discfrail

Francesca Gasperoni

2018-12-02

Contents

Introduction

Package **discfrail** (**disc**rete **frail**ty) is an `R` package that provides novel method to deal with hierarchcal time-to-event data in which a clustering structure of groups is suspetcted. The package contains the implementation of the Cox model with a nonparametric discrete frailty term (Gasperoni *et al.* 2018) and two functions for simulating data.

Notation and Model

Consider a random sample with a hierarchical structure, i.e, where each statistical unit belongs to one group. Define T^*_{ij} as the survival time and C_{ij} as the censoring time of subject i, $i = 1, ..., n_j$, in the j-th group, $j = 1, ..., J$. Let $\mathbf{X}_{ij} = (X_{ij1}, ..., X_{ijp})^T$ be the vector of covariates, assumed constant over time, for subject i in group j. Then, we define $T_{ij} = min(T^*_{ij}, C_{ij})$, t_{ij} its realization and $\delta_{ij} = \mathbf{1}_{(T^*_{ij} \leq C_{ij})}$. Let $\tilde{\mathbf{w}}$ be the vector of shared random effects, and $\mathbf{w}$, $\mathbf{w} = \exp\{\tilde{\mathbf{w}}\}$, be the vector of shared frailties. In `discfrail` package, we consider a nonparametric frailty term, which is modeled through a random variable with discrete distribution, with an unknown number of points in the support. In particular, we assume that each group j can belong to one latent population k, $k = 1, ..., K$, with probability

π_k. In this case, $w_1, ..., w_K$ are the points in the support of w, K is the support's cardinality and $\mathbb{P}\{w = w_k\} = \pi_k$.

We introduce also an auxiliary indicator random variable z_{jk} which is equal to 1 if the j-th group belongs to the k-th population, so, considering k as a fixed term, $z_{jk} \overset{i.i.d}{\sim} Bern(\pi_k)$.

The requirement $\sum_{k=1}^{K} z_{jk} = 1$, for each j, is equivalent to the assumption that each group belongs to only one population. The vector $\mathbf{z}_j$ is distributed as a multinomial. Note that there are two levels of clustering: the first one is known (i.e., healthcare providers as clusters of patients) and we refer to these clusters as *groups*, while the second level is the unknown clustering of healthcare providers that we want to detect and we refer to these clusters as *latent populations.*

Than, the hazard function for individual i in group j conditional on w and on z_{jk} is:

$$\lambda(t; \mathbf{X}_{ij}, w_k, z_{jk}) = \prod_{k=1}^{K} \left[\lambda_0(t) w_k \exp(\mathbf{X}_{ij}^T \boldsymbol{\beta})\right]^{z_{jk}}, \qquad (1)$$

where $\lambda_0(t)$ represents the baseline hazard, $\boldsymbol{\beta}$ is the vector of regression coefficients and w_k is the frailty term shared among groups of the same latent population k. Both the frailty and the baseline hazard are assumed to be nonparametric, which makes model (1) an extension of a proportional hazard Cox model.

We assume that censoring is noninformative, thus that T_{ij}^* and C_{ij} are conditionally independent, given $\mathbf{X}_{ij}$, w_k and z_{jk}.

A real example of this theoretical structure is the time-to-readmission in hospital recorded for patients that are grouped in healthcare providers (Section 5 of Gasperoni *et al.* (2018)).

Simulation of hierarchical time-to-event data with a latent clustering structure of groups

Two functions are focused on the simulation of hierarchical time-to-event data in which a clustering structure of groups is present.

The input parameters of the simulation algorithm are:

- J: the number of groups;
- N_j: the number of individuals in each group (N_j can be a *scalar* and in this case N_j is assumed to be equal in all groups; N_j can be a *vector*, J x 1, or it can be *NULL* in that case $N_j \overset{i.i.d.}{\sim} Poisson(50)$);
- $\Lambda_0(t)$: the cumulative baseline hazard;
- $\boldsymbol{\pi}$: the K-dimensional probability vector of belonging to a single population;
- $\mathbf{w}$: the K-dimensional frailty vector;
- $\boldsymbol{\beta}$: the vector of regression parameters;
- the percentage of censored events.

Several steps are needed for building the dataset. The simulation model proposed in this work is based on the paper by Bender *et al.* (2005). We use two probability results to obtain Eq.(2): the inverse probability method and the fact that if a generic random variable V is distributed as $\mathcal{U}[0,1]$ then $1-V$ is still a $\mathcal{U}[0,1]$.

$$U_{ij} = 1 - F(t_{ij}; \mathbf{X}_{ij}, w) = \exp\{-\Lambda_0(T_{ij}) w_k \exp\{X_{ij}^T \beta\}\} \sim \mathcal{U}[0,1] \tag{2}$$

Then, we are able to compute the survival times with Eq.(3) by inverting Eq.(2).

$$T_{ij} = \Lambda_0^{-1}\left(\frac{-\log(U_{ij})}{w_k \exp\{\mathbf{X}_{ij}^T \beta\}}\right) \tag{3}$$

The main difference between equations (2), (3) and the ones showed in Bender *et al.* (2005) is the frailty term, w_k.

The covariates are randomly generated and normally distributed, $\mathbf{X}_{ij} \overset{i.i.d.}{\sim} N(0,1)$.

Exploiting equations (2) and (3) we are able to compute the times T_{ij}. Then, we generate the censoring times according to a Normal distribution, independently from the T_{ij}. Tuning the mean and the variance of this distribution, we have a higher (or lower) percentage of censored statistical units. Finally, we obtain the status variable as: $\delta_{ij} = \mathbf{1}_{(T_{ij}^* \leq C_{ij})}$.

A key point is the choice of the baseline. We propose two functions to address this issue:

- `sim_weibdf`, a parametric baseline (a Weibull distributed $\lambda_0(t_{ij})$ characterised by two paramters λ and ρ);
- `sim_npdf`, a nonparametric baseline (the user can implement Λ_0^{-1}).

In the first case, equation (3) becomes:

$$T_{ij} = \left(\frac{-\log(U_{ij})}{w_k \cdot \lambda \exp\{\mathbf{X}_{ij}^T \beta\}} \right)^{1/\rho} \tag{4}$$

```
library( discfrail )
```

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

```
#J is the total number of groups
J <- 100
#N is the number of units per group
N <- 40
#lambda: parameter of the Weibull distribution
lambda <- 0.5
#rho: parameter of the Weibull distribution
rho <- 1.4
#regression parameter
beta <- c( 1.6, 0.4 )
#vector of mixing proportion
p <- c( 0.7, 0.3 )
#vector of frailty values
w_values <- c( 1.2, 2.1 )
#percentage of censored events
cens_perc <- 0.1

set.seed(1200)
data_weib <- sim_weibdf( J, N, lambda, rho, beta, p, w_values, cens_perc)
```

```r
head( data_weib )
```

```
##   family       time status           x.1        x.2 belong
## 1      1 5.87449844      0 -0.848038891 -1.5494394    2.1
## 2      1 1.10231027      1 -0.047688486  1.8599481    2.1
## 3      1 0.66006984      1 -0.005320405  1.4354504    2.1
## 4      1 0.01513767      1  0.354866150  0.6519595    2.1
## 5      1 0.16845065      1  1.333647771  1.3741231    2.1
## 6      1 0.28319912      1  0.805291478  0.4975223    2.1
```

```r
#family represents the group index
table(data_weib$family)
```

```
## 
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18 
##  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40 
##  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36 
##  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40 
##  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54 
##  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40 
##  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72 
##  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40 
##  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90 
##  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40 
##  91  92  93  94  95  96  97  98  99 100 
##  40  40  40  40  40  40  40  40  40  40
```

```r
#status is the $\delta_{ij}
table(data_weib$status)/sum(table(data_weib$status))
```

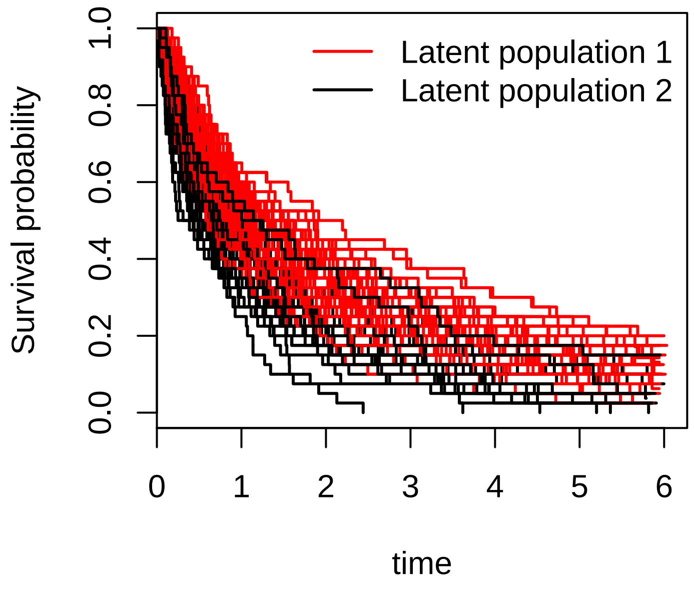
```
## 
##      0      1
## 0.1005 0.8995
```

```r
#belong is the w
table(data_weib$belong)/sum(table(data_weib$belong))
```

```
##
## 1.2 2.1
## 0.7 0.3
```

We can visualize the data_weib.

```r
fit1 <- coxph( Surv( time, status ) ~ family, data_weib, method = 'breslow')
sfit1 <- survfit(Surv( time, status ) ~ family, data_weib)

#setting the colors according to 'belong'
lat_pop = rep( 2, J )
lat_pop[ which( data_weib$belong[ seq( 1, dim(data_weib)[1], N ) ] %in%
                  w_values[2]) ] = 1

plot( sfit1, col = lat_pop, xlab = 'time', ylab = 'Survival probability',
      lwd = 1.5 )
legend( 'topright', paste('Latent population', 1:2), col = 2:1,
        lty = rep(1,2), lwd = 1.5, bty = 'n' )
```



We note from the plot of the group-specific Survival probability functions that there is a great variability related to groups.

We show an application of the second case:

```
J <- 100
N <- 40
Lambda_0_inv = function( t, c=0.01, d=4.6 ) ( t^( 1/d ) )/c
beta <- 1.6
p <- c( 0.8, 0.2 )
w_values <- c( 0.8, 1.6 )
cens_perc <- 0.2
data_np <- sim_npdf( J, N, beta, Lambda_0_inv, p, w_values, cens_perc)
head( data_np )
```

```
##   family      time status           x belong
## 1      1  68.63515      1  0.29441497    0.8
## 2      1  85.59369      1  0.23352922    0.8
## 3      1  30.41017      1  1.47340277    0.8
## 4      1 132.27330      0 -1.78088450    0.8
## 5      1 103.06927      1 -0.03710575    0.8
## 6      1 131.78735      0 -1.11903050    0.8
```

```
#family represents the group index
table(data_np$family)
```

```
##
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40
##  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40
##  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40
##  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40
##  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40  40
##  91  92  93  94  95  96  97  98  99 100
##  40  40  40  40  40  40  40  40  40  40
```

```
#status is the $\delta_{ij}
table(data_np$status)/sum(table(data_np$status))
```

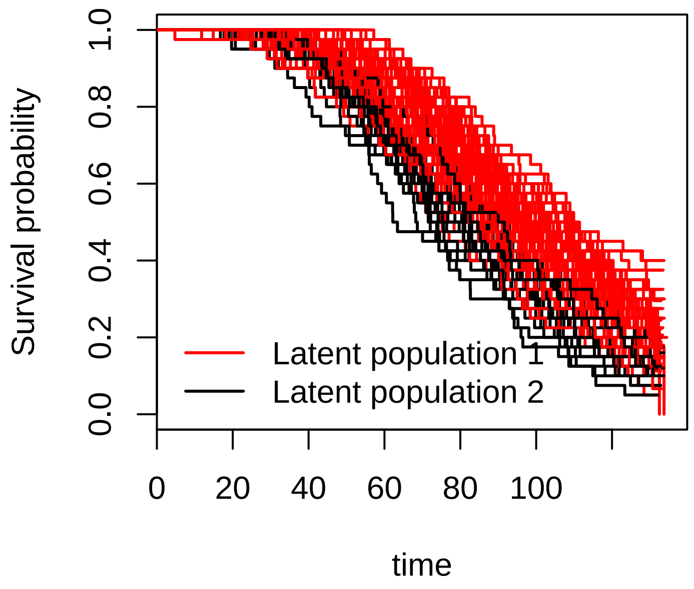
```
##
##      0      1
## 0.1985 0.8015
```

```
#belong is the w
table(data_np$belong)/sum(table(data_np$belong))
```

```
##
## 0.8 1.6
## 0.8 0.2
```

We can visualize the data_np.

```
fit1 <- coxph( Surv( time, status ) ~ family, data_np, method = 'breslow')
sfit1 <- survfit(Surv( time, status ) ~ family, data_np)

#setting the colors according to 'belong'
lat_pop = rep( 2, J )
lat_pop[ which( data_np$belong[ seq( 1, dim(data_np)[1], N ) ] %in%
                  w_values[2]) ] = 1

plot( sfit1, col = lat_pop, xlab = 'time', ylab = 'Survival probability',
      lwd = 1.5 )
legend( 'bottomleft', paste('Latent population', 1:2), col = 2:1,
        lty = rep(1,2), lwd = 1.5, bty = 'n' )
```



Cox model with a nonparametric discrete frailty

In `npdf_cox` function, the method for estimating a Cox model with a nonparametric discrete frailty is implemented. See Section 3 of Gasperoni *et al.* (2018).

We now apply `npdf_cox` on the previously mentioned `data_weib`, starting from $K = 4$ latent popuations.

```r
test_weib <- npdf_cox( Surv(time, status) ~ x.1 + x.2, groups = family,
                       data = data_weib, K = 4, eps_conv=10^-4)
test_weib
```

```
##
## Call:
## npdf_cox(formula = Surv(time, status) ~ x.1 + x.2, groups = family,
##     data = data_weib, K = 4, eps_conv = 10^-4)
##
## Model comparison:
##   K K_fitted      llik      AIC      BIC
## 1 1        1 -28552.01 57110.02 57128.59
## 2 2        2 -28503.72 57017.45 57048.39
## 3 3        3 -28512.46 57038.93 57082.24
## 4 4        4 -28517.39 57052.79 57108.48
## Optimal K:
## Laird   AIC   BIC
##     4     2     2
##
## Best model according to BIC:
## Nonparametric discrete frailty Cox model fit with K=2 latent populations
##
## Estimated parameters and standard errors:
##          est seLouis seExact
## p1    0.706  0.0525  0.0515
## p2    0.294  0.0525  0.0515
## w2/w1 1.755  0.0702  0.0697
## x.1   1.665  0.0273  0.0273
## x.2   0.414  0.0175  0.0175
##
```

```
## Log-likelihood: -28504
## AIC: 57017
## BIC: 57048
## Fitted K: 2
##
## To examine other models, look at `models` component
```

According to AIC and BIC, the optimal $\hat{K}$ is equal to 2, which is correct. Laird criterion (Laird 1978) states that the optimal $\hat{K}$ is 4. The estimates $\hat{\boldsymbol{\pi}}$ are $[0.665, 0.335]$, while the real ones are $[0.7, 0.3]$. The estimated ratio $\hat{w}_2/\hat{w}_1 = 1.747$, while the real one is 1.75 (2.1/1.2). Exact standard errors and standard errors according to Louis method (Louis 1982) are also reported (see Supplementary Material of Gasperoni et al.(2018) for more detailed insights of the stanadrd errors computation). If `Fitted K` is different from the number of latent populations of the related iteration (i.e., `Fitted K: 3` and `K = 4`, means that there is a latent population to which no group is assigned), the standard errors are not computed.

We focus on the output of `npdf_cox`:

- `npdf_cox$model` is a list composed by 4 elements. The i^{th} element of the list is composed by the estimates related to $K = i$ latent populations (we show the output related to $K = 3$).

```
test_weib$model[[3]]
```

```
##
## Nonparametric discrete frailty Cox model fit with K=3 latent populations
##
## Estimated parameters and standard errors:
##          est seLouis seExact
## p1    0.0394  0.0635  0.0635
## p2    0.6678  0.0751  0.0757
## p3    0.2928  0.0537  0.0553
## w2/w1 1.3710  0.2557  0.2565
## w3/w1 2.3585  0.4726  0.4751
## x.1   1.6697  0.0276  0.0276
## x.2   0.4133  0.0176  0.0176
##
## Log-likelihood: -28512
## AIC: 57039
## BIC: 57082
## Fitted K: 3
```

- `test_weib$comparison` is a matrix in which the values of log-likelihhod, AIC and BIC are reported for each value of K.

```
test_weib$comparison
```

```
##   K K_fitted      llik      AIC      BIC
## 1 1        1 -28552.01 57110.02 57128.59
## 2 2        2 -28503.72 57017.45 57048.39
## 3 3        3 -28512.46 57038.93 57082.24
## 4 4        4 -28517.39 57052.79 57108.48
```

- `test_weib$Kopt` report the optimal K according to Laird, AIC and BIC.

```
test_weib$Kopt
```

```
## Laird   AIC   BIC
##     4     2     2
```

- `test_weib$criterion` is the creterion according to which the optimal K is chosen. The default option is BIC, but the user can choose between AIC, BIC and Laird.

```
test_weib$criterion
```

```
## [1] "BIC"
```

- `test_weib$mf` is the representation of the dataset (we do not report it for the sake of space).

Since the data are simulated, we are also able to check whether the groups are correctly assigned. We note that only 3 groups are misclassified (1, 42 and 98).

```
#real_lat_pop: vector Jx1 of real latent populations index.
real_lat_pop = rep( 1, length( unique( data_weib$family ) ) )

w_values = sort( unique( data_weib$belong ) )

#group indices in latent population 1
ind_w_1 = data_weib$belong[seq(1, dim(data_weib)[1],
                           length(which(data_weib$family == 1))) ] %in%
                            w_values[ 1 ]
#group indices in latent population 2
ind_w_2 = data_weib$belong[seq(1, dim(data_weib)[1],
                           length(which(data_weib$family == 1))) ] %in%
                            w_values[ 2 ]

real_lat_pop[ ind_w_1 ] = 1
real_lat_pop[ ind_w_2 ] = 2

#match of estimated and real latent populations
table( test_weib$models[[2]]$belonging, real_lat_pop )
```

```
##    real_lat_pop
##       1  2
##   1 69  2
##   2  1 28
```

```
which( test_weib$models[[2]]$belonging != real_lat_pop )
```

```
## [1]  1 42 98
```

We can investigate the posterior probabilities of being part of a specific latent population

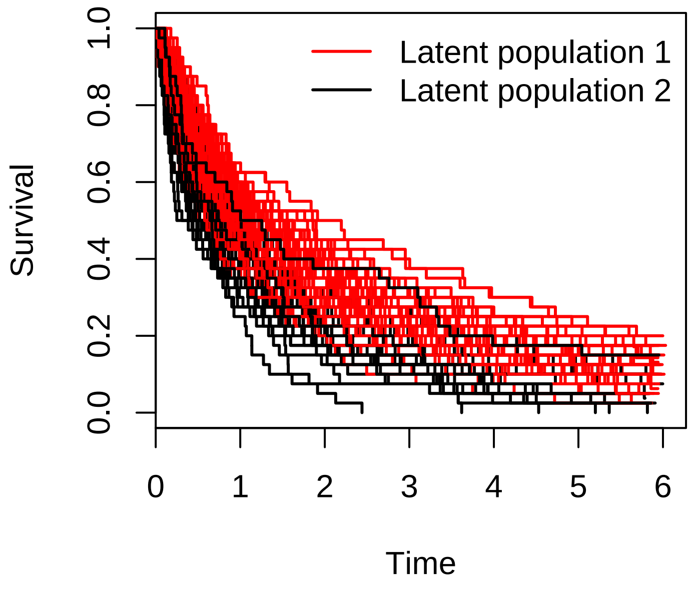
(α_{jk}, see Section 3.1 of Gasperoni *et al.* (2018)). It is possible to note that few groups are not definitely assigned to a specific latent population (i.e., group 1 is assigned to latent population 1 with a probability of 0.59 and is assigned to latent population 2 with a probability of 0.41). It is also interesting to see that all the groups that are misclassified are part of the vector `assign_not_sure1`.

```r
#alpha matrix: alpha_{jk}
#is the probability that group j is assigned to latent population k.
#test_weib$models[[2]]$alpha

assign_not_sure1 = which( test_weib$models[[2]]$alpha[ ,1 ] > 0.15 &
                          test_weib$models[[2]]$alpha[ ,1 ] < 0.85 )

assign_not_sure1
```

```
##  [1]   1   8  10  42  49  56  63  79  96  98 100
```

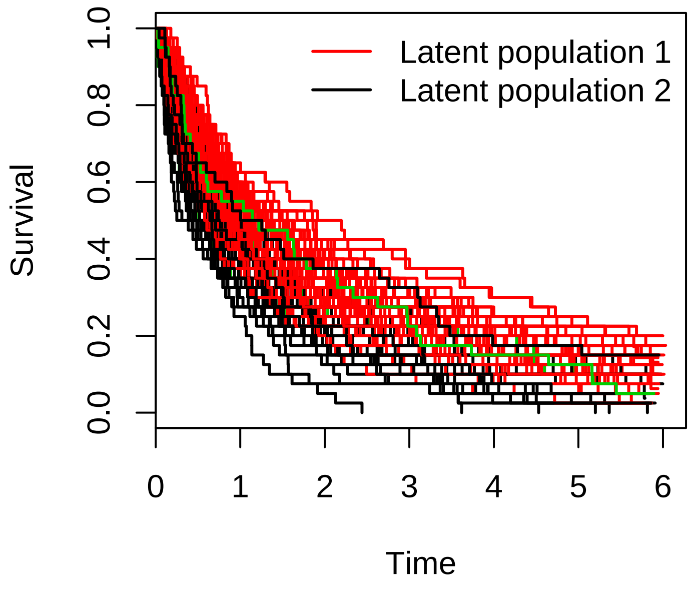
We can visualize the results through the `plot` function. It is also possible to plot Nelson-Aalen estimates (choosing `type = 'na'`).

```r
#Kaplan-Meyer estimate
plot( test_weib, type = 'km', lwd = 1.5  )
legend( 'topright', paste( 'Latent population', c( 1, 2 ) ), lwd = 1.5,
        col = c( 2, 1 ), lty = rep( 1, 2 ), bty = 'n' )
```



We can highlight those groups that are misclassified in the plot. The lines in green are the one estimated for misclassified groups.

```r
#Kaplan-Meyer estimate
colors_misc = test_weib$models[[2]]$belonging + 1
#higher frailty -> black
colors_misc[ which( colors_misc == 3 ) ] = 1
#misclassified group -> green
colors_misc[ which( test_weib$models[[2]]$belonging != real_lat_pop ) ] = 3
plot( test_weib, type = 'km', col = colors_misc, lwd = 1.5  )
legend( 'topright', paste( 'Latent population', c( 1, 2 ) ), lwd = 1.5,
        col = c( 2, 1 ), lty = rep( 1, 2 ), bty = 'n' )
```



References

Bender R, Augustin T, Blettner M. **2005**. Generating survival times to simulate cox proportional hazards models. *Statistics in medicine* **24**: 1713–1723.

Gasperoni F, Ieva F, Paganoni A, Jackson C, Sharples L. **2018**. Nonparametric frailty cox models for hierarchical time-to-event data. *Biostatistics, to appear.*

Laird N. **1978**. Nonparametric maximum likelihood estimation of a mixing distribution. *Journal of the American Statistical Association* **73**: 805–811.

Louis TA. **1982**. Finding the observed information matrix when using the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*: 226–233.