homework1

Question #2 Researchers wish to explore the efficacy of triple-drug combinations of antiretroviral therapy for treatment of HIV-infected patients. Because of limitations on potency and the continuing emergence of drug resistance seen with the use of currently available antiretroviral agents in monotherapy and two-drug regimens, triple combination regimens should represent a more promising approach to maximize antiviral activity, maintain long-term efficacy, and reduce the incidence of drug resistance. Towards this end, investigators performed a randomized study comparing AZT + zalcitabine (ddC) versus AZT + zalcitabine (ddC) + saquinavir. The data, time from administration of treatment (in days) until the CD4 count reached a pre-specified level, is given below for the two groups: AZT + zalcitabine (ddC): 4+, 6, 11, 12, 32, 35, 38+, 39, 45, 49, 75, 80, 84, 85, 87, 102, 180+ AZT + zalcitabine (ddC) + saquinivir: 2, 3, 4, 12, 22, 48, 51+, 56+, 80, 85, 90, 94+, 160, 171, 180, 180+, 238

azt\_ddc=c("4+",6,11,12,32,35,"38+",39,45,49,75,80,84,85,87,102,"180+")  
azt\_ddc\_saq=c(2,3,4,12,22,48,51,56,80,85,90,94,160,171,180,180,238)  
  
# kmTable=setNames(data.frame(matrix(NA,nrow=1,ncol=length(kmTableColumnNames))),kmTableColumnNames)  
kmTable=data.frame()  
getKMTable = function(censoredTimesVector,censorSymbol){  
 #get numeric representation of censor vector  
 censoredTimesVectorNumeric=as.numeric(sub(censorSymbol,'',censoredTimesVector,fixed=TRUE))  
 #count number of actual rows in KM table  
 cnt\_n=length(censoredTimesVectorNumeric)  
 #create first row of KM table  
 kmTable=setNames(data.frame(matrix(nrow=1,c(0,0,0,cnt\_n,as.character("-"),as.character(paste0(cnt\_n,"/",cnt\_n)),1)),stringsAsFactors=FALSE),c("orderedEventTimes\_tj","eventsAtEventTime\_ej",  
 "censoredObservationsInInterval\_cj","inRiskSetAtTime\_nj","kaplanMeirSurvivalCurveAtTime\_s\_tj-1","c\_tj-1","kaplanMeirSurvivalCurveAtTime\_s\_tj"))  
 # orderedIndices=order(censoredTimesVectorNumeric)  
 # censoredTimesVectorNumeric=censoredTimesVectorNumeric[orderedIndices]  
 # censoredTimesVector=censoredTimesVector[orderedIndices]  
 censoredTimesVectorNumeric=sort(censoredTimesVectorNumeric)  
 for (i in 1:max(censoredTimesVectorNumeric)){  
 if(i %in% censoredTimesVectorNumeric){  
 #create empty row to fill in  
 kmTableRow=setNames(data.frame(matrix(NA,nrow=1,ncol=length(names(kmTable)))),names(kmTable))  
 kmTableRow$orderedEventTimes\_tj=i  
 #count how many events at time  
 kmTableRow$eventsAtEventTime\_ej=length(which(censoredTimesVector==i))  
 #count how many censured at time  
 kmTableRow$censoredObservationsInInterval\_cj=length(which(censoredTimesVector==paste0(i,censorSymbol)))  
 kmTableRow$inRiskSetAtTime\_nj=cnt\_n  
 #sum events and number censored at time  
 loss=kmTableRow$eventsAtEventTime\_ej+kmTableRow$censoredObservationsInInterval\_cj  
 kmTableRow[c("kaplanMeirSurvivalCurveAtTime\_s\_tj-1")]=kmTable[dim(kmTable)[1],c("kaplanMeirSurvivalCurveAtTime\_s\_tj")]  
 kmTableRow[c("c\_tj-1")]=paste0((cnt\_n-loss),"/",cnt\_n)  
 kmTableRow$kaplanMeirSurvivalCurveAtTime\_s\_tj=round((cnt\_n-loss)/length(censoredTimesVectorNumeric),3)  
 #update count  
 cnt\_n=cnt\_n-loss  
 #add row to kmtable  
 kmTable=rbind(kmTable,kmTableRow)  
 }  
 }  
 kmTable  
}  
azt\_ddc\_KM=getKMTable(azt\_ddc,"+")  
azt\_ddc\_saq\_KM=getKMTable(azt\_ddc\_saq,"+")  
# write.table(azt\_ddc\_saq\_KM,file="C:/Users/an052283/OneDrive - Cerner Corporation/MSASA/Stat-845/Homeworks/output/Homework1/azt\_ddc\_saq\_KM.tsv",sep="\t",row.names = FALSE)  
# write.table(azt\_ddc\_KM,file="C:/Users/an052283/OneDrive - Cerner Corporation/MSASA/Stat-845/Homeworks/output/Homework1/azt\_ddc\_KM.tsv",sep="\t",row.names = FALSE)  
View(azt\_ddc\_KM)  
View(azt\_ddc\_saq\_KM)  
  
library(survival)

## Warning: package 'survival' was built under R version 3.5.2

#numeric times and censor list (0 for not censored 1 for censored)  
Surv(as.numeric(sub("+","",azt\_ddc,fixed=TRUE)),ifelse(grepl("+",azt\_ddc,fixed=TRUE),0,1))

## [1] 4+ 6 11 12 32 35 38+ 39 45 49 75 80 84 85   
## [15] 87 102 180+

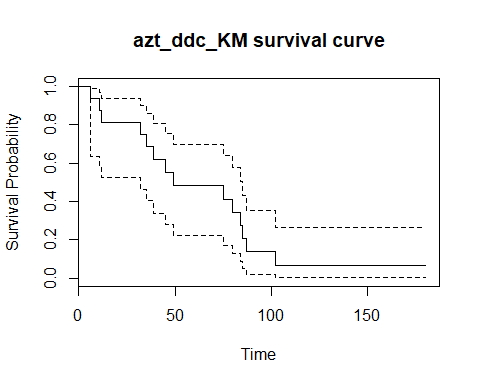
azt\_ddc\_KM\_R=survfit(Surv(as.numeric(sub("+","",azt\_ddc,fixed=TRUE)),ifelse(grepl("+",azt\_ddc,fixed=TRUE),0,1))~1,conf.type="log-log")  
summary(azt\_ddc\_KM\_R)

## Call: survfit(formula = Surv(as.numeric(sub("+", "", azt\_ddc, fixed = TRUE)),   
## ifelse(grepl("+", azt\_ddc, fixed = TRUE), 0, 1)) ~ 1, conf.type = "log-log")  
##   
## time n.risk n.event survival std.err lower 95% CI upper 95% CI  
## 6 16 1 0.9375 0.0605 0.63235 0.991  
## 11 15 1 0.8750 0.0827 0.58598 0.967  
## 12 14 1 0.8125 0.0976 0.52460 0.935  
## 32 13 1 0.7500 0.1083 0.46343 0.898  
## 35 12 1 0.6875 0.1159 0.40460 0.856  
## 39 10 1 0.6188 0.1230 0.33929 0.808  
## 45 9 1 0.5500 0.1271 0.27933 0.756  
## 49 8 1 0.4813 0.1285 0.22410 0.699  
## 75 7 1 0.4125 0.1272 0.17339 0.639  
## 80 6 1 0.3438 0.1232 0.12728 0.575  
## 84 5 1 0.2750 0.1162 0.08617 0.507  
## 85 4 1 0.2063 0.1055 0.05082 0.433  
## 87 3 1 0.1375 0.0900 0.02265 0.354  
## 102 2 1 0.0688 0.0662 0.00443 0.267

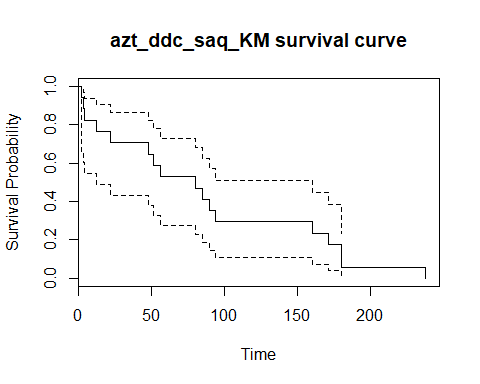
azt\_ddc\_saq\_KM\_R = survfit(Surv(as.numeric(sub("+","",azt\_ddc\_saq,fixed=TRUE)),ifelse(grepl("+",azt\_ddc\_saq,fixed=TRUE),0,1))~1,conf.type="log-log")  
summary(azt\_ddc\_saq\_KM\_R)

## Call: survfit(formula = Surv(as.numeric(sub("+", "", azt\_ddc\_saq, fixed = TRUE)),   
## ifelse(grepl("+", azt\_ddc\_saq, fixed = TRUE), 0, 1)) ~ 1,   
## conf.type = "log-log")  
##   
## time n.risk n.event survival std.err lower 95% CI upper 95% CI  
## 2 17 1 0.9412 0.0571 0.65018 0.991  
## 3 16 1 0.8824 0.0781 0.60598 0.969  
## 4 15 1 0.8235 0.0925 0.54713 0.939  
## 12 14 1 0.7647 0.1029 0.48828 0.904  
## 22 13 1 0.7059 0.1105 0.43148 0.866  
## 48 12 1 0.6471 0.1159 0.37715 0.823  
## 51 11 1 0.5882 0.1194 0.32537 0.778  
## 56 10 1 0.5294 0.1211 0.27617 0.730  
## 80 9 1 0.4706 0.1211 0.22960 0.680  
## 85 8 1 0.4118 0.1194 0.18576 0.626  
## 90 7 1 0.3529 0.1159 0.14483 0.570  
## 94 6 1 0.2941 0.1105 0.10712 0.511  
## 160 5 1 0.2353 0.1029 0.07308 0.449  
## 171 4 1 0.1765 0.0925 0.04348 0.383  
## 180 3 2 0.0588 0.0571 0.00391 0.235  
## 238 1 1 0.0000 NaN NA NA

plot(azt\_ddc\_KM\_R,xlab="Time",ylab="Survival Probability",main="azt\_ddc\_KM survival curve")



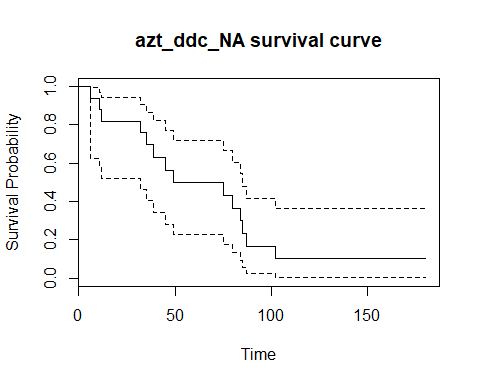
plot(azt\_ddc\_saq\_KM\_R,xlab="Time",ylab="Survival Probability",main="azt\_ddc\_saq\_KM survival curve")

 For both groups separately, construct a data layout (similar to what was done in lecture slides) containing the unique, ordered event times, the number of events that occurred at those unique event times, the number of censored observations in the relevant interval, the number in the risk set at that time, and the Kaplan-Meier estimate of the survival curve at that time. What is the median survival time in the two groups? Will you be comfortable reporting the mean survival time in the two groups?

#6MP as test case  
leukemia\_6MP = c(10,7,"32+",23,22,6,16,"34+","32+","25+","11+","20+","19+",6,"17+","35+",6,13,"9+","6+","10+")  
getNATable = function(censoredTimesVector,censorSymbol){  
 #get numeric representation of censor vector  
 censoredTimesVectorNumeric=as.numeric(sub(censorSymbol,'',censoredTimesVector,fixed=TRUE))  
 #count number of actual rows in NA table  
 cnt\_n=length(censoredTimesVectorNumeric)  
 #create first row of NA table  
 naTable=setNames(data.frame(matrix(nrow=1,c(0,0,cnt\_n,0,0,as.character(paste0(cnt\_n,"/",cnt\_n)),0,0)),stringsAsFactors=FALSE),c("orderedEventTimes\_tj","eventsAtEventTime\_ej","inRiskSetAtTime\_nj","censoredObservationsInInterval\_cj","cumulativeHazardRate\_ht","d\_Y\_ratio","cumulativeHazardEstimatedVariance\_vt","nelsonAalenSurvivalCurveAtTime\_s\_tj"))  
 censoredTimesVectorNumeric=sort(censoredTimesVectorNumeric)  
 sumCensoredInInterval=0  
 for (i in 1:max(censoredTimesVectorNumeric)){  
 if(i %in% censoredTimesVectorNumeric){  
 #create empty row to fill in  
 naTableRow=setNames(data.frame(matrix(NA,nrow=1,ncol=length(names(naTable)))),names(naTable))  
 naTableRow$orderedEventTimes\_tj=i  
 #count how many events at time  
 naTableRow$eventsAtEventTime\_ej=length(which(censoredTimesVector==i))  
 #running total of censured between censured time intervals  
 naTableRow$censoredObservationsInInterval\_cj=length(which(censoredTimesVector==paste0(i,censorSymbol)))  
 sumCensoredInInterval=sumCensoredInInterval+naTableRow$censoredObservationsInInterval\_cj  
 naTableRow$inRiskSetAtTime\_nj=cnt\_n  
 naTableRow$d\_Y\_ratio=paste0((naTableRow$eventsAtEventTime\_ej),"/",cnt\_n)  
 naTableRow$cumulativeHazardRate\_ht=round(as.numeric(naTable[dim(naTable)[1],c("cumulativeHazardRate\_ht")])+naTableRow$eventsAtEventTime\_ej/cnt\_n,3)  
 naTableRow$cumulativeHazardEstimatedVariance\_vt=round(as.numeric(naTable[dim(naTable)[1],c("cumulativeHazardEstimatedVariance\_vt")])+naTableRow$eventsAtEventTime\_ej/(cnt\_n)^2,3)  
 naTableRow$nelsonAalenSurvivalCurveAtTime\_s\_tj=round(exp(-naTableRow$cumulativeHazardRate\_ht),3)  
 #sum events and number censored at time  
 loss=naTableRow$eventsAtEventTime\_ej+naTableRow$censoredObservationsInInterval\_cj  
 #update count  
 cnt\_n=cnt\_n-loss  
 #add row to na table if at least one uncensored variable  
 if (i %in% censoredTimesVector){  
 naTableRow$censoredObservationsInInterval\_cj=sumCensoredInInterval  
 naTable=rbind(naTable,naTableRow)  
 sumCensoredInInterval=0  
 }  
 }  
 }  
 naTable  
}  
  
#adapted from http://sas-and-r.blogspot.com/2010/05/example-739-nelson-aalen-estimate-of.html  
getCumulativeHazardNA = function(time, event) {  
 na.fit = survfit(coxph(Surv(time,event)~1), type="aalen")  
 jumps = c(0, na.fit$time, max(time))  
 # need to be careful at the beginning and end  
 surv = c(1, na.fit$surv, na.fit$surv[length(na.fit$surv)])  
  
 # apply appropriate transformation  
 neglogsurv = -log(surv)   
   
 # create placeholder of correct length  
 naest = numeric(length(time))   
 for (i in 2:length(jumps)) {  
 naest[which(time>=jumps[i-1] & time<=jumps[i])] =   
 neglogsurv[i-1] # snag the appropriate value  
 }  
 return(sort(unique(naest)))  
}  
  
#TEST SET VALIDATED BY TABLE 4.2 in text no page 95  
# leukemia\_6MP\_NA=getNATable(leukemia\_6MP,"+")  
# leukemia\_6MP\_NA\_R=survfit(Surv(as.numeric(sub("+","",leukemia\_6MP,fixed=TRUE)),ifelse(grepl("+",leukemia\_6MP,fixed=TRUE),0,1))~1,conf.type="log-log",type="fh")  
# summary(leukemia\_6MP\_NA\_R)  
# plot(leukemia\_6MP\_NA\_R,xlab="Time",ylab="Survival Probability",main="leukemia\_6MP\_NA\_R survival curve")  
# leukemia\_6MP\_NA\_CH\_R=getCumulativeHazardNA(as.numeric(sub("+","",leukemia\_6MP,fixed=TRUE)),ifelse(grepl("+",leukemia\_6MP,fixed=TRUE),0,1))  
  
azt\_ddc\_NA=getNATable(azt\_ddc,"+")  
azt\_ddc\_NA\_R=survfit(Surv(as.numeric(sub("+","",azt\_ddc,fixed=TRUE)),ifelse(grepl("+",azt\_ddc,fixed=TRUE),0,1))~1,conf.type="log-log",type="fh")  
summary(azt\_ddc\_NA\_R)

## Call: survfit(formula = Surv(as.numeric(sub("+", "", azt\_ddc, fixed = TRUE)),   
## ifelse(grepl("+", azt\_ddc, fixed = TRUE), 0, 1)) ~ 1, conf.type = "log-log",   
## type = "fh")  
##   
## time n.risk n.event survival std.err lower 95% CI upper 95% CI  
## 6 16 1 0.939 0.0606 0.62304 0.992  
## 11 15 1 0.879 0.0830 0.58170 0.970  
## 12 14 1 0.818 0.0983 0.52281 0.940  
## 32 13 1 0.758 0.1094 0.46341 0.905  
## 35 12 1 0.697 0.1175 0.40600 0.865  
## 39 10 1 0.631 0.1254 0.34186 0.820  
## 45 9 1 0.564 0.1304 0.28289 0.772  
## 49 8 1 0.498 0.1330 0.22843 0.720  
## 75 7 1 0.432 0.1331 0.17823 0.664  
## 80 6 1 0.366 0.1310 0.13234 0.606  
## 84 5 1 0.299 0.1264 0.09106 0.545  
## 85 4 1 0.233 0.1192 0.05506 0.481  
## 87 3 1 0.167 0.1093 0.02560 0.417  
## 102 2 1 0.101 0.0976 0.00539 0.367

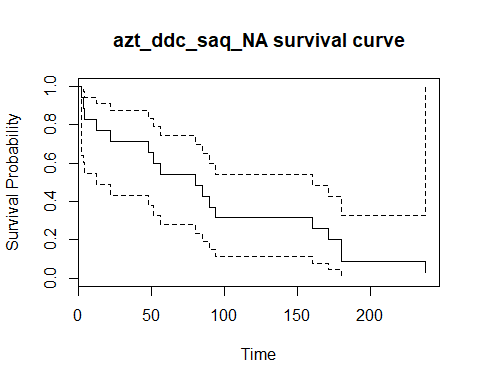
plot(azt\_ddc\_NA\_R,xlab="Time",ylab="Survival Probability",main="azt\_ddc\_NA survival curve")



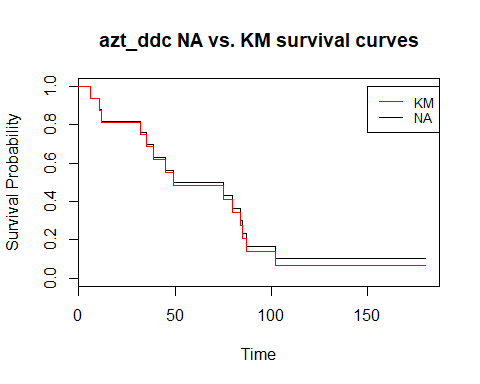
#cumulative hazard, confirmation of results  
azt\_ddc\_NA\_CH\_R=getCumulativeHazardNA(as.numeric(sub("+","",azt\_ddc,fixed=TRUE)),ifelse(grepl("+",azt\_ddc,fixed=TRUE),0,1))  
# write.table(azt\_ddc\_NA,file="C:/Users/an052283/OneDrive - Cerner Corporation/MSASA/Stat-845/Homeworks/output/Homework1/azt\_ddc\_NA.tsv",sep="\t",row.names = FALSE)  
View(azt\_ddc\_NA)  
azt\_ddc\_saq\_NA=getNATable(azt\_ddc\_saq,"+")  
azt\_ddc\_saq\_NA\_R = survfit(Surv(as.numeric(sub("+","",azt\_ddc\_saq,fixed=TRUE)),ifelse(grepl("+",azt\_ddc\_saq,fixed=TRUE),0,1))~1,conf.type="log-log",type="fh")  
summary(azt\_ddc\_saq\_NA\_R)

## Call: survfit(formula = Surv(as.numeric(sub("+", "", azt\_ddc\_saq, fixed = TRUE)),   
## ifelse(grepl("+", azt\_ddc\_saq, fixed = TRUE), 0, 1)) ~ 1,   
## conf.type = "log-log", type = "fh")  
##   
## time n.risk n.event survival std.err lower 95% CI upper 95% CI  
## 2 17 1 0.9429 0.0572 0.6417 0.992  
## 3 16 1 0.8857 0.0784 0.6021 0.971  
## 4 15 1 0.8286 0.0930 0.5455 0.943  
## 12 14 1 0.7715 0.1038 0.4883 0.910  
## 22 13 1 0.7144 0.1118 0.4328 0.874  
## 48 12 1 0.6573 0.1177 0.3795 0.834  
## 51 11 1 0.6001 0.1218 0.3287 0.791  
## 56 10 1 0.5430 0.1242 0.2802 0.746  
## 80 9 1 0.4859 0.1250 0.2343 0.698  
## 85 8 1 0.4288 0.1243 0.1908 0.649  
## 90 7 1 0.3717 0.1221 0.1501 0.597  
## 94 6 1 0.3147 0.1182 0.1124 0.543  
## 160 5 1 0.2576 0.1126 0.0780 0.486  
## 171 4 1 0.2006 0.1051 0.0476 0.428  
## 180 3 2 0.0872 0.0846 0.0049 0.327  
## 238 1 1 0.0321 Inf NaN 1.000

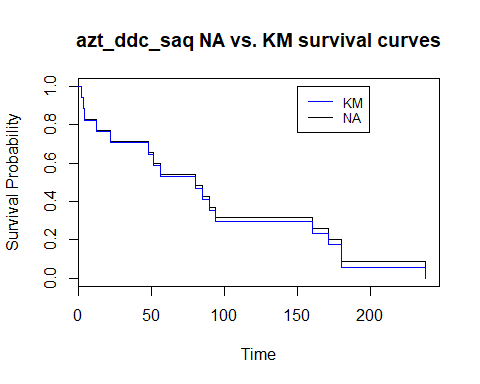
plot(azt\_ddc\_saq\_NA\_R,xlab="Time",ylab="Survival Probability",main="azt\_ddc\_saq\_NA survival curve")



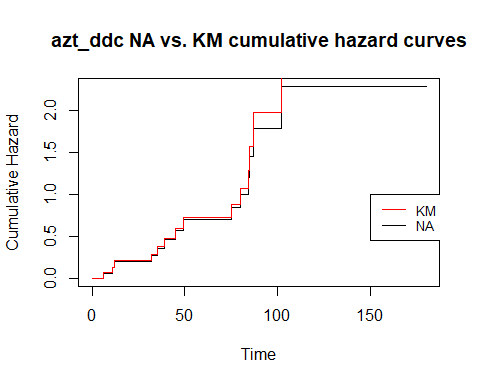
#cumulative hazard, confirmation of results  
azt\_ddc\_saq\_NA\_CH\_R=getCumulativeHazardNA(as.numeric(sub("+","",azt\_ddc\_saq,fixed=TRUE)),ifelse(grepl("+",azt\_ddc\_saq,fixed=TRUE),0,1))  
View(azt\_ddc\_saq\_NA)  
#write.table(azt\_ddc\_saq\_NA,file="C:/Users/an052283/OneDrive - Cerner Corporation/MSASA/Stat-845/Homeworks/output/Homework1/azt\_ddc\_saq\_NA.tsv",sep="\t",row.names = FALSE)  
  
#compare NA to KM survival curve graphically  
#azt\_ddc  
plot(survfit(Surv(as.numeric(sub("+","",azt\_ddc,fixed=TRUE)),ifelse(grepl("+",azt\_ddc,fixed=TRUE),0,1))~1,conf.type="none",type="fh"),xlab="Time",ylab="Survival Probability",main="azt\_ddc NA vs. KM survival curves")  
lines(survfit(Surv(as.numeric(sub("+","",azt\_ddc,fixed=TRUE)),ifelse(grepl("+",azt\_ddc,fixed=TRUE),0,1))~1,conf.type="none",type="kaplan-meier"),col="red")  
legend(150, 1, legend=c("KM", "NA"),  
 col=c("red", "black"), lty=1,cex=0.8)



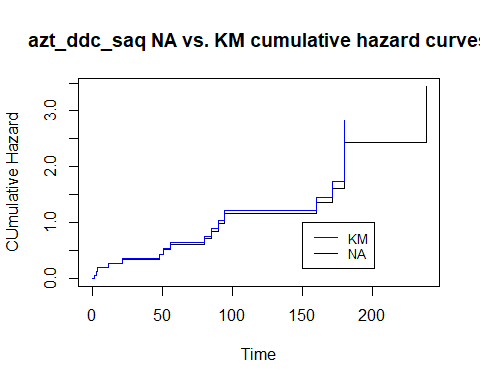
#azt\_ddc\_saq  
plot(survfit(Surv(as.numeric(sub("+","",azt\_ddc\_saq,fixed=TRUE)),ifelse(grepl("+",azt\_ddc\_saq,fixed=TRUE),0,1))~1,conf.type="none",type="fh"),xlab="Time",ylab="Survival Probability",main="azt\_ddc\_saq NA vs. KM survival curves")  
lines(survfit(Surv(as.numeric(sub("+","",azt\_ddc\_saq,fixed=TRUE)),ifelse(grepl("+",azt\_ddc\_saq,fixed=TRUE),0,1))~1,conf.type="none",type="kaplan-meier"),col="blue")  
legend(150, 1, legend=c("KM", "NA"),  
 col=c("blue", "black"), lty=1,cex=0.8)



#compare NA to KM cumulative hazard curve graphically  
plot(survfit(Surv(as.numeric(sub("+","",azt\_ddc,fixed=TRUE)),ifelse(grepl("+",azt\_ddc,fixed=TRUE),0,1))~1,conf.type="none",type="fh"),xlab="Time",ylab="Cumulative Hazard",main="azt\_ddc NA vs. KM cumulative hazard curves",fun="cumhaz")  
lines(survfit(Surv(as.numeric(sub("+","",azt\_ddc,fixed=TRUE)),ifelse(grepl("+",azt\_ddc,fixed=TRUE),0,1))~1,conf.type="none",type="kaplan-meier"),col="red",fun="cumhaz")  
legend(150, 1, legend=c("KM", "NA"),  
 col=c("red", "black"), lty=1,cex=0.8)



#azt\_ddc\_saq  
plot(survfit(Surv(as.numeric(sub("+","",azt\_ddc\_saq,fixed=TRUE)),ifelse(grepl("+",azt\_ddc\_saq,fixed=TRUE),0,1))~1,conf.type="none",type="fh"),xlab="Time",ylab="CUmulative Hazard",main="azt\_ddc\_saq NA vs. KM cumulative hazard curves",fun="cumhaz")  
lines(survfit(Surv(as.numeric(sub("+","",azt\_ddc\_saq,fixed=TRUE)),ifelse(grepl("+",azt\_ddc\_saq,fixed=TRUE),0,1))~1,conf.type="none",type="kaplan-meier"),col="blue",fun="cumhaz")  
legend(150, 1, legend=c("KM", "NA"),  
 col=c("blue", "black"), lty=1,cex=0.8)



library(KMsurv)

## Warning: package 'KMsurv' was built under R version 3.5.2

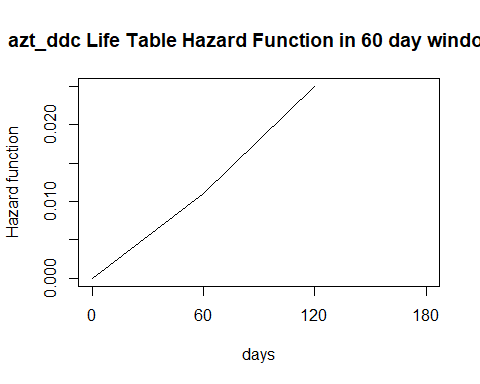
library(tidyverse)

## -- Attaching packages ----------------------------------------------------------------------------------------------------------------------- tidyverse 1.2.1 --

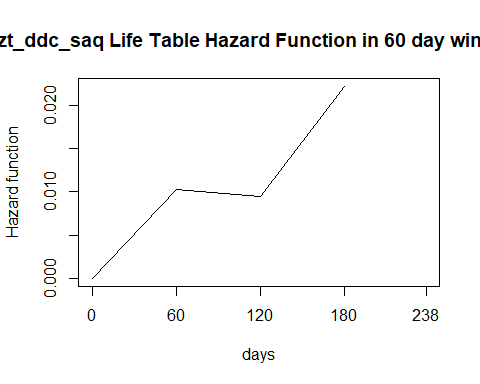
## v ggplot2 3.1.0 v purrr 0.2.5  
## v tibble 1.4.2 v dplyr 0.7.6  
## v tidyr 0.8.1 v stringr 1.3.1  
## v readr 1.2.1 v forcats 0.3.0

## -- Conflicts -------------------------------------------------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

#ninit = length(observations)  
  
#vector of time points, whose length is 1 greater than nlost and nevent  
# azt\_ddc\_c=ifelse(grepl("+",azt\_ddc,fixed=TRUE),1,0)  
  
getLifeTableInput = function(censoredTimesVector,censorSymbol){  
 #get numeric representation of censor vector  
 censoredTimesVectorNumeric=as.numeric(sub(censorSymbol,'',censoredTimesVector,fixed=TRUE))  
 #count number of actual rows in KM table  
 cnt\_n=length(censoredTimesVectorNumeric)  
 #create first row of KM table  
 lifeTableInputTable=setNames(data.frame(matrix(nrow=1,c(NA,NA,NA)),stringsAsFactors=FALSE),c("time","nlost","nevent"))  
 censoredTimesVectorNumeric=sort(censoredTimesVectorNumeric)  
 for (i in 1:max(censoredTimesVectorNumeric)){  
 if(i %in% censoredTimesVectorNumeric){  
 #create empty row to fill in  
 lifeTableInputRow=setNames(data.frame(matrix(NA,nrow=1,ncol=length(names(lifeTableInputTable)))),names(lifeTableInputTable))  
 lifeTableInputRow$time=i  
 #count how many events at time  
 lifeTableInputRow$nevent=length(which(censoredTimesVector==i))  
 #count how many censured at time  
 lifeTableInputRow$nlost=length(which(censoredTimesVector==paste0(i,censorSymbol)))  
 lifeTableInputTable=rbind(lifeTableInputTable,lifeTableInputRow)  
 }  
 }  
 na.omit(lifeTableInputTable)  
}  
azt\_ddc\_numeric=as.numeric(gsub("+","",azt\_ddc,fixed=TRUE))  
cuts\_ad=seq(0,max(azt\_ddc\_numeric),60)  
#ensure no loss of upper bound when incrementing  
if(max(azt\_ddc\_numeric)>cuts\_ad[length(cuts\_ad)]){cuts\_ad=cuts\_ad=c(cuts\_ad,(max(azt\_ddc\_numeric)))}  
azt\_ddc\_lt\_raw=getLifeTableInput(azt\_ddc,"+")  
lifetab\_dat=mutate(azt\_ddc\_lt\_raw,time\_cat = cut(time, cuts\_ad)) %>% group\_by(time\_cat) %>% summarize(ilost=sum(nlost),ievent=sum(nevent))  
azt\_ddc\_lt=lifetab(tis = c(0,azt\_ddc\_lt\_raw$time), ninit = length(azt\_ddc), nlost = azt\_ddc\_lt\_raw$nlost, nevent = azt\_ddc\_lt\_raw$nevent) %>% drop\_na(hazard)  
azt\_ddc\_lt\_60=lifetab(tis = cuts\_ad, ninit = length(azt\_ddc), nlost = lifetab\_dat$ilost, nevent = lifetab\_dat$ievent)   
  
azt\_ddc\_saq\_numeric=as.numeric(gsub("+","",azt\_ddc\_saq,fixed=TRUE))  
cuts\_ads=seq(0,max(azt\_ddc\_saq\_numeric),by=60)  
#ensure no loss of upper bound when incrementing  
if(max(azt\_ddc\_saq\_numeric)>cuts\_ads[length(cuts\_ads)]){cuts\_ads=c(cuts\_ads,(max(azt\_ddc\_saq\_numeric)))}  
azt\_ddc\_saq\_lt\_raw=getLifeTableInput(azt\_ddc\_saq,"+")  
lifetab\_dat=mutate(azt\_ddc\_saq\_lt\_raw,time\_cat = cut(time, cuts\_ads)) %>% group\_by(time\_cat) %>% summarize(ilost=sum(nlost),ievent=sum(nevent))  
azt\_ddc\_saq\_lt=lifetab(tis = c(0,azt\_ddc\_saq\_lt\_raw$time), ninit = length(azt\_ddc\_saq), nlost = azt\_ddc\_saq\_lt\_raw$nlost, nevent = azt\_ddc\_saq\_lt\_raw$nevent) %>% drop\_na(hazard)  
azt\_ddc\_saq\_lt\_60=lifetab(tis = cuts\_ads, ninit = length(azt\_ddc\_saq), nlost = lifetab\_dat$ilost, nevent = lifetab\_dat$ievent)   
  
#plot azt\_ddc  
plot(cuts\_ad,c(0,azt\_ddc\_lt\_60$hazard),type='l',ylab="Hazard function",xlab="days",xaxt="n",main="azt\_ddc Life Table Hazard Function in 60 day windows")  
axis(1, at = cuts\_ad, las=1)



#plot azt\_ddc\_saq  
plot(cuts\_ads,c(0,azt\_ddc\_saq\_lt\_60$hazard),type='l',ylab="Hazard function",xlab="days",xaxt="n",main="azt\_ddc\_saq Life Table Hazard Function in 60 day windows")  
axis(1, at = cuts\_ads, las=1)



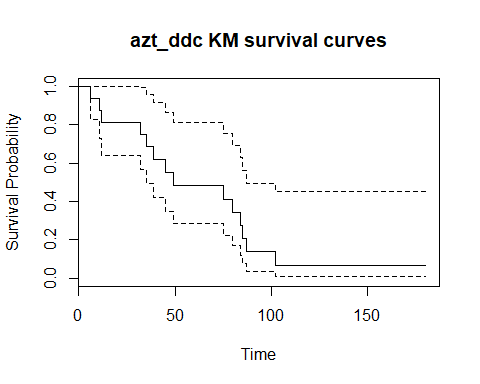
library(kmconfband)

## Warning: package 'kmconfband' was built under R version 3.5.2

#survival function confidence intervals for azt\_ddc  
azt\_ddc\_s=survfit(Surv(as.numeric(sub("+","",azt\_ddc,fixed=TRUE)),ifelse(grepl("+",azt\_ddc,fixed=TRUE),0,1))~1)  
azt\_ddc\_s\_ci=summary(azt\_ddc\_s)  
azt\_ddc\_s\_ci\_df=data.frame(azt\_ddc\_s\_ci$time,azt\_ddc\_s\_ci$n.risk,azt\_ddc\_s\_ci$n.event,azt\_ddc\_s\_ci$surv,azt\_ddc\_s\_ci$std.err,azt\_ddc\_s\_ci$lower,azt\_ddc\_s\_ci$upper)  
#lower and upper intervals and bounds  
azt\_ddc\_s\_ci\_cb=setNames(cbind(azt\_ddc\_s\_ci\_df,confband(azt\_ddc\_s)[1:dim(azt\_ddc\_s\_ci\_df)[1],]),c("time","n.risk","n.event","survival","std.err","lower 95% CI" ,"upper 95% CI","lower 95% CB" ,"upper 95% CB"))

## The critical value required is 0.4404776

plot(azt\_ddc\_s,xlab="Time",ylab="Survival Probability",main="azt\_ddc KM survival curves")



rtr\_example=c(3,4,"5+",6,"6+","8+",11,14,15,"16+")  
decrement=0  
#redistribute to right value  
getRedistributeToRightTable = function(censoredTimesVector,censorSymbol){  
 #get numeric representation of censor vector  
 censoredTimesVectorNumeric=as.numeric(sub(censorSymbol,'',censoredTimesVector,fixed=TRUE))  
 #sort to ensure when determine step# omit correct last element  
 censoredTimesVector=censoredTimesVector[order(censoredTimesVectorNumeric)]  
 #count number of actual rows in RTR table  
 cnt\_n=length(censoredTimesVectorNumeric)  
 steps=length(which(grepl(censorSymbol,censoredTimesVector[1:length(censoredTimesVector)-1],fixed=TRUE)))  
 #create first row of RTR table  
 rtrTable=setNames(data.frame(matrix(nrow=1,c(0,(1/cnt\_n),"",1)),stringsAsFactors=FALSE),c("observation","decrement","action","S\_t"))  
 censoredTimesVectorNumeric=sort(censoredTimesVectorNumeric)  
 decrement = 0  
 base=1/cnt\_n  
 i=0  
 uniqueCensoredTimesVector=unique(censoredTimesVector)  
 for (dataPoint in uniqueCensoredTimesVector){  
 i=i+1  
 #create empty row to fill in  
 rtrTableRow=setNames(data.frame(matrix(NA,nrow=1,ncol=length(names(rtrTable)))),names(rtrTable))  
 rtrTableRow$observation=dataPoint  
 #count how many events or censures at unique dataPoint  
 numerator=length(which(censoredTimesVector==dataPoint))  
 #censured at time?  
 if(length(which(censoredTimesVector==dataPoint & grep(censorSymbol,dataPoint,fixed=TRUE)))>0){  
 #no change in survival function  
 rtrTableRow$S\_t=rtrTable[dim(rtrTable)[1],c("S\_t")]  
 #display decrement of 0  
 rtrTableRow$decrement=0  
 #update denominator   
 denominator=length(censoredTimesVector)-max(which(dataPoint == censoredTimesVector))  
 if(i==length(uniqueCensoredTimesVector)){  
 rtrTableRow$action=paste0("survival is 1-(1)")  
 rtrTableRow$S\_t=0   
 rtrTableRow$decrement=1  
 }  
 else{  
 rtrTableRow$action=paste0("next time survival is 1-(",decrement," + ",base," + ",base,"\*",numerator,"/",denominator,")")  
 }  
 if(length(uniqueCensoredTimesVector)>=(i+1) && (!grepl(censorSymbol,uniqueCensoredTimesVector[i+1],fixed=TRUE))){  
 #update decrement only if not proceeded by a censored observation  
 decrement = decrement + base+(base\*numerator/denominator)  
 }  
 #update base  
 base=base+(base\*numerator/denominator)  
 }  
 else{  
 #update decrement and show action  
 if (as.numeric(rtrTable[dim(rtrTable)[1],c("decrement")])==0)  
 {  
 rtrTableRow$action=paste0("survival is 1-(",decrement,")")  
 decrement = decrement  
 }  
 else{  
 decrement = decrement + base  
 rtrTableRow$action=paste0("survival is 1-(",decrement,")")  
 }  
 rtrTableRow$S\_t=1-decrement  
 #update decrement display  
 rtrTableRow$decrement=decrement  
 }  
 #add row to rtrtable  
 rtrTable=rbind(rtrTable,rtrTableRow)  
 }  
 #eleminate initial bogus row  
 rtrTable$decrement=round(as.numeric(rtrTable$decrement),3)  
 rtrTable$S\_t=round(as.numeric(rtrTable$S\_t),3)  
 rtrTable[-1,]  
}  
  
azt\_ddc\_saq\_RTR=getRedistributeToRightTable(azt\_ddc\_saq,"+")  
azt\_ddc\_RTR=getRedistributeToRightTable(azt\_ddc,"+")  
rtr\_example\_RTR = getRedistributeToRightTable(rtr\_example,"+")

#column1 of book page 137 problem 4.7  
entry\_c1 = c(58,58,59,60,60,61,61,62,62,62,63,63,64,66,66)  
exit\_c1=c(60,63,69,62,65,72,69,73,66,65,68,74,71,68,69)  
death\_c1 = c(1,1,0,1,1,0,0,0,1,1,1,0,1,1,1)  
#column1 of book page 137 problem 4.7  
entry\_c2=c(67,67,67,68,69,69,69,70,70,70,71,72,72,73,73)  
exit\_c2=c(70,77,69,72,79,72,70,76,71,78,79,76,73,80,74)  
death\_c2=c(1,1,1,1,0,1,1,0,1,0,0,1,1,0,1)  
df2\_lec = data.frame(cbind(entry\_c1,exit\_c1,death\_c1))  
df2\_prob = setNames(data.frame(cbind(c(entry\_c1,entry\_c2),c(exit\_c1,exit\_c2),c(death\_c1,death\_c2))),c("entry","exit","death"))  
  
df2\_lec\_censored\_noLT=c(60,63,"69+",62,65,"72+","69+","73+",66,65,68,"74+",71,68,69)  
df2\_prob\_censored\_noLT=c(60,63,"69+",62,65,"72+","69+","73+",66,65,68,"74+",71,68,69,70,77,69,72,"79+",72,70,"76+",71,"78+","79+",76,73,"80+",74)  
  
getKM\_LT\_Table = function(entryExitDeathVector,deathSymbol){  
 #create first row of KM\_LT table  
 km\_ltTable=setNames(data.frame(matrix(nrow=1,c(0,0,0,length(entryExitDeathVector),"",1)),stringsAsFactors=FALSE),c("tj","ej","cj","nj","c\_tj-1","s\_tj"))  
 #sort by exit  
 orderedIndices=order(entryExitDeathVector$exit)  
 entryExitDeathVector=entryExitDeathVector[orderedIndices,]  
 for (time in unique(entryExitDeathVector$exit)){  
 total\_at\_risk=length(which(entryExitDeathVector$entry<=time))  
 gone=length(which(entryExitDeathVector$exit<time & entryExitDeathVector$death==deathSymbol))  
 events=length(which(entryExitDeathVector$exit==time & entryExitDeathVector$death==deathSymbol))  
 truncated\_at\_risk=total\_at\_risk-gone-as.numeric(km\_ltTable$cj[length(km\_ltTable$cj)])  
 censored=length(which(entryExitDeathVector$exit==time & entryExitDeathVector$death!=deathSymbol))  
 numerator=truncated\_at\_risk-events  
 denominator=truncated\_at\_risk  
 #print(paste0(time,": ",numerator,"/",denominator))  
 #create empty row to fill in  
 km\_ltTableRow=setNames(data.frame(matrix(NA,nrow=1,ncol=length(names(km\_ltTable)))),names(km\_ltTable))  
 km\_ltTableRow$tj=time  
 #count how many events at time  
 km\_ltTableRow$ej=events  
 #count how many censured at time  
 km\_ltTableRow$cj=censored  
 km\_ltTableRow$nj=truncated\_at\_risk  
 #sum events and number censored at time  
 km\_ltTableRow[c("c\_tj-1")]=paste0(numerator,"/",denominator)  
 km\_ltTableRow$s\_tj=round((numerator/denominator)\*as.numeric(km\_ltTable[dim(km\_ltTable)[1],c("s\_tj")]),4)  
 if (km\_ltTableRow$ej>0 | km\_ltTableRow$s\_tj==1){  
 #add row to km\_lttable  
 km\_ltTable=rbind(km\_ltTable,km\_ltTableRow)  
 }  
 }  
 km\_ltTable  
}  
df2\_lec\_LT=getKM\_LT\_Table(df2\_lec,1)  
View(df2\_lec\_LT)  
  
getKMTableNoCensorRemoval = function(censoredTimesVector,censorSymbol){  
 #get numeric representation of censor vector  
 censoredTimesVectorNumeric=as.numeric(sub(censorSymbol,'',censoredTimesVector,fixed=TRUE))  
 #count number of actual rows in KM table  
 cnt\_n=length(censoredTimesVectorNumeric)  
 #create first row of KM table  
 kmTable=setNames(data.frame(matrix(nrow=1,c(0,0,0,cnt\_n,as.character("-"),as.character(paste0(cnt\_n,"/",cnt\_n)),1)),stringsAsFactors=FALSE),c("orderedEventTimes\_tj","eventsAtEventTime\_ej",  
 "censoredObservationsInInterval\_cj","inRiskSetAtTime\_nj","kaplanMeirSurvivalCurveAtTime\_s\_tj-1","c\_tj-1","kaplanMeirSurvivalCurveAtTime\_s\_tj"))  
 # orderedIndices=order(censoredTimesVectorNumeric)  
 # censoredTimesVectorNumeric=censoredTimesVectorNumeric[orderedIndices]  
 # censoredTimesVector=censoredTimesVector[orderedIndices]  
 censoredTimesVectorNumeric=sort(censoredTimesVectorNumeric)  
 for (i in 1:max(censoredTimesVectorNumeric)){  
 if(i %in% censoredTimesVectorNumeric){  
 #create empty row to fill in  
 kmTableRow=setNames(data.frame(matrix(NA,nrow=1,ncol=length(names(kmTable)))),names(kmTable))  
 kmTableRow$orderedEventTimes\_tj=i  
 #count how many events at time  
 kmTableRow$eventsAtEventTime\_ej=length(which(censoredTimesVector==i))  
 #count how many censured at time  
 kmTableRow$censoredObservationsInInterval\_cj=length(which(censoredTimesVector==paste0(i,censorSymbol)))  
 kmTableRow$inRiskSetAtTime\_nj=cnt\_n  
 #sum events and censored  
 loss=kmTableRow$eventsAtEventTime\_ej+kmTableRow$censoredObservationsInInterval\_cj  
 kmTableRow[c("kaplanMeirSurvivalCurveAtTime\_s\_tj-1")]=kmTable[dim(kmTable)[1],c("kaplanMeirSurvivalCurveAtTime\_s\_tj")]  
 #TOOK LAZY WAY OUT AND JUST ADDED BACK IN THE CENSORED OBS - WILL DO CORRECT WAY LATER I Hope  
 numerator=(cnt\_n-loss+kmTableRow$censoredObservationsInInterval\_cj)  
 denominator=cnt\_n  
 kmTableRow[c("c\_tj-1")]=paste0(numerator,"/",denominator)  
 kmTableRow$kaplanMeirSurvivalCurveAtTime\_s\_tj=round(numerator/denominator\*as.numeric(kmTable[dim(kmTable)[1],c("kaplanMeirSurvivalCurveAtTime\_s\_tj")]),2)  
 #update count  
 cnt\_n=cnt\_n-loss  
 #don't add a row when no events 0 should put this at top but no time :0  
 if (kmTableRow$eventsAtEventTime\_ej>0 | cnt\_n==length(censoredTimesVectorNumeric)){  
 #add row to kmtable  
 kmTable=rbind(kmTable,kmTableRow)  
 }  
 }  
 }  
 kmTable  
}  
df2\_lec\_censored\_noLT\_KM=getKMTableNoCensorRemoval(df2\_lec\_censored\_noLT,"+")  
View(df2\_lec\_censored\_noLT\_KM)

4.7(a) Since the diabetics needed to survive long enough from birth until the study began, the data is left truncated. Construct a table showing the number of subjects at risk, Y, as a function of age.

#above code and data match output in lecture slide 33 output table - now try problem data  
df2\_prob\_LT=getKM\_LT\_Table(df2\_prob,1)  
View(df2\_prob\_LT)

df2\_prob\_censored\_noLT\_KM=getKMTableNoCensorRemoval(df2\_prob\_censored\_noLT,"+")  
View(df2\_prob\_censored\_noLT\_KM)