```
##### SIMULATION STUDY FOR THE IV PAPER VAN DEN BERG, BONEV, MAMMEN
##### WHAT IS THE IMPACT OF CENSORING?

# ASSUMPTION: COMPLIANCE AND DURATION ARE DEPENDEND!

# Load necessary packages:
library(splines)
library(survival)



# TYPE DEPENDENCE: noncompliers stochastically dominate compliers without treatment

# The following specifications are done (durations are only nonnegative):
# 1) Compliers without the treatment have duration distributions N(60, sd=15)
# 2) The treatment is obtained on day 20. The distribution function is shifted to N(30, sd=10).
# 3) The noncompliers have N(45, sd= 10)
# 4) The following possibilities are exploited:
# i) censoring is jointly independend from (Compliance, Duration)
# ii) censoring and compliance are dependend
# iii) censoring and duration are dependend (higher risk are more likely to be censored)
# iv) censoring is dependend both from compliance and duration


# Cohort 1
n= 10000                        # sample size
U.1=runif(n)            #  simulate individual duration probabilties under the uniform distr.
compl.prob=0.8          # the theoretical probability to be a complier at age 20
complier.1=sample(size=n,c(0,1),prob=c(1-compl.prob,compl.prob), replace=TRUE)      #
compliance indicator
T.1=vector()                    # vector for the duration realizations

for(i in 1:n) {if (complier.1[i]==0) T.1[i]=qnorm(U.1[i], mean=45, sd=15) else if (U.1[i]<
pnorm(20,mean=60, sd=15)) T.1[i] =qnorm(U.1[i], mean=60, sd=15) else T.1[i] =qnorm(U.1[i],
mean=30, sd=10)  }             #  Noncompliers obtain N(45, sd=15). Compliers obtain N(30, sd=10)
before day 50 and N(60, sd=15) after.

# Cohort 2
U.2=runif(n)            #  simulate individual duration probabilties under the uniform distr.
complier.2=sample(size=n,c(0,1),prob=c(0.2,0.8), replace=TRUE)          # compliance indicator
T.2=vector()                    # vector for the duration realizations
for(i in 1:n) {if (complier.2[i]==0) T.2[i]=qnorm(U.2[i], mean=45, sd=15) else T.2[i] =qnorm(U.2[i],
mean=60, sd=15)  }             #  Noncompliers obtain N(45, sd=15). Compliers obtain N(60, sd=15),
there is no treatment effect.


T.1.uncensored=T.1
T.2.uncensored=T.2
#####################################################################
# THEORETICAL TREATMENT EFFECT (= TE in the population)
#####################################################################
TE=vector()
for (i in 1:30)  TE[i] = compl.prob^(-1)  *  (         (   (1 - pnorm(20+i, mean=60, sd=15))*compl.prob +
(1 - pnorm(20+i, mean=45, sd=15))*(1- compl.prob)  )/ (    (1 - pnorm(20, mean=60,
sd=15))*compl.prob +  (1 - pnorm(20, mean=45, sd=15))*(1- compl.prob)   )   -     (   (1 -
```

```
pnorm(20+i, mean=30, sd=10))*compl.prob +  (1 - pnorm(20+i, mean=45, sd=15))*(1- compl.prob)
)/ (    (1 - pnorm(20, mean=30, sd=10))*compl.prob +  (1 - pnorm(20, mean=45, sd=15))*(1-
compl.prob)   )                )



##############################################
# Uncensored analysis: estimated treatment effect
Cens.1= rep(1, length(U.1))                    # censoring indicator, 1= uncensored. Here: all = 1.
Cens.2 = rep(1, length(U.2))
p.compl=mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))



model.data.t=Surv(T.1, event=Cens.1)       # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                                  #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)



TE.U= vector()                                 # The Treatment Effect
for(i in 1:30) TE.U[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl


rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)
##############################################
# Independent censoring NO! THIS IS NOT AN INDEPDENDENT CENSORING!
# Cens.1= sample(size=n,c(0,1),prob=c(0.2,0.8), replace=TRUE)           # censoring times.
# Cens.2 = sample(size=n,c(0,1),prob=c(0.2,0.8), replace=TRUE)                  # censoring times
# p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))



# model.data.t=Surv(T.1, event=Cens.1)     # prepares the data as an object for the function survfit
# model.data.u=Surv(T.2, event=Cens.2)
# S.t=survfit(model.data.t~1)                                  #   Kaplan Meier estimator, cohort 1
# S.u=survfit(model.data.u~1)



# TE.I= vector()                                 # The Treatment Effect
# for(i in 1:30) TE.I[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl



# rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)
##############################################
```

```
# Censoring dependend on compliance. Complier have lower risk of censoring.
 C.1 = vector()
for (i in 1:n){ if (complier.1[i]==1) C.1[i] = rnorm(1, mean= 30, sd =15) else C.1[i] =rnorm(1,
mean=50, sd=15) }
 C.2 = vector()
for (i in 1:n){ if (complier.2[i]==1) C.2[i] = rnorm(1, mean= 30, sd =15) else C.2[i] =rnorm(1,
mean=50, sd=15) }
Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
Cens.1=as.numeric(T.1.uncensored < C.1 )
Cens.2=as.numeric(T.2.uncensored < C.2 )

for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                                 #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                                 # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DC1= vector()                                # The Treatment Effect
for(i in 1:30) TE.DC1[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl


rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)
##############################################
# Censoring dependend on compliance 2. Complier have higher risk of censoring.
 C.1 = vector()
for (i in 1:n){ if (complier.1[i]==1) C.1[i] = rnorm(1, mean= 30, sd =15) else C.1[i] =rnorm(1,
mean=40, sd=15) }
 C.2 = vector()
for (i in 1:n){ if (complier.2[i]==1) C.2[i] = rnorm(1, mean= 30, sd =15) else C.2[i] =rnorm(1,
mean=40, sd=15) }
Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
Cens.1=as.numeric(T.1.uncensored < C.1 )
Cens.2=as.numeric(T.2.uncensored < C.2 )
```

```
for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                                  #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                                  # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DC2= vector()                                  # The Treatment Effect
for(i in 1:30) TE.DC2[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl



rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)
##############################################
# Censoring dependend on compliance 3. INDEPENDENT CENSORING CASE! DISTRIBUTION
DOES NOT DEPEND ON COMPLIANCE.
 C.1 = vector()
for (i in 1:n){ if (complier.1[i]==1) C.1[i] = rnorm(1, mean= 40, sd =15) else C.1[i] =rnorm(1,
mean=40, sd=15) }
 C.2 = vector()
for (i in 1:n){ if (complier.2[i]==1) C.2[i] = rnorm(1, mean= 40, sd =15) else C.2[i] =rnorm(1,
mean=40, sd=15) }
Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
Cens.1=as.numeric(T.1.uncensored < C.1 )
Cens.2=as.numeric(T.2.uncensored < C.2 )

for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
```

```
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                                  #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                                  # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DC3= vector()                                  # The Treatment Effect
for(i in 1:30) TE.DC3[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl
rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)
##############################################
# Censoring dependend on compliance 4.
 C.1 = vector()
for (i in 1:n){ if (complier.1[i]==1) C.1[i] = rnorm(1, mean= 40, sd =10) else C.1[i] =rnorm(1,
mean=30, sd=15) }
 C.2 = vector()
for (i in 1:n){ if (complier.2[i]==1) C.2[i] = rnorm(1, mean= 40, sd =10) else C.2[i] =rnorm(1,
mean=30, sd=15) }
Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
Cens.1=as.numeric(T.1.uncensored < C.1 )
Cens.2=as.numeric(T.2.uncensored < C.2 )

for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                                  #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                                  # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DC4= vector()                                  # The Treatment Effect
```

```
for(i in 1:30) TE.DC4[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl



rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)
##############################################
# Censoring dependend on compliance 3. Complier have higher risk of censoring.
 C.1 = vector()
for (i in 1:n){ if (complier.1[i]==1) C.1[i] = rnorm(1, mean= 50, sd =15) else C.1[i] =rnorm(1,
mean=30, sd=15) }
 C.2 = vector()
for (i in 1:n){ if (complier.2[i]==1) C.2[i] = rnorm(1, mean= 50, sd =15) else C.2[i] =rnorm(1,
mean=30, sd=15) }
Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
Cens.1=as.numeric(T.1.uncensored < C.1 )
Cens.2=as.numeric(T.2.uncensored < C.2 )

for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                          #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                          # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DC5= vector()                             # The Treatment Effect
for(i in 1:30) TE.DC5[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl



rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)
```

```
#####################################################
DURATION DEPENDENCE (DD) CENSORING
#############################################
# Censoring dependend on time 1. Higher risk - more likely to be censored.
 C.1 = vector()
for (i in 1:n){ if (T.1.uncensored [i]>40) C.1[i] = rnorm(1, mean= 30, sd =20) else C.1[i] =rnorm(1,
mean=40, sd=20) }
 C.2 = vector()
for (i in 1:n){ if ( T.2.uncensored[i] >40) C.2[i] = rnorm(1, mean= 30, sd =20) else C.2[i] =rnorm(1,
mean=40, sd=20) }

Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
Cens.1=as.numeric(T.1.uncensored < C.1 )
Cens.2 =as.numeric(T.2.uncensored < C.2 )

for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                          #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                          # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DD1= vector()                             # The Treatment Effect: duration and compliance
dependence
for(i in 1:30) TE.DD1[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl



rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)

#############################################
# Censoring dependend on time 2. Higher risk - more likely to be censored.
```

```
 C.1 = vector()
for (i in 1:n){ if (T.1.uncensored [i]>40) C.1[i] = rnorm(1, mean= 25, sd =20) else C.1[i] =rnorm(1,
mean=40, sd=20) }
 C.2 = vector()
for (i in 1:n){ if ( T.2.uncensored[i] >40) C.2[i] = rnorm(1, mean= 25, sd =20) else C.2[i] =rnorm(1,
mean=40, sd=20) }

Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
Cens.1=as.numeric(T.1.uncensored < C.1 )
Cens.2 =as.numeric(T.2.uncensored < C.2 )

for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                                 #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                                 # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DD2= vector()                            # The Treatment Effect: duration and compliance
dependence
for(i in 1:30) TE.DD2[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl


rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)

##############################################
# Censoring dependend on time 3. Higher risk - more likely to be censored.
 C.1 = vector()
for (i in 1:n){ if (T.1.uncensored [i]>40) C.1[i] = rnorm(1, mean= 20, sd =20) else C.1[i] =rnorm(1,
mean=40, sd=20) }
 C.2 = vector()
for (i in 1:n){ if ( T.2.uncensored[i] >40) C.2[i] = rnorm(1, mean= 20, sd =20) else C.2[i] =rnorm(1,
mean=40, sd=20) }

Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
Cens.1=as.numeric(T.1.uncensored < C.1 )
```

```
Cens.2 =as.numeric(T.2.uncensored < C.2 )

for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                                  #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                                  # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DD3= vector()                                 # The Treatment Effect: duration and compliance
dependence
for(i in 1:30) TE.DD3[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl



rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)


#############################################################################
# COMPLIANCE AND DURATION DEPENDENCE
#############################################
# Censoring dependend on time and compliance1. Higher risk - more likely to be censored.
Noncomplier more likely to be censored.
 C.1 = vector()
for (i in 1:n){ if (complier.1[i]==1 & T.1.uncensored [i]>30) C.1[i] = rnorm(1, mean= 30, sd =20) else
if (complier.1[i]==1 & T.1.uncensored[i] <30) C.1[i] =rnorm(1, mean=50, sd=20)  else if (complier.
1[i]==0 & T.1.uncensored [i]>30) C.1[i] =rnorm(1, mean=20, sd=20)  else C.1[i] =rnorm(1,
mean=30, sd=20)}
 C.2 = vector()
for (i in 1:n){ if (complier.2[i]==1 & T.2.uncensored [i]>30) C.2[i] = rnorm(1, mean= 30, sd =20) else
if (complier.2[i]==1 & T.2.uncensored[i] <30) C.2[i] =rnorm(1, mean=50, sd=20)  else if (complier.
2[i]==0 & T.2.uncensored [i]>30) C.2[i] =rnorm(1, mean=20, sd=20)  else C.2[i] =rnorm(1,
mean=30, sd=20)}


Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
```

```
Cens.1=as.numeric(T.1.uncensored < C.1 )
Cens.2 =as.numeric(T.2.uncensored < C.2 )

for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                                  #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                                  # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DDC1= vector()                            # The Treatment Effect: duration and compliance
dependence
for(i in 1:30) TE.DDC1[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl



rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)



##############################################
# Censoring dependend on time and compliance2. Higher risk - more likely to be censored.
Noncomplier more likely to be censored.
 C.1 = vector()
for (i in 1:n){ if (complier.1[i]==1 & T.1.uncensored [i]>30) C.1[i] = rnorm(1, mean= 30, sd =20) else
if (complier.1[i]==1 & T.1.uncensored[i] <30) C.1[i] =rnorm(1, mean=40, sd=20)  else if (complier.
1[i]==0 & T.1.uncensored [i]>30) C.1[i] =rnorm(1, mean=20, sd=20)  else C.1[i] =rnorm(1,
mean=30, sd=20)}
 C.2 = vector()
for (i in 1:n){ if (complier.2[i]==1 & T.2.uncensored [i]>30) C.2[i] = rnorm(1, mean= 30, sd =20) else
if (complier.2[i]==1 & T.2.uncensored[i] <30) C.2[i] =rnorm(1, mean=40, sd=10)  else if (complier.
2[i]==0 & T.2.uncensored [i]>30) C.2[i] =rnorm(1, mean=20, sd=15)  else C.2[i] =rnorm(1,
mean=30, sd=15)}


Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
```

```
Cens.1=as.numeric(T.1.uncensored < C.1 )
Cens.2 =as.numeric(T.2.uncensored < C.2 )

for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                                  #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                                  # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DDC2= vector()                            # The Treatment Effect: duration and compliance
dependence
for(i in 1:30) TE.DDC2[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl



rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)


##############################################
# Censoring dependend on time and compliance 3. Higher risk - more likely to be censored.
Noncomplier more likely to be censored.
 C.1 = vector()
for (i in 1:n){ if (complier.1[i]==1 & T.1.uncensored [i]>30) C.1[i] = rnorm(1, mean= 20, sd =20) else
if (complier.1[i]==1 & T.1.uncensored[i] <30) C.1[i] =rnorm(1, mean=30, sd=20)  else if (complier.
1[i]==0 & T.1.uncensored [i]>30) C.1[i] =rnorm(1, mean=30, sd=20)  else C.1[i] =rnorm(1,
mean=40, sd=20)}
 C.2 = vector()
for (i in 1:n){ if (complier.2[i]==1 & T.2.uncensored [i]>30) C.2[i] = rnorm(1, mean= 20, sd =20) else
if (complier.2[i]==1 & T.2.uncensored[i] <30) C.2[i] =rnorm(1, mean=30, sd=20)  else if (complier.
2[i]==0 & T.2.uncensored [i]>30) C.2[i] =rnorm(1, mean=30, sd=20)  else C.2[i] =rnorm(1,
mean=40, sd=20)}

Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
Cens.1=as.numeric(T.1.uncensored < C.1 )
Cens.2 =as.numeric(T.2.uncensored < C.2 )
```

```
for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                                  #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                                  # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DDC3= vector()                                  # The Treatment Effect: duration and compliance
dependence
for(i in 1:30) TE.DDC3[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl



rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)



##############################################
# Censoring dependend on time and compliance 4. Higher risk - more likely to be censored.
Noncomplier more likely to be censored.
 C.1 = vector()
for (i in 1:n){ if (complier.1[i]==1 & T.1.uncensored [i]>30) C.1[i] = rnorm(1, mean= 20, sd =20) else
if (complier.1[i]==1 & T.1.uncensored[i] <30) C.1[i] =rnorm(1, mean=30, sd=20)  else if (complier.
1[i]==0 & T.1.uncensored [i]>30) C.1[i] =rnorm(1, mean=30, sd=20)  else C.1[i] =rnorm(1,
mean=50, sd=20)}
 C.2 = vector()
for (i in 1:n){ if (complier.2[i]==1 & T.2.uncensored [i]>30) C.2[i] = rnorm(1, mean= 20, sd =20) else
if (complier.2[i]==1 & T.2.uncensored[i] <30) C.2[i] =rnorm(1, mean=30, sd=20)  else if (complier.
2[i]==0 & T.2.uncensored [i]>30) C.2[i] =rnorm(1, mean=30, sd=20)  else C.2[i] =rnorm(1,
mean=50, sd=20)}

Cens.1= vector()                # censoring indicator.
Cens.2 = vector()               # censoring indicator.
Cens.1=as.numeric(T.1.uncensored < C.1 )
Cens.2 =as.numeric(T.2.uncensored < C.2 )
```

```
for (i in 1:n){ if (C.1[i]<T.1.uncensored[i]) T.1[i]= C.1[i] else T.1[i]=T.1.uncensored[i]}
for (i in 1:n){ if (C.2[i]<T.2.uncensored[i]) T.2[i]= C.2[i] else T.2[i]=T.2.uncensored[i]}

# Alternative: actually, we are not interested how big are the censoring values. Only uncensored
values and the indicator function are needed for the estimation of the Survival function. The
problem is only for the probability to be a complier, there we need also the other probabilities.
# Cens.1=vector()
# Cens.2=vector()
# for(i in 1: n){if (complier.1==1) Cens.1[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
1[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}
# for(i in 1: n){if (complier.2==1) Cens.2[i]=sample(size=1,c=(0,1), prob=c(0.2,0.8))  else  Cens.
2[i]=sample(size=1,c=(0,1), prob=c(0.4,0.6))}


model.data.t=Surv(T.1, event=Cens.1)        # prepares the data as an object for the function survfit
model.data.u=Surv(T.2, event=Cens.2)
S.t=survfit(model.data.t~1)                                  #   Kaplan Meier estimator, cohort 1
S.u=survfit(model.data.u~1)                                  # Dependend on Compliance (DC)
p.compl = mean(as.numeric(complier.1==1 & T.1 >=20))/mean(as.numeric(T.1>=20))     # the
probability to be a complier conditionally on surviving up to 50


TE.DDC4= vector()                                  # The Treatment Effect: duration and compliance
dependence
for(i in 1:30) TE.DDC4[i] = (  S.u$surv[S.u$time==min(S.u$time[S.u$time>=i+20])]/S.u$surv[S.u
$time==min(S.u$time[S.u$time>=20])]- S.t$surv[S.t$time==min(S.t$time[S.t$time>=i+20])]/S.t
$surv[S.t$time==min(S.t$time[S.t$time>=20])]         )/p.compl



rm(Cens.1, Cens.2, model.data.t, model.data.u, S.t, S.u, p.compl)



jpeg("Simulation_Main.jpg")
plot(TE, type="l", xlab="time", ylab="Estimated Treatment Effect", main="Treatment Effect
Estimation and Independent Censoring")
lines(TE.DC3, col="red", lty=2)
dev.off()

jpeg("Simulation_Time_Dependence.jpg")
plot(TE, type="l", xlab="time", ylab="Estimated Treatment Effect", main="Estimation with
Censoring Depending on Time")
lines(TE.DD1, col="green", lty=2)
lines(TE.DD2, col="red", lty=3)
lines(TE.DD3, col="blue", lty=6)
dev.off()


jpeg("Simulation_Compliance_dependence.jpg")
```

```
plot(TE, type="l", xlab="time", ylab="Estimated Treatment Effect", main="Estimation with
Censoring Depending on Compliance")
lines(TE.DC1, col="green", lty=2)
lines(TE.DC2, col="red", lty=3)
lines(TE.DC4, col="blue", lty=5)
lines(TE.DC5, col="grey", lty=6)
dev.off()


jpeg("Simulation_Compliance_ And_Time_Dependence.jpg")
plot(TE, type="l", xlab="time", ylab="Estimated Treatment Effect", main="Estimation with
Censoring Depending on Compliance and Time")
lines(TE.DDC1, col="green", lty=2)
lines(TE.DDC2, col="red", lty=3)
lines(TE.DDC3, col="blue", lty=6)
lines(TE.DDC4, col="grey", lty=5)
dev.off()
```