smms: analysing CAV data with a Weibull-GenWeibull model, with covariates

Data

We will use the CAV dataset from the `msm` package (Jackson 2011) as an illustration. The dataset monitors a number of patients for a number of years after heart transplantation. Coronary allograft vasculopathy (CAV) is a condition potentially occurring after hear transplantation. At each time-point the patients are assigned to one of four states: well, mild CAV, severe CAV and death. The time of death is recorded precisely, but the times of entrance into the CAV-states are interval censored

Below we see the observations belonging to two patients. The states are numbered from 1 to 4, and I will use that numbering when specifying the graph below. After deleting some observations that are deemed incorrect (because they appear to get better, see next section), we end up with 2398 observations in 556 different patients.

There are several potential covariates in the dataset, but in the following we have only made use of two, `dage` the age of the heart transplant donor (standardized), and `ihd` a binary covariate related to the reason for transplantation (whether the primary diagnosis was ischaemic heart disease or not).

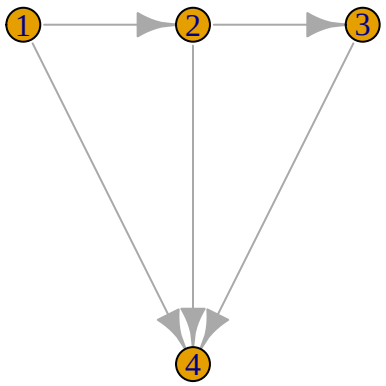
```
library(smms)
library(igraph) # For specifying the multi-state graph
library(msm) # To get the CAV dataset

dd = cav
dd = dd[!is.na(dd$pdiag),]

# Remove observations where the patient appears to go back to a previous state
# (assumed to be impossible):
id_wrong = unique(dd$PTNUM[which(dd$state!=dd$statemax)])
dd = dd[-which(dd$PTNUM %in% id_wrong),]

dd = dd[ ,-c(2, 5, 7, 9, 10)]
# rename relevant columns (necessary in current version):
colnames(dd)[1:2] <- c("patient","time")

# Covariates:
dd$dage_st = (dd$dage-mean(dd$dage))/sd(dd$dage)
dd$ihd = (dd$pdiag=="IHD")
colnames(dd)[1:2] <- c("patient","time")

X_data = aggregate(dd[,c("dage_st","ihd")],by=list(dd$patient),FUN=median)
#FUN does not matter
X_data = as.matrix(X_data[,2:3])

print(dd[1:11,])
##    patient     time dage pdiag state    dage_st  ihd
## 1   100002 0.000000   21   IHD     1 -0.6755105 TRUE
```

```
## 2   100002 1.002740   21   IHD     1 -0.6755105 TRUE
## 3   100002 2.002740   21   IHD     2 -0.6755105 TRUE
## 4   100002 3.093151   21   IHD     2 -0.6755105 TRUE
## 5   100002 4.000000   21   IHD     2 -0.6755105 TRUE
## 6   100002 4.997260   21   IHD     3 -0.6755105 TRUE
## 7   100002 5.854795   21   IHD     4 -0.6755105 TRUE
## 8   100003 0.000000   17   IHD     1 -1.0252054 TRUE
## 9   100003 1.189041   17   IHD     1 -1.0252054 TRUE
## 10  100003 2.008219   17   IHD     3 -1.0252054 TRUE
## 11  100003 2.991781   17   IHD     4 -1.0252054 TRUE
```

Specifying the model graph

Here we assume a four-state illness death model, since we consider CAV to be irreversible (so we do not allow for patients to move back to less severe states). It is convenient to stick to the same state names as in the dataset when specifying the model graph.

```
# Specify the graph:
gg = graph_from_literal("1"--+"2"--+"3"--+"4", "1"--+"4",
                        "2"--+"4")
par(mar=c(1,1,1,1))
plot(gg,layout=layout_with_sugiyama(gg,layers=c(1,1,1,2))$layout,vertex.size=20)
```



Specifying parametric models

Then, the user has to specify parametric models for all transition times (meaning one for each edge in the graph). In the current version of the package, these models have to be specified by providing density functions (in a specific format detailed below), as well as the corresponding survival functions. The functions will look like the following using a mix of ordinary Weibull models, and the generalised Weibull model. The models used here are described in more detail in the Aastveit, Cunen and Hjort (2022) paper.

```
# User defined density and survival function for the generalised Weibull:
pXweibull <- function(tt,a,b,th){
  pp <- 1-exp(1-(1+(tt/b)^a)^(1/th))
  pp[tt<0] <- 0 #to ensure that the survival function returns 1 when tt<0
  return(pp)
}
dXweibull <- function(tt,a,b,th){
  (1/th)*(a/b)*(tt/b)^(a-1)*(1+(tt/b)^a)^(1/th-1)*exp(1-(1+(tt/b)^a)^(1/th))
```

```
}

# Model:
S_01 = function(param, x, t){1-pweibull(t,exp(param[1]),
                                        exp(param[2]+param[3]*x[1]+param[4]*x[2]))}
S_12 = function(param, x, t){1-pweibull(t,exp(param[5]),
                                        exp(param[6]+param[7]*x[1]+param[8]*x[2]))}
S_23 = function(param, x, t){1-pweibull(t,exp(param[9]),
                                        exp(param[10]+param[11]*x[1]+param[12]*x[2]))}
S_03 = function(param, x, t){1-pXweibull(t,exp(param[13]),
                                         exp(param[14]+param[15]*x[1]+param[16]*x[2]),
                                         exp(param[17]))}
S_13 = function(param, x, t){1-pweibull(t,exp(param[18]),exp(param[19]))}


f_01 = function(param, x, t){dweibull(t,exp(param[1]),
                                      exp(param[2]+param[3]*x[1]+param[4]*x[2]))}
f_12 = function(param, x, t){dweibull(t,exp(param[5]),
                                      exp(param[6]+param[7]*x[1]+param[8]*x[2]))}
f_23 = function(param, x, t){dweibull(t,exp(param[9]),
                                      exp(param[10]+param[11]*x[1]+param[12]*x[2]))}
f_03 = function(param, x, t){dXweibull(t,exp(param[13]),
                                       exp(param[14]+param[15]*x[1]+param[16]*x[2]),
                                       exp(param[17]))}
f_13 = function(param, x, t){dweibull(t,exp(param[18]),exp(param[19]))}
```

Here we let the covariates influence all transition, except for the transition between mild and death ("1-3" in the interal naming system, "2-4" using the user defined names).

Important to note:

- **the names of the functions**: these have to be in the form `f_ij` and `S_ij` with i indicating the source state (in the internal numbering system) and j indicating the receiving state. More details on the naming convention below.
- **the arguments of the functions**: these should always be given as `(param, x, tt)` as above. `x` points to the vector of measured covariates (for a patient).
- **the scale of the parameters**: for the sake of stable optimisation it is convenient that the parameters live on the real line (instead of the positive half-line as in the common parameterisation of the exponential distribution). Therefore, we include an exponential transformation of the `param` vector, and we recommend that transformation for all positive parameters.
- **the ordering of the parameters**: `param` denotes the full parameter vector for the model.
- Survival functions should be written so that they return 1 when `tt` is negative. Using build-in R CDFs for distributions over the positive half-line will ensure this. Otherwise, if the user codes the survival functions herself, she should ensure that they return 1 when `tt` is negative (see the R chunck above).

As we saw above, the user has to follow a strict naming convention when specifying the densities and survival functions: within the package, the states are numbered from 0 to $k-1$ (k being the number of states), in a specific order which depends on the graph. To find out how the user defined state names relate to the internal numbering system, use the `names_of_survival_density` function. This is always recommended before specifying the model:

```
print(names_of_survival_density(gg))
##   edge_name survival_name density_name from_prev to_prev  type
## 1        01          S_01         f_01         1       2 trans
## 2        03          S_03         f_03         1       4   abs
## 3        12          S_12         f_12         2       3 trans
```

```
## 4        13          S_13         f_13         2       4   abs
## 5        23          S_23         f_23         3       4   abs
```

Here we see for example that the density for the edge between "2" and "3" should be named `f_12` (as we do above).

Fitting the model

Now we have everything in place in order to fit the multi-state model we have specified above:

```
startval <- c(0.36,2.38,-0.18,-0.30,-0.08,7.13,0.23,-0.33,
              -0.55,7.92,0.18,0.36,-0.98,7.35,0,0,-0.09,
              0,0)

mlo <- smms(startval,dd,gg,X_data, mc_cores = 5,hessian_matrix=T)
#calculates hessian too, slow with 1 core
```

One needs some start values for the optimisation, and we will soon add a function which calculates good starting values for a given model. Increasing the number of cores will make optimisation faster (but will not work on Windows machines). Here we choose to compute the hessian matrix too, which takes a bit more time. With one core this might take some time to compute.

We can compute AIC, and look at the estimated parameters and approximate 95% confidence intervals.

```
# Compute AIC (higher values are better with this definition)
aic <- (-2*mlo$opt$objective)-2*length(mlo$opt$par) #-2786.5

# Look at estimates and 95% confidence intervals.
# On the -Inf to Inf scale:
print(round(est_ci(mlo$opt$par,mlo$hess),2))
##    estimate lower.ci upper.ci
## 1      0.42     0.30     0.53
## 2      2.33     2.18     2.48
## 3     -0.15    -0.24    -0.07
## 4     -0.32    -0.50    -0.13
## 5     -0.13    -0.35     0.10
## 6      0.98     0.64     1.32
## 7      0.37     0.14     0.60
## 8     -0.37    -0.79     0.06
## 9     -0.43    -0.68    -0.17
## 10     0.53     0.06     1.00
## 11     0.38     0.04     0.72
## 12     0.24    -0.26     0.75
## 13    -0.47    -0.79    -0.14
## 14     1.07    -1.25     3.38
## 15    -1.26    -1.91    -0.61
## 16    -0.75    -2.00     0.49
## 17     2.08     1.08     3.09
## 18     2.08     1.20     2.96
## 19     1.85     1.70     2.00

# On the 0 to Inf scale (on the transition intensity scale):
round(exp(est_ci(mlo$opt$par,mlo$hess)),2)
##    estimate lower.ci upper.ci
```

```
## 1      1.52     1.35     1.71
## 2     10.28     8.85    11.93
## 3      0.86     0.78     0.94
## 4      0.73     0.60     0.88
## 5      0.88     0.70     1.11
## 6      2.67     1.89     3.76
## 7      1.45     1.15     1.82
## 8      0.69     0.45     1.06
## 9      0.65     0.50     0.85
## 10     1.70     1.06     2.71
## 11     1.46     1.04     2.06
## 12     1.28     0.77     2.11
## 13     0.63     0.45     0.87
## 14     2.91     0.29    29.42
## 15     0.28     0.15     0.54
## 16     0.47     0.14     1.64
## 17     8.03     2.94    21.98
## 18     8.00     3.31    19.32
## 19     6.35     5.46     7.38
```

Figures for interepretation and diagnostics

In the `smms` package, we have included code for computing various functions of interest, after having fitted a model. Confidence bands are available for all these functions, but computing these bands require differentiating the functions with respect to the parameters and that can be time-consuming for big models.

Occupancy probabilities

Gives the probability that a patient is found in state i at time t. The state should be provided using the user-defined names (hence 1-4 below, instead of 0-3). When covariates are present, one has to choose some specific covariate values for which to compute the probabilities (here I let the first covariate take values 1, older donor, and -1, younger donor, and the second covariate take value 0, no IHD, and 1, IHD).

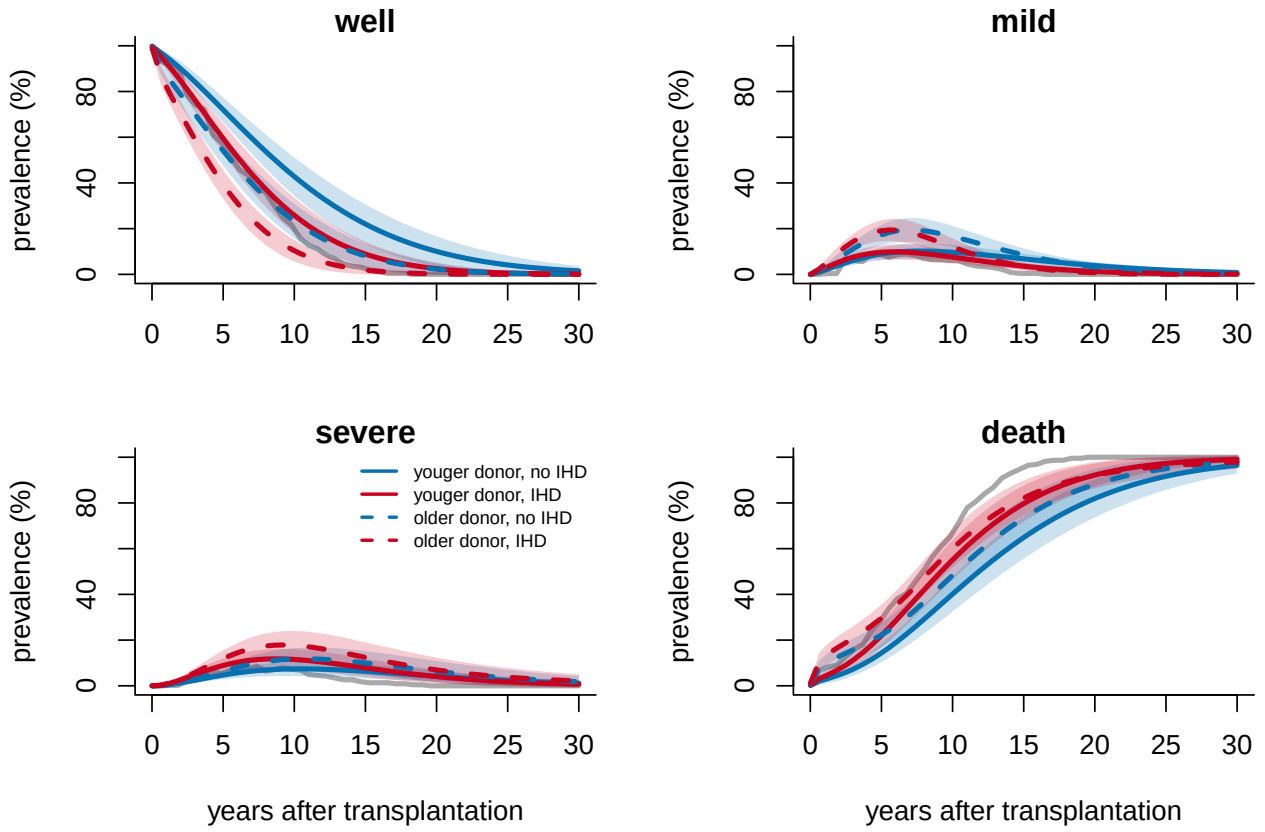
```
tval <- seq(0.01,30,length=50)
p0_y0_ci <- occupancy_prob_ci_band("1",tval,mlo$opt$par,gg,xval=c(-1,0),mlo$hess)
p0_y1_ci <- occupancy_prob_ci_band("1",tval,mlo$opt$par,gg,xval=c(-1,1),mlo$hess)
p0_o0_ci <- occupancy_prob_ci_band("1",tval,mlo$opt$par,gg,xval=c(1,0),mlo$hess)
p0_o1_ci <- occupancy_prob_ci_band("1",tval,mlo$opt$par,gg,xval=c(1,1),mlo$hess)
p1_y0_ci <- occupancy_prob_ci_band("2",tval,mlo$opt$par,gg,xval=c(-1,0),mlo$hess)
p1_y1_ci <- occupancy_prob_ci_band("2",tval,mlo$opt$par,gg,xval=c(-1,1),mlo$hess)
p1_o0_ci <- occupancy_prob_ci_band("2",tval,mlo$opt$par,gg,xval=c(1,0),mlo$hess)
p1_o1_ci <- occupancy_prob_ci_band("2",tval,mlo$opt$par,gg,xval=c(1,1),mlo$hess)
p2_y0_ci <- occupancy_prob_ci_band("3",tval,mlo$opt$par,gg,xval=c(-1,0),mlo$hess)
p2_y1_ci <- occupancy_prob_ci_band("3",tval,mlo$opt$par,gg,xval=c(-1,1),mlo$hess)
p2_o0_ci <- occupancy_prob_ci_band("3",tval,mlo$opt$par,gg,xval=c(1,0),mlo$hess)
p2_o1_ci <- occupancy_prob_ci_band("3",tval,mlo$opt$par,gg,xval=c(1,1),mlo$hess)
p3_y0_ci <- occupancy_prob_ci_band("4",tval,mlo$opt$par,gg,xval=c(-1,0),mlo$hess)
p3_y1_ci <- occupancy_prob_ci_band("4",tval,mlo$opt$par,gg,xval=c(-1,1),mlo$hess)
p3_o0_ci <- occupancy_prob_ci_band("4",tval,mlo$opt$par,gg,xval=c(1,0),mlo$hess)
p3_o1_ci <- occupancy_prob_ci_band("4",tval,mlo$opt$par,gg,xval=c(1,1),mlo$hess)
```

Plot the occupancy probabilities (the colored lines are the fitted state occupancies, the grey lines are the non-parametric estimates):

```
tval <- seq(0.01,30,length=50)
# msm package  (to get non-parameteric prevalence curves)
oneway4.q <- rbind(c(0, 0.25, 0, 0.25), c(0, 0, 0.25, 0.25),c(0, 0, 0, 0.5),
```

```
                    c(0, 0, 0, 0))
rownames(oneway4.q) <- colnames(oneway4.q) <- c("Well", "Mild","Severe",
                                                "Death")
cav.msm <- msm(state ~ time, subject = patient, data = dd,qmatrix = oneway4.q,
               death = 4,method = "BFGS")

prev_np <- prevalence.msm(cav.msm,times=tval)

# Plot
par(mfrow=c(2,2))
par(bty="l")
par(mar=c(4,4,1,2))
par(cex=1)
plot(tval,prev_np$`Observed percentages`[,1],type="l",ylim=c(0,100),lwd=3,
     xlab=" ",col="dark grey",ylab="prevalence (%)",main="well")
polygon(c(tval, rev(tval)), c(p0_y0_ci$upper*100, rev(p0_y0_ci$lower*100)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p0_y1_ci$upper*100, rev(p0_y1_ci$lower*100)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p0_o0_ci$upper*100, rev(p0_o0_ci$lower*100)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p0_o1_ci$upper*100, rev(p0_o1_ci$lower*100)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
lines(tval,p0_y0_ci$est*100,col="#0571b0",lwd=3)
lines(tval,p0_y1_ci$est*100,col="#ca0020",lwd=3)
lines(tval,p0_o0_ci$est*100,col="#0571b0",lwd=3,lty=2)
lines(tval,p0_o1_ci$est*100,col="#ca0020",lwd=3,lty=2)

plot(tval,prev_np$`Observed percentages`[,2],type="l",col="dark grey",
     ylim=c(0,100),lwd=3,xlab=" ",ylab="prevalence (%)",main="mild")
polygon(c(tval, rev(tval)), c(p1_y0_ci$upper*100, rev(p1_y0_ci$lower*100)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p1_y1_ci$upper*100, rev(p1_y1_ci$lower*100)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p1_o0_ci$upper*100, rev(p1_o0_ci$lower*100)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p1_o1_ci$upper*100, rev(p1_o1_ci$lower*100)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
lines(tval,p1_y0_ci$est*100,col="#0571b0",lwd=3)
lines(tval,p1_y1_ci$est*100,col="#ca0020",lwd=3)
lines(tval,p1_o0_ci$est*100,col="#0571b0",lwd=3,lty=2)
lines(tval,p1_o1_ci$est*100,col="#ca0020",lwd=3,lty=2)


plot(tval,prev_np$`Observed percentages`[,3],type="l",col="dark grey",
     ylim=c(0,100),lwd=3,xlab="years after transplantation",
     ylab="prevalence (%)",main="severe")
polygon(c(tval, rev(tval)), c(p2_y0_ci$upper*100, rev(p2_y0_ci$lower*100)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p2_y1_ci$upper*100, rev(p2_y1_ci$lower*100)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p2_o0_ci$upper*100, rev(p2_o0_ci$lower*100)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p2_o1_ci$upper*100, rev(p2_o1_ci$lower*100)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
lines(tval,p2_y0_ci$est*100,col="#0571b0",lwd=3)
lines(tval,p2_y1_ci$est*100,col="#ca0020",lwd=3)
lines(tval,p2_o0_ci$est*100,col="#0571b0",lwd=3,lty=2)
lines(tval,p2_o1_ci$est*100,col="#ca0020",lwd=3,lty=2)
legend("topright",legend=c("youger donor, no IHD","youger donor, IHD",
                           "older donor, no IHD","older donor, IHD"),
       col=c("#0571b0","#ca0020","#0571b0","#ca0020"),lwd=2,bty="n",
       lty=c(1,1,2,2),cex=0.7)

plot(tval,prev_np$`Observed percentages`[,4],type="l",col="dark grey",
     ylim=c(0,100),lwd=3,xlab="years after transplantation",
     ylab="prevalence (%)",main="death")
```

```
polygon(c(tval, rev(tval)), c(p3_y0_ci$upper*100, rev(p3_y0_ci$lower*100)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p3_y1_ci$upper*100, rev(p3_y1_ci$lower*100)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p3_o0_ci$upper*100, rev(p3_o0_ci$lower*100)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(p3_o1_ci$upper*100, rev(p3_o1_ci$lower*100)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
lines(tval,p3_y0_ci$est*100,col="#0571b0",lwd=3)
lines(tval,p3_y1_ci$est*100,col="#ca0020",lwd=3)
lines(tval,p3_o0_ci$est*100,col="#0571b0",lwd=3,lty=2)
lines(tval,p3_o1_ci$est*100,col="#ca0020",lwd=3,lty=2)
```



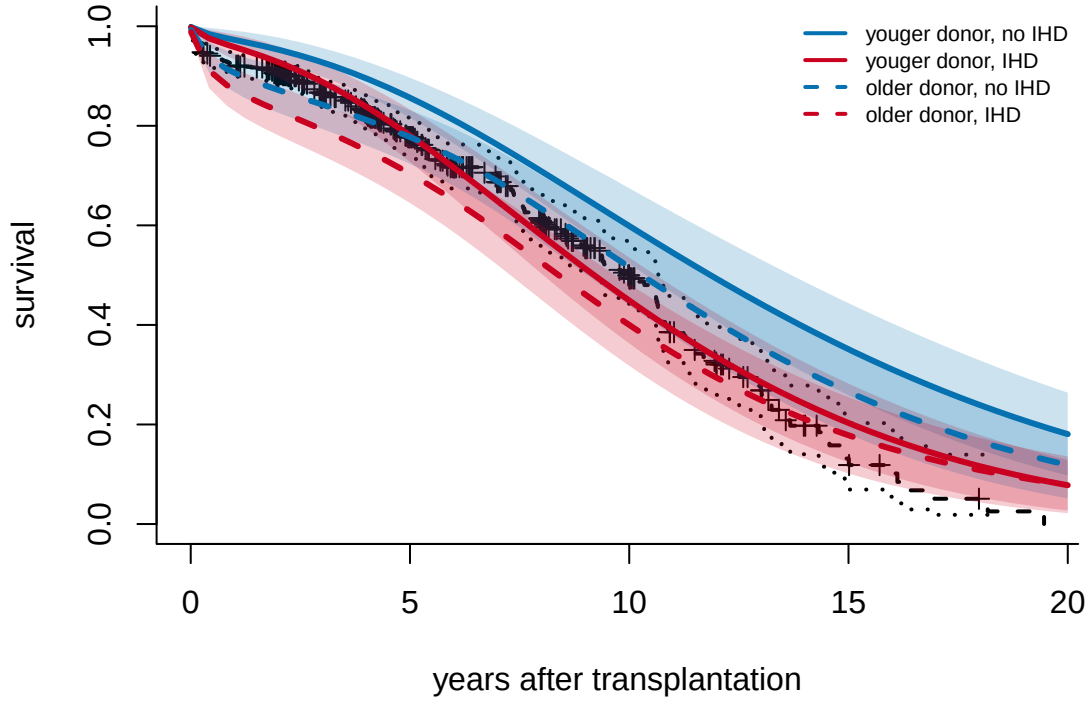
Overall survival

The overall survival curve gives the probability of not having reached an absorbing state at time t. The grey line is the Kaplan-Meier estimator, the colored lines are the fitted lines from the model. When covariates are present, one has to choose some specific covariate values for which to compute the survival curves (here I let the first covariate take values 1, older donor, and -1, younger donor, and the second covariate take value 0, no IHD, and 1, IHD).

```
tval <- seq(0.01,20,length=50)
Sy0 <- overall_survival_ci_band(tval,mlo$opt$par,gg,c(-1,0),mlo$hess)
Sy1 <- overall_survival_ci_band(tval,mlo$opt$par,gg,c(-1,1),mlo$hess)
So0 <- overall_survival_ci_band(tval,mlo$opt$par,gg,c(1,0),mlo$hess)
So1 <- overall_survival_ci_band(tval,mlo$opt$par,gg,c(1,1),mlo$hess)
```

```
tval <- seq(0.01,20,length=50)
par(mfrow=c(1,1))
par(bty="l")
par(mar=c(4,4,2,2))
```

```
par(cex=1)
plot.survfit.msm(cav.msm, col.surv="black",lwd.surv=2,
                 xlab="years after transplantation",
                 ylab="survival",main=" ",legend.pos=c(30,2),col=NULL,lwd=3)
# using msm package for the nonparametric estimate (for now)
polygon(c(tval, rev(tval)), c(Sy0$upper, rev(Sy0$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(Sy1$upper, rev(Sy1$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(So0$upper, rev(So0$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(So1$upper, rev(So1$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
lines(tval,Sy0$est,col="#0571b0",lwd=3)
lines(tval,Sy1$est,col="#ca0020",lwd=3)
lines(tval,So0$est,col="#0571b0",lwd=3,lty=2)
lines(tval,So1$est,col="#ca0020",lwd=3,lty=2)
legend("topright",legend=c("youger donor, no IHD","youger donor, IHD",
                           "older donor, no IHD","older donor, IHD"),
       col=c("#0571b0","#ca0020","#0571b0","#ca0020"),lwd=2,bty="n",
       lty=c(1,1,2,2),cex=0.7)
```



Transition probabilities

The probability that a patient is found in state i at a time-point t given that she was in state j at a prior time-point v. In the current implementation, t can be a vector of time-points while v is a single number. The user has to specify a particular edge or transition for which the function computes the transition probability, for example the transition between "mild" and "severe" (i.e. "2-3" using the user defined names). When covariates are present, one has to choose some specific covariate values for which to compute the survival curves (here I let the first covariate take values 1, older donor, and -1, younger donor, and the second covariate take value 0, no IHD, and 1, IHD).

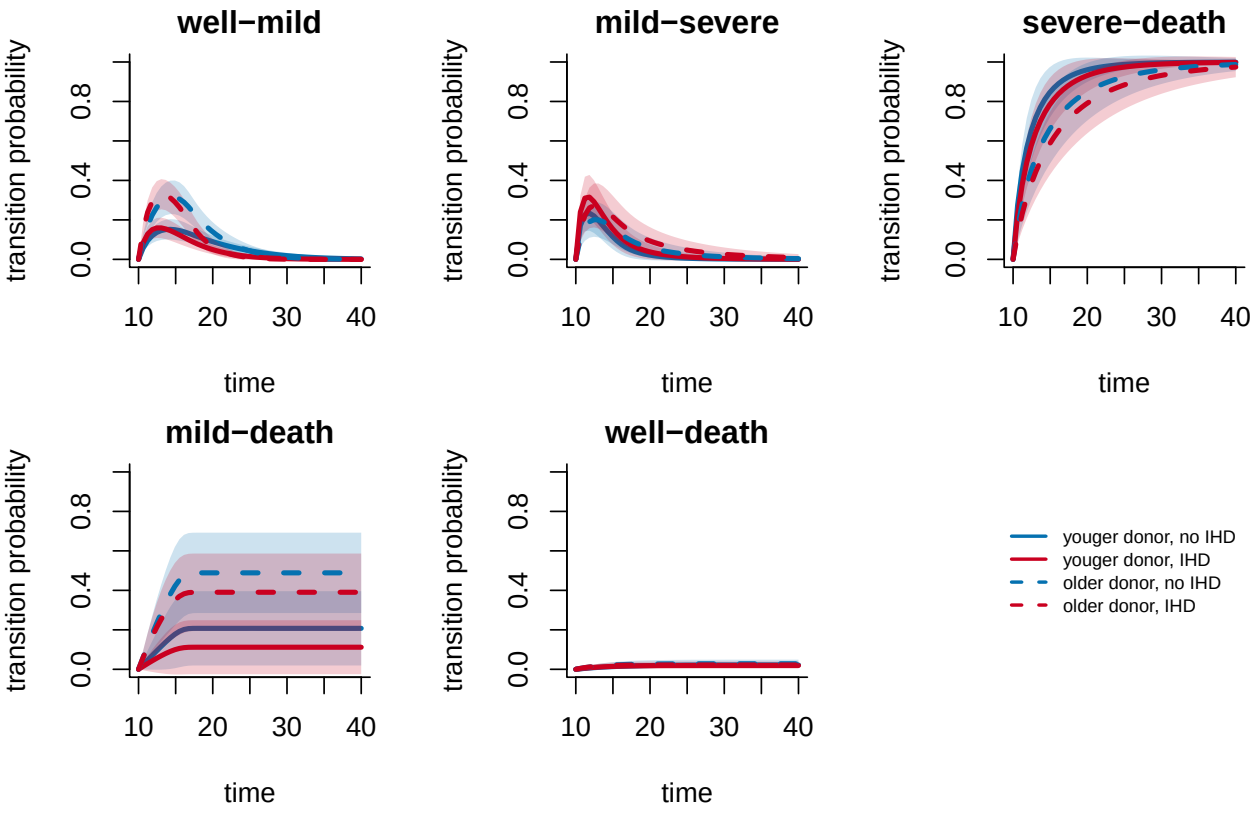
```
tval <- seq(10,40,length=50)
vt <- 10
t01_y0_ci <- transition_prob_ci_band("1-2",tval,vt,mlo$opt$par,gg,xval=c(-1,0),
```

```
                                    hessian=mlo$hess)
t01_y1_ci <- transition_prob_ci_band("1-2",tval,vt,mlo$opt$par,gg,xval=c(-1,1),
                                    hessian=mlo$hess)
t01_o0_ci <- transition_prob_ci_band("1-2",tval,vt,mlo$opt$par,gg,xval=c(1,0),hessian=mlo$hess)
t01_o1_ci <- transition_prob_ci_band("1-2",tval,vt,mlo$opt$par,gg,xval=c(1,1),hessian=mlo$hess)
t12_y0_ci <- transition_prob_ci_band("2-3",tval,vt,mlo$opt$par,gg,xval=c(-1,0),hessian=mlo$hess)
t12_y1_ci <- transition_prob_ci_band("2-3",tval,vt,mlo$opt$par,gg,xval=c(-1,1),hessian=mlo$hess)
t12_o0_ci <- transition_prob_ci_band("2-3",tval,vt,mlo$opt$par,gg,xval=c(1,0),hessian=mlo$hess)
t12_o1_ci <- transition_prob_ci_band("2-3",tval,vt,mlo$opt$par,gg,xval=c(1,1),hessian=mlo$hess)
t23_y0_ci <- transition_prob_ci_band("3-4",tval,vt,mlo$opt$par,gg,xval=c(-1,0),hessian=mlo$hess)
t23_y1_ci <- transition_prob_ci_band("3-4",tval,vt,mlo$opt$par,gg,xval=c(-1,1),hessian=mlo$hess)
t23_o0_ci <- transition_prob_ci_band("3-4",tval,vt,mlo$opt$par,gg,xval=c(1,0),hessian=mlo$hess)
t23_o1_ci <- transition_prob_ci_band("3-4",tval,vt,mlo$opt$par,gg,xval=c(1,1),hessian=mlo$hess)
t13_y0_ci <- transition_prob_ci_band("2-4",tval,vt,mlo$opt$par,gg,xval=c(-1,0),hessian=mlo$hess)
t13_y1_ci <- transition_prob_ci_band("2-4",tval,vt,mlo$opt$par,gg,xval=c(-1,1),hessian=mlo$hess)
t13_o0_ci <- transition_prob_ci_band("2-4",tval,vt,mlo$opt$par,gg,xval=c(1,0),hessian=mlo$hess)
t13_o1_ci <- transition_prob_ci_band("2-4",tval,vt,mlo$opt$par,gg,xval=c(1,1),hessian=mlo$hess)
t03_y0_ci <- transition_prob_ci_band("1-4",tval,vt,mlo$opt$par,gg,xval=c(-1,0),hessian=mlo$hess)
t03_y1_ci <- transition_prob_ci_band("1-4",tval,vt,mlo$opt$par,gg,xval=c(-1,1),hessian=mlo$hess)
t03_o0_ci <- transition_prob_ci_band("1-4",tval,vt,mlo$opt$par,gg,xval=c(1,0),hessian=mlo$hess)
t03_o1_ci <- transition_prob_ci_band("1-4",tval,vt,mlo$opt$par,gg,xval=c(1,1),hessian=mlo$hess)
```

Plot the transition probabilities (the colored lines are the fitted transition probabilities):

```
tval <- seq(10,40,length=50)

par(mfrow=c(2,3))
par(bty="l")
par(mar=c(4,4,2,2))
par(cex=1)
plot(tval,t01_y0_ci$est,type="l",ylim=c(0,1),col="#0571b0",lwd=3,
     ylab="transition probability",xlab="time",main="well-mild")
polygon(c(tval, rev(tval)), c(t01_y0_ci$upper, rev(t01_y0_ci$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t01_y1_ci$upper, rev(t01_y1_ci$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t01_o0_ci$upper, rev(t01_o0_ci$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t01_o1_ci$upper, rev(t01_o1_ci$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
lines(tval,t01_y1_ci$est,lwd=3,col="#ca0020")
lines(tval,t01_o0_ci$est,lwd=3,col="#0571b0",lty=2)
lines(tval,t01_o1_ci$est,lwd=3,col="#ca0020",lty=2)


plot(tval,t12_y0_ci$est,type="l",ylim=c(0,1),col="#0571b0",lwd=3,
     ylab="transition probability",xlab="time",main="mild-severe")
polygon(c(tval, rev(tval)), c(t12_y0_ci$upper, rev(t12_y0_ci$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t12_y1_ci$upper, rev(t12_y1_ci$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t12_o0_ci$upper, rev(t12_o0_ci$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t12_o1_ci$upper, rev(t12_o1_ci$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
lines(tval,t12_y1_ci$est,lwd=3,col="#ca0020")
lines(tval,t12_o0_ci$est,lwd=3,col="#0571b0",lty=2)
lines(tval,t12_o1_ci$est,lwd=3,col="#ca0020",lty=2)


plot(tval,t23_y0_ci$est,type="l",ylim=c(0,1),col="#0571b0",lwd=3,
     ylab="transition probability",xlab="time",main="severe-death")
polygon(c(tval, rev(tval)), c(t23_y0_ci$upper, rev(t23_y0_ci$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t23_y1_ci$upper, rev(t23_y1_ci$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t23_o0_ci$upper, rev(t23_o0_ci$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
```

```
polygon(c(tval, rev(tval)), c(t23_o1_ci$upper, rev(t23_o1_ci$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
lines(tval,t23_y1_ci$est,lwd=3,col="#ca0020")
lines(tval,t23_o0_ci$est,lwd=3,col="#0571b0",lty=2)
lines(tval,t23_o1_ci$est,lwd=3,col="#ca0020",lty=2)

plot(tval,t13_y0_ci$est,type="l",ylim=c(0,1),col="#0571b0",lwd=3,
     ylab="transition probability",xlab="time",main="mild-death")
polygon(c(tval, rev(tval)), c(t13_y0_ci$upper, rev(t13_y0_ci$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t13_y1_ci$upper, rev(t13_y1_ci$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t13_o0_ci$upper, rev(t13_o0_ci$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t13_o1_ci$upper, rev(t13_o1_ci$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
lines(tval,t13_y1_ci$est,lwd=3,col="#ca0020")
lines(tval,t13_o0_ci$est,lwd=3,col="#0571b0",lty=2)
lines(tval,t13_o1_ci$est,lwd=3,col="#ca0020",lty=2)

plot(tval,t03_y0_ci$est,type="l",ylim=c(0,1),col="#0571b0",lwd=3,
     ylab="transition probability",xlab="time",main="well-death")
polygon(c(tval, rev(tval)), c(t03_y0_ci$upper, rev(t03_y0_ci$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t03_y1_ci$upper, rev(t03_y1_ci$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t03_o0_ci$upper, rev(t03_o0_ci$lower)),
        col = adjustcolor("#0571b0",alpha.f=0.2), border=NA)
polygon(c(tval, rev(tval)), c(t03_o1_ci$upper, rev(t03_o1_ci$lower)),
        col = adjustcolor("#ca0020",alpha.f=0.2), border=NA)
lines(tval,t03_y1_ci$est,lwd=3,col="#ca0020")
lines(tval,t03_o0_ci$est,lwd=3,col="#0571b0",lty=2)
lines(tval,t03_o1_ci$est,lwd=3,col="#ca0020",lty=2)

plot(0, xaxt = 'n', yaxt = 'n', bty = 'n', pch = '', ylab = ''
     , xlab = '',ylim=c(0,1),xlim=c(0,40))
legend("right",legend=c("youger donor, no IHD","youger donor, IHD",
                        "older donor, no IHD","older donor, IHD"),
       col=c("#0571b0","#ca0020","#0571b0","#ca0020"),lwd=2,
       bty="n",lty=c(1,1,2,2),cex=0.7)
```


well−mild
mild−severe
severe−death
mild−death
well−death
transition probability
0.0
0.4
0.8
10
20
30
40
time
youger donor, no IHD
youger donor, IHD
older donor, no IHD
older donor, IHD