.3735  0.45328 0.730  3.250  1 1.16e-03  **
## KRAS_0.1          Genes  1.7434 -0.17684 0.195  8.955  1 3.38e-19 ***
## NF1_0.1           Genes  1.8059 -0.11434 0.190  9.518  1 1.77e-21 ***
## PHF6_0.1          Genes  2.2276  0.30741 0.144 15.462  1 6.24e-54 ***
## PTPN11_0.1        Genes  2.5970  0.67679 0.277  9.366  1 7.52e-21 ***
## RUNX1_0.1         Genes  0.7172 -1.20303 0.137  5.235  1 1.65e-07 ***
## SF3B1_0.1         Genes  3.2528  1.33260 0.321 10.149  1 3.36e-24 ***
## SRSF2_0.1         Genes  1.4698 -0.45035 0.170  8.656  1 4.91e-18 ***
## TET2_0.1          Genes  0.5707 -1.34952 0.211  2.699  1 6.96e-03  **
## TP53_0.1          Genes  5.2413  3.32111 0.440 11.916  1 9.82e-33 ***
## U2AF1_0.1         Genes  3.9483  2.02809 0.365 10.817  1 2.87e-27 ***
## age_10     Demographics -0.0820 -0.08201 0.117 -0.700  1 4.84e-01    
## gender     Demographics -0.0899 -0.08989 0.117 -0.771  1 4.41e-01    
```

```
survConcordance(fitWeightedToronto$surv ~ fitWeightedToronto$linear.predictors, we
ights=weights[cohort=="Toronto"])
```

```
## Call:
## survConcordance(formula = fitWeightedToronto$surv ~ fitWeightedToronto$linear.p
redictors, 
##     weights = weights[cohort == "Toronto"])
## 
##   n= 221 
## Concordance= 0.8454794 se= 0.03633541
## concordant discordant  tied.risk  tied.time   std(c-d) 
##  2196217.1   401382.8        0.0        1.0   188769.7 
```

Uno’s estimator of cumulative/dynamic AUC

```
a <- AUC.uno(torontoSurv, torontoSurv, fitWeightedToronto$linear.predictors, times
= seq(0,12, 0.1))
round(a$iauc, digits = 3)
```

```
## [1] 0.791
```

12.5.2 Validation cohort

12.5.2.1 Raw

```
fitSanger <- CoxRFX(sangerX, sangerSurv, groups=sangerGroups, which.mu=which.mu, n
u=nu, sigma0=sigma0)
waldSanger <- WaldTest(fitSanger)
```

```
##                    group      coef   coef-mu      sd        z df  p.value sig
## ASXL1_0.1          Genes  0.673478  0.158950 0.12882  5.22794  1 1.71e-07 ***
## CBL_0.1            Genes  0.495353 -0.019175 0.10735  4.61426  1 3.94e-06 ***
## DNMT3A_0.1         Genes  0.328415 -0.186113 0.13178  2.49210  1 1.27e-02   *
## JAK2_0.1           Genes  0.493355 -0.021173 0.11739  4.20278  1 2.64e-05 ***
## KMT2C_0.1          Genes  0.519077  0.004549 0.10042  5.16888  1 2.36e-07 ***
## KMT2D_0.1          Genes  0.341708 -0.172820 0.16670  2.04989  1 4.04e-02   *
## KRAS_0.1           Genes  0.517799  0.003272 0.09650  5.36592  1 8.05e-08 ***
## NF1_0.1            Genes  0.501902 -0.012625 0.09919  5.06022  1 4.19e-07 ***
## NRAS_0.1           Genes  0.534425  0.019897 0.09703  5.50790  1 3.63e-08 ***
## RAD21_0.1          Genes  0.503868 -0.010660 0.09793  5.14544  1 2.67e-07 ***
## SF3B1_0.1          Genes  0.507855 -0.006673 0.09801  5.18184  1 2.20e-07 ***
## SRSF2_0.1          Genes  0.529928  0.015400 0.14168  3.74021  1 1.84e-04 ***
## TET2_0.1           Genes  0.593720  0.079192 0.12273  4.83743  1 1.32e-06 ***
## TP53_0.1           Genes  0.584538  0.070010 0.09773  5.98121  1 2.21e-09 ***
## U2AF1_0.1          Genes  0.592496  0.077968 0.09770  6.06442  1 1.32e-09 ***
## age_10      Demographics  0.084731  0.084731 0.12166  0.69645  1 4.86e-01    
## gender      Demographics -0.007960 -0.007960 0.10340 -0.07698  1 9.39e-01    
## systol_100         Blood  0.033564  0.033564 0.03644  0.92111  1 3.57e-01    
## diastol_100        Blood  0.032432  0.032432 0.02299  1.41095  1 1.58e-01    
## bmi_10             Blood  0.081752  0.081752 0.06892  1.18610  1 2.36e-01    
## cholestl_10        Blood  0.014082  0.014082 0.01344  1.04742  1 2.95e-01    
## triglyc            Blood -0.000827 -0.000827 0.10813 -0.00765  1 9.94e-01    
## hdl                Blood -0.007587 -0.007587 0.06927 -0.10952  1 9.13e-01    
## ldl                Blood  0.134372  0.134372 0.11043  1.21684  1 2.24e-01    
## lym                Blood  0.076500  0.076500 0.08867  0.86278  1 3.88e-01    
## mcv_100            Blood -0.012801 -0.012801 0.00713 -1.79436  1 7.28e-02   .
## rdw_10             Blood  0.058557  0.058557 0.01828  3.20254  1 1.36e-03  **
## wbc_10             Blood  0.016691  0.016691 0.03908  0.42707  1 6.69e-01    
## plt_100            Blood  0.095820  0.095820 0.09229  1.03821  1 2.99e-01    
## hgb_10             Blood  0.006904  0.006904 0.01981  0.34856  1 7.27e-01    
```

RDW p-val 9.171138e-03

```
survConcordance(sangerSurv ~ fitSanger$linear.predictors)
```

```
## Call:
## survConcordance(formula = sangerSurv ~ fitSanger$linear.predictors)
## 
##   n= 149 
## Concordance= 0.7918502 se= 0.06247796
## concordant discordant  tied.risk  tied.time   std(c-d) 
##    1438.00     378.00       0.00       0.00     226.92 
```

12.5.2.2 Adjusted

```
fitWeightedSanger <- CoxRFX(sangerX, sangerSurv, sangerGroups, which.mu=which.mu,
sigma0=sigma0, nu=nu, weights=weights[cohort=="Sanger"])
waldWeightedSanger <- WaldTest(fitWeightedSanger)
```

```
##                     group    coef coef-mu     sd       z df  p.value sig
## ASXL1_0.1           Genes  3.2736  1.1639 0.5035  6.5016  1 7.95e-11 ***
## CBL_0.1             Genes  0.4415 -1.6682 1.4885  0.2966  1 7.67e-01
## DNMT3A_0.1          Genes  0.5963 -1.5134 0.2434  2.4497  1 1.43e-02   *
## JAK2_0.1            Genes -0.0225 -2.1322 1.0506 -0.0214  1 9.83e-01
## KMT2C_0.1           Genes  0.8233 -1.2864 1.4975  0.5498  1 5.82e-01
## KMT2D_0.1           Genes -0.1936 -2.3033 0.9186 -0.2108  1 8.33e-01
## KRAS_0.1            Genes  2.6546  0.5449 0.6402  4.1468  1 3.37e-05 ***
## NF1_0.1             Genes  0.8839 -1.2258 1.4275  0.6192  1 5.36e-01
## NRAS_0.1            Genes  4.8796  2.7699 0.6294  7.7532  1 8.96e-15 ***
## RAD21_0.1           Genes  0.8665 -1.2432 1.4103  0.6144  1 5.39e-01
## SF3B1_0.1           Genes  1.2701 -0.8396 1.4768  0.8601  1 3.90e-01
## SRSF2_0.1           Genes  1.6909 -0.4188 0.2626  6.4399  1 1.20e-10 ***
## TET2_0.1            Genes  1.3640 -0.7457 0.1595  8.5534  1 1.19e-17 ***
## TP53_0.1            Genes  5.1102  3.0005 1.0728  4.7634  1 1.90e-06 ***
## U2AF1_0.1           Genes  8.0069  5.8972 0.9739  8.2214  1 2.01e-16 ***
## age_10       Demographics -0.0522 -0.0522 0.1212 -0.4306  1 6.67e-01
## gender       Demographics -0.0216 -0.0216 0.0988 -0.2185  1 8.27e-01
## systol_100          Blood  0.0064  0.0064 0.0409  0.1566  1 8.76e-01
## diastol_100         Blood  0.0251  0.0251 0.0269  0.9320  1 3.51e-01
## bmi_10              Blood  0.0956  0.0956 0.0826  1.1574  1 2.47e-01
## cholestl_10         Blood  0.0143  0.0143 0.0155  0.9246  1 3.55e-01
## triglyc             Blood -0.0533 -0.0533 0.1279 -0.4169  1 6.77e-01
## hdl                 Blood -0.0505 -0.0505 0.0839 -0.6015  1 5.48e-01
## ldl                 Blood  0.2011  0.2011 0.1239  1.6229  1 1.05e-01
## lym                 Blood  0.0499  0.0499 0.0996  0.5009  1 6.16e-01
## mcv_100             Blood -0.0238 -0.0238 0.0075 -3.1777  1 1.48e-03  **
## rdw_10              Blood  0.0832  0.0832 0.0142  5.8698  1 4.36e-09 ***
## wbc_10              Blood  0.0108  0.0108 0.0544  0.1988  1 8.42e-01
## plt_100             Blood  0.1509  0.1509 0.1056  1.4297  1 1.53e-01
## hgb_10              Blood -0.0224 -0.0224 0.0217 -1.0308  1 3.03e-01
```

RDW p-val 1.233241e-07

```
survConcordance(sangerSurv ~ fitWeightedSanger$linear.predictors, weights=weights[
cohort=="Sanger"])
```

```
## Call:
## survConcordance(formula = sangerSurv ~ fitWeightedSanger$linear.predictors,
##     weights = weights[cohort == "Sanger"])
##
##   n= 149
## Concordance= 0.8671072 se= 0.06105924
## concordant discordant  tied.risk  tied.time   std(c-d)
##  135478.93   20763.49       0.00       0.00   19080.09
```

Uno's estimator of cumulative/dynamic AUC

```
w <- c(which(sangerSurv[,1]==0)[-1]-1, nrow(sangerSurv))
s <- Surv(sangerSurv[w,2], sangerSurv[w,3])
a <- AUC.uno(s, s, fitWeightedSanger$linear.predictors[w], times= c(0, 22, 0.1))
round(a$iauc, digits = 3)
```

```
## [1] 0.587
```

13 Session

```
devtools::session_info()
```

```
## Session info -------------------------------------------------------------------
---------------------------------------
```

```
##  setting  value
##  version  R version 3.4.2 (2017-09-28)
##  system   x86_64, darwin15.6.0
##  ui       X11
##  language (EN)
##  collate  en_US.UTF-8
##  tz       Europe/London
##  date     2018-02-16
```

```
## Packages -----------------------------------------------------------------------
---------------------------------------
```

```
##  package      * version date       source
##  backports      1.1.1   2017-09-25 CRAN (R 3.4.2)
##  base         * 3.4.2   2017-10-04 local
##  bitops         1.0-6   2013-08-17 CRAN (R 3.4.0)
##  car            2.1-6   2017-11-19 CRAN (R 3.4.3)
##  caTools        1.17.1  2014-09-10 CRAN (R 3.4.0)
##  codetools      0.2-15  2016-10-05 CRAN (R 3.4.2)
##  compiler       3.4.2   2017-10-04 local
##  CoxHD        * 0.0.73  2018-01-08 Github (gerstung-lab/CoxHD@bc60c16)
##  datasets     * 3.4.2   2017-10-04 local
##  devtools       1.13.4  2017-11-09 CRAN (R 3.4.2)
##  digest         0.6.12  2017-01-27 CRAN (R 3.4.0)
##  evaluate       0.10.1  2017-06-24 CRAN (R 3.4.1)
##  foreach      * 1.4.3   2015-10-13 cran (@1.4.3)
##  gdata          2.18.0  2017-06-06 CRAN (R 3.4.0)
##  glmnet       * 2.0-13  2017-09-22 cran (@2.0-13)
##  gplots       * 3.0.1   2016-03-30 CRAN (R 3.4.0)
##  graphics     * 3.4.2   2017-10-04 local
##  grDevices    * 3.4.2   2017-10-04 local
##  grid           3.4.2   2017-10-04 local
##  gtools         3.5.0   2015-05-29 CRAN (R 3.4.0)
##  htmltools      0.3.6   2017-04-28 CRAN (R 3.4.0)
##  iterators      1.0.8   2015-10-13 cran (@1.0.8)
##  jsonlite       1.5     2017-06-01 CRAN (R 3.4.0)
##  KernSmooth     2.23-15 2015-06-29 CRAN (R 3.4.2)
##  knitr        * 1.17    2017-08-10 CRAN (R 3.4.1)
##  lattice        0.20-35 2017-03-25 CRAN (R 3.4.2)
##  lme4           1.1-15  2017-12-21 CRAN (R 3.4.3)
##  magrittr       1.5     2014-11-22 CRAN (R 3.4.0)
##  MASS           7.3-48  2017-12-25 CRAN (R 3.4.3)
##  Matrix       * 1.2-11  2017-08-21 CRAN (R 3.4.2)
##  MatrixModels   0.4-1   2015-08-22 CRAN (R 3.4.0)
##  memoise        1.1.0   2017-04-21 CRAN (R 3.4.0)
##  methods      * 3.4.2   2017-10-04 local
##  mg14           0.0.5   2017-11-18 Github (mg14/mg14@a8b4ba8)
##  mgcv           1.8-20  2017-09-14 CRAN (R 3.4.2)
##  mice           2.46.0  2017-10-24 cran (@2.46.0)
##  minqa          1.2.4   2014-10-09 CRAN (R 3.4.0)
##  mvtnorm        1.0-6   2017-03-02 CRAN (R 3.4.0)
##  nlme           3.1-131 2017-02-06 CRAN (R 3.4.2)
##  nloptr         1.0.4   2014-08-04 CRAN (R 3.4.0)
##  nnet           7.3-12  2016-02-02 CRAN (R 3.4.2)
##  parallel     * 3.4.2   2017-10-04 local
##  pbkrtest       0.4-7   2017-03-15 CRAN (R 3.4.0)
##  quantreg       5.34    2017-10-25 CRAN (R 3.4.2)
##  RColorBrewer * 1.1-2   2014-12-07 CRAN (R 3.4.0)
##  Rcpp           0.12.15 2018-01-20 cran (@0.12.15)
##  rmarkdown      1.8     2017-11-17 CRAN (R 3.4.2)
##  ROCR         * 1.0-7   2015-03-26 CRAN (R 3.4.0)
##  rpart          4.1-11  2017-03-13 CRAN (R 3.4.2)
##  rprojroot      1.3-1   2017-12-18 CRAN (R 3.4.2)
##  SparseM        1.77    2017-04-23 CRAN (R 3.4.0)
##  splines        3.4.2   2017-10-04 local
##  stats        * 3.4.2   2017-10-04 local
##  stringi        1.1.6   2017-11-17 CRAN (R 3.4.2)
##  stringr        1.2.0   2017-02-18 CRAN (R 3.4.0)
##  survAUC      * 1.0-5   2012-09-04 CRAN (R 3.4.0)
##  survival     * 2.41-3  2017-04-04 CRAN (R 3.4.0)
##  survivalROC  * 1.0.3   2013-01-13 CRAN (R 3.4.0)
##  tools          3.4.2   2017-10-04 local
##  utils        * 3.4.2   2017-10-04 local
##  withr          2.1.0   2017-11-01 CRAN (R 3.4.2)
##  yaml           2.1.14  2016-11-12 CRAN (R 3.4.0)
```

This code and all data necessary to execute it is availalbe from http://www.github.com/gerstung-lab/ (http://www.github.com/gerstung-lab/)