**BIDA in Stan: Piecewise constant hazard Cox**

***Kazuki Yoshida***

***2018-11-15***

**https://rpubs.com/kaz\_yos/surv\_stan\_piecewise1**

**References**

* Online
  + [Stan Leukemia example](https://github.com/stan-dev/example-models/blob/master/bugs_examples/vol1/leuk/leuk.stan)
  + [Stan for survival models](https://discourse.mc-stan.org/t/stan-for-survival-models/4146)
  + [PyMC3 Bayesian Survival Analysis](https://docs.pymc.io/notebooks/survival_analysis.html)
  + [Piece-Wise Exponential Model](http://data.princeton.edu/wws509/notes/c7s4.html)
  + [Stan-dev Prior Choice Recommendations](https://github.com/stan-dev/stan/wiki/Prior-Choice-Recommendations)
* Books
  + [(BIDA) Bayesian Ideas and Data Analysis An Introduction for Scientists and Statisticians](http://blogs.oregonstate.edu/bida/)
  + [(BUGS) The BUGS Book: A Practical Introduction to Bayesian Analysis](https://www.mrc-bsu.cam.ac.uk/software/bugs/the-bugs-project-the-bugs-book/)
  + [(BSA) Bayesian Survival Analysis](https://www.springer.com/us/book/9780387952772)

**Background**

The proportional hazards (PH) model (Cox 1972) is a very popular regression method for survival data. The popularity likely originated in not having to specify the (baseline) hazard function. These days, it is popular probably because it is the only survival regression model that many applied researchers are aware of. In Frequentist PH model, parameter estimation is conducted via partial likelihood from which the baseline hazard function has dropped out (BSA p16).

In Bayesian paradigm, obtaining the posterior distribution of parameters requires the full likelihood function involving all parameters including nuisance ones and the prior for all these parameters. In the case of survival analysis, one of the parameters that requires modeling is the entire baseline hazard function. One way to proceed is to parametrize the baseline hazard function parsimoniously, i.e., parametric survival analysis (BIDA p325, BSA chap 2). The other approach is to more flexibly model the baseline hazard function.

Here we will examine the simplest form of the latter, the piecewise constant hazard model (piecewise exponential model).

**Piecewise constant hazard model**

**Likelihood**

This model formulation was taken from BSA (p47-).

Firstly, partition the time axis into *J*

intervals using 0<*s*1<*s*2...<*sJ*<∞

.

(0,*s*1],(*s*1,*s*2],...,(*sJ*−1,*sJ*]

*sJ*

is a finite value that has to be larger than the largest observed time in the study. Name each interval *Ij*=(*sj*−1,*sj*]. We assume a constant hazard *λ* within each interval. Let *D*=(*n*,**y**,*X*,*ν*)

describe the observed data.

**y**=(*y*1,...,*yn*)*T*

is the observed times.

*X*

is a *n*×*p* matrix of covariates associated with a length *p* vector ***β***

.

***ν***=(*ν*1,...,*νn*)*T*

is a vector of failure (censoring) indicators.

Let ***λ***=(*λ*1,...,*λJ*)*T*

. The full likelihood for (***β***,***λ***)

is the following.

*L*(***β***,***λ***|*D*)=∏*i*=1*n*∏*j*=1*J*(*λj*exp(**x***Ti****β***))*δijνi*exp{−*δij*[*λj*(*yi*−*sj*−1)+∑*g*=1*j*−1*λg*(*sg*−*sg*−1)]exp(**x***Ti****β***)}

where *δij*

is an interval specific indicator of end of follow up indicator *I*[*yi*∈*Ij*]

.

This is really hard, so let us dissect this into more manageable pieces. Firstly, we will focus on one individual rather than the entire dataset.

*L*(***β***,***λ***|*Di*)=∏*j*=1*J*(*λj*exp(**x***Ti****β***))*δijνi*exp{−*δij*[*λj*(*yi*−*sj*−1)+∑*g*=1*j*−1*λg*(*sg*−*sg*−1)]exp(**x***Ti****β***)}

Thus, this is a product of interval-specific contributions. However, for interval *Ij*

in which the individual just survived without death or censoring *δij*=0

, there is no contribution (no parameters left).

(*λj*exp(**x***Ti****β***))(0)*νi*exp{−(0)[*λj*(*yi*−*sj*−1)+∑*g*=1*j*−1*λg*(*sg*−*sg*−1)]exp(**x***Ti****β***)}=1

Therefore, the only contribution happens at the interval when the individual either dies or becomes censored. Note in general, an event individual contributes a density, which is a product of hazard and survival. On the other hand, a censored individual can only contribute survival information.

**Event individual contribution**

Firstly, we will consider death in interval *Ij*

.

(*λj*exp(**x***Ti****β***))(1)(1)exp{−(1)[*λj*(*yi*−*sj*−1)+∑*g*=1*j*−1*λg*(*sg*−*sg*−1)]exp(**x***Ti****β***)}

(*λj*exp(**x***Ti****β***))(1)(1)

is the hazard contribution.

exp{−(1)[*λj*(*yi*−*sj*−1)+∑*j*−1*g*=1*λg*(*sg*−*sg*−1)]exp(**x***Ti****β***)}

is the survival contribution. Taken together this individual contributes density information.

Note [*λj*(*yi*−*sj*−1)+∑*j*−1*g*=1*λg*(*sg*−*sg*−1)]

is the cumulative baseline hazard that this individual faced. By multiplying this with time-constant (by the PH assumption) multiplication factor, exp(**x***Ti****β***)

, we obtain the individual-specific cumulative hazard. An exponential of the negative cumulative hazard is the survival.

**Censored individual contribution**

Now we will consider censoring in interval *Ij*

.

(*λj*exp(**x***Ti****β***))(1)(0)exp{−(1)[*λj*(*yi*−*sj*−1)+∑*g*=1*j*−1*λg*(*sg*−*sg*−1)]exp(**x***Ti****β***)}

(*λj*exp(**x***Ti****β***))(1)(0)=1

. There is no hazard contribution.

exp{−(1)[*λj*(*yi*−*sj*−1)+∑*j*−1*g*=1*λg*(*sg*−*sg*−1)]exp(**x***Ti****β***)}

is the survival contribution.

**Poisson trick**

This part follows BIDA (p347-) and [Piece-Wise Exponential Model](http://data.princeton.edu/wws509/notes/c7s4.html). Let us examine the likelihood for individual *i*

further. As stated above only contribution happens at the interval in which follow up ends by death or censoring. Without loss of generality, consider this interval as *Ij*

.

*L*(***β***,***λ***|*Di*)=(*λj*exp(**x***Ti****β***))*δijνi*exp{−*δij*[*λj*(*yi*−*sj*−1)+∑*g*=1*j*−1*λg*(*sg*−*sg*−1)]exp(**x***Ti****β***)}   By *δij*=1=(*λj*exp(**x***Ti****β***))*νi*exp{−[*λj*(*yi*−*sj*−1)+∑*g*=1*j*−1*λg*(*sg*−*sg*−1)]exp(**x***Ti****β***)}   Let *Hi*,*g* represent at-risk time during *Ig* for *i*.=(*λj*exp(**x***Ti****β***))*νi*exp{−[∑*g*=1*jλgHi*,*g*]exp(**x***Ti****β***)}=(*λj*exp(**x***Ti****β***))*νi*exp{−∑*g*=1*jλgHi*,*g*exp(**x***Ti****β***)}=(*λj*exp(**x***Ti****β***))*νi*∏*g*=1*j*exp{−*λgHi*,*g*exp(**x***Ti****β***)}   Note the new middle piece is 1.=(*λj*exp(**x***Ti****β***))*νi*[∏*g*=1*j*−1(*λgHi*,*g*exp(**x***Ti****β***))0]∏*g*=1*j*exp{−*λgHi*,*g*exp(**x***Ti****β***)}∝(*λjHi*,*j*exp(**x***Ti****β***))*νi*[∏*g*=1*j*−1(*λgHi*,*g*exp(**x***Ti****β***))0]∏*g*=1*j*exp{−*λgHi*,*g*exp(**x***Ti****β***)}   By reintroducing *δi*,*g*, which is 0 for *g*<*j*=[∏*g*=1*j*(*λgHi*,*g*exp(**x***Ti****β***))*δi*,*gνi*]∏*g*=1*j*exp{−*λgHi*,*g*exp(**x***Ti****β***)}=∏*g*=1*j*exp{−*λgHi*,*g*exp(**x***Ti****β***)}(*λgHi*,*g*exp(**x***Ti****β***))*δi*,*gνi*   By *δi*,*gνi*∈{0,1}=∏*g*=1*j*exp{−*λgHi*,*g*exp(**x***Ti****β***)}(*λgHi*,*g*exp(**x***Ti****β***))*δi*,*gνi*/(*δi*,*gνi*)!

We can multiply the likelihood with a term that does not contain parameters and retain the same inference. Note the last expression is a product of individual- and interval-specific Poisson likelihood. This transformation implies that we can split each individual’s observation into interval-specific observations up until the interval in which follow up ended. In addition to copying covariates, each interval-specific observation has to have the duration of the at-risk time and an indicator that is 1 in the last interval if the individual died otherwise 0. The latter indicator serves as the outcome of the Poisson model.

The corresponding Poisson model for individual *i*

interval *g*

is the following.

*μi*,*g*log(*μi*,*g*)=*λgHi*,*g*exp(**x***Ti****β***)=log(*λgHi*,*g*exp(**x***Ti****β***))=log(*Hi*,*g*)+log(*λg*)+**x***Ti****β***=log(*Hi*,*g*)+*β*0,*g*+**x***Ti****β***

Therefore, log(*Hi*,*g*)

becomes the offset. The intercept *β*0,*g*=log(*λg*) is interval-specific. The outcome, the indicator variable (*δi*,*gνi*)

is not independent within an individual because it is all zero except the last one, which can be 1 if the individual died. However, the likelihood has the form of a product of interval-specific Poisson contributions. Thus, the Poisson modeling can proceed as if these interval-specific observations from one individual were independent. In R, [survival::survSplit](https://stat.ethz.ch/R-manual/R-devel/library/survival/html/survSplit.html) function can create this long-format dataset from a single-row-per-person dataset.

**Prior specification**

We have clarified the likelihood part, so now we need to specify the priors for all parameters. We have the covariate coefficient vector ***β***

. After the Poisson transformation, we also have the interval-specific intercepts. Each one of the parameters can take on any value on the real line.

The Stan developer website has a web page dedicated to prior choice philosophy [(Prior Choice Recommendations)](https://github.com/stan-dev/stan/wiki/Prior-Choice-Recommendations). Here we adopt the principle of weakly informative priors that are informative enough to regularize. The direct quote is the following.

Weakly informative prior should contain enough information to regularize: the idea is that the prior rules out unreasonable parameter values but is not so strong as to rule out values that might make sense.

**Priors for covariate coefficients**

These coefficients for covariates are on the log hazard ratio scale. Based on the substantive ground, we would like to rule out hazard ratios that are greater than 50 or smaller than 1/50. log(50)=3.912023

. Thus, *N*(0,22)

may be a good choice. This prior puts approximately 5% of the prior probability located outside the above stated reasonable range.

**Priors for baseline hazards**

BSA (p48) suggests independent gamma priors for the piecewise baseline hazard parameters and multivariate normal prior for the vector of log baseline hazard parameters. BSA does not mention numerical values. It is hard to think of the reasonable range for the piecewise baseline hazards.

BUGS (p290) uses independent *Gamma*(0.001,0.001)

(dgamma) as a prior for each *λ*

. This has mean 1 and variance 1000.

BIDA (p351-) acknowledges this difficulty and suggest centering the priors on an exponential regression model.

*λk*∼*Gamma*(*λ*∗*wk*,*wk*)

where *wk*

is the interval length times some hyperparameter *w*

.

**Data analysis example**

**Set up multicore environment**

## Configure parallelization

## Parallel backend for foreach (also loads foreach and parallel; includes doMC)

library(doParallel)

## Reproducible parallelization

library(doRNG)

## Detect core count (Do not use on clusters)

n\_cores <- parallel::detectCores()

## Used by parallel::mclapply() as default

options(mc.cores = n\_cores)

## Used by doParallel as default

options(cores = n\_cores)

## Register doParallel as the parallel backend with foreach

## http://stackoverflow.com/questions/28989855/the-difference-between-domc-and-doparallel-in-r

doParallel::registerDoParallel(cores = n\_cores)

## Report multicore use

## cat("### Using", foreach::getDoParWorkers(), "cores\n")

## cat("### Using", foreach::getDoParName(), "as backend\n")

**Load packages**

library(tidyverse)

library(survival)

library(rstanarm)

library(broom)

library(directlabels)

**Load and prepare dataset**

aml package:survival R Documentation

Acute Myelogenous Leukemia survival data

Description:

Survival in patients with Acute Myelogenous Leukemia. The

question at the time was whether the standard course of

chemotherapy should be extended ('maintainance') for additional

cycles.

Usage:

aml

leukemia

Format:

time: survival or censoring time

status: censoring status

x: maintenance chemotherapy given? (factor)

Source:

Rupert G. Miller (1997), \_Survival Analysis\_. John Wiley & Sons.

ISBN: 0-471-25218-2.

data(leukemia, package = "survival")

leukemia <- as\_data\_frame(leukemia) %>%

mutate(id = seq\_len(n())) %>%

select(id, everything())

leukemia

## # A tibble: 23 x 4

## id time status x

## <int> <dbl> <dbl> <fct>

## 1 1 9 1 Maintained

## 2 2 13 1 Maintained

## 3 3 13 0 Maintained

## 4 4 18 1 Maintained

## 5 5 23 1 Maintained

## 6 6 28 0 Maintained

## 7 7 31 1 Maintained

## 8 8 34 1 Maintained

## 9 9 45 0 Maintained

## 10 10 48 1 Maintained

## # ... with 13 more rows

Check distribution of observed times and decide cut points for piecewise constant hazard function.

## One piece

cut\_one <- max(leukemia$time) + 1

cut\_one

## [1] 162

## Two pieces

cut\_two <- c(median(leukemia$time), cut\_one) %>% round()

cut\_two

## [1] 23 162

## Three pieces

cut\_three <- c(quantile(leukemia$time, probs = c(1/3, 2/3)), cut\_one) %>% round()

cut\_three

## 33.33333% 66.66667%

## 14 31 162

## At all event times

cut\_events <- c(leukemia %>%

filter(status == 1) %>%

select(time) %>%

arrange(time) %>%

magrittr::extract2("time") %>%

unique,

cut\_one) %>% round()

cut\_events

## [1] 5 8 9 12 13 18 23 27 30 31 33 34 43 45 48 162

## At all event or censoring

cut\_times <- c(sort(unique(leukemia$time)),

cut\_one) %>% round()

cut\_times

## [1] 5 8 9 12 13 16 18 23 27 28 30 31 33 34 43 45 48 161 162

Now transform dataset into long-format ones. We need an interval indicator and an interval length variable.

## No cut for all same constant hazard

leukemia\_one <- survival::survSplit(formula = Surv(time, status) ~ ., data = leukemia, cut = cut\_one) %>%

mutate(interval = factor(tstart),

interval\_length = time - tstart) %>%

as\_data\_frame

leukemia\_one

## # A tibble: 23 x 7

## id x tstart time status interval interval\_length

## <int> <fct> <dbl> <dbl> <dbl> <fct> <dbl>

## 1 1 Maintained 0 9 1 0 9

## 2 2 Maintained 0 13 1 0 13

## 3 3 Maintained 0 13 0 0 13

## 4 4 Maintained 0 18 1 0 18

## 5 5 Maintained 0 23 1 0 23

## 6 6 Maintained 0 28 0 0 28

## 7 7 Maintained 0 31 1 0 31

## 8 8 Maintained 0 34 1 0 34

## 9 9 Maintained 0 45 0 0 45

## 10 10 Maintained 0 48 1 0 48

## # ... with 13 more rows

## Split into two observations

leukemia\_two <- survival::survSplit(formula = Surv(time, status) ~ ., data = leukemia, cut = cut\_two) %>%

mutate(interval = factor(tstart),

interval\_length = time - tstart) %>%

as\_data\_frame

leukemia\_two

## # A tibble: 34 x 7

## id x tstart time status interval interval\_length

## <int> <fct> <dbl> <dbl> <dbl> <fct> <dbl>

## 1 1 Maintained 0 9 1 0 9

## 2 2 Maintained 0 13 1 0 13

## 3 3 Maintained 0 13 0 0 13

## 4 4 Maintained 0 18 1 0 18

## 5 5 Maintained 0 23 1 0 23

## 6 6 Maintained 0 23 0 0 23

## 7 6 Maintained 23 28 0 23 5

## 8 7 Maintained 0 23 0 0 23

## 9 7 Maintained 23 31 1 23 8

## 10 8 Maintained 0 23 0 0 23

## # ... with 24 more rows

## Split into three observations

leukemia\_three <- survival::survSplit(formula = Surv(time, status) ~ ., data = leukemia, cut = cut\_three) %>%

mutate(interval = factor(tstart),

interval\_length = time - tstart) %>%

as\_data\_frame

leukemia\_three

## # A tibble: 45 x 7

## id x tstart time status interval interval\_length

## <int> <fct> <dbl> <dbl> <dbl> <fct> <dbl>

## 1 1 Maintained 0 9 1 0 9

## 2 2 Maintained 0 13 1 0 13

## 3 3 Maintained 0 13 0 0 13

## 4 4 Maintained 0 14 0 0 14

## 5 4 Maintained 14 18 1 14 4

## 6 5 Maintained 0 14 0 0 14

## 7 5 Maintained 14 23 1 14 9

## 8 6 Maintained 0 14 0 0 14

## 9 6 Maintained 14 28 0 14 14

## 10 7 Maintained 0 14 0 0 14

## # ... with 35 more rows

## Split at event times

leukemia\_events <- survival::survSplit(formula = Surv(time, status) ~ ., data = leukemia, cut = cut\_events) %>%

mutate(interval = factor(tstart),

interval\_length = time - tstart) %>%

as\_data\_frame

leukemia\_events

## # A tibble: 180 x 7

## id x tstart time status interval interval\_length

## <int> <fct> <dbl> <dbl> <dbl> <fct> <dbl>

## 1 1 Maintained 0 5 0 0 5

## 2 1 Maintained 5 8 0 5 3

## 3 1 Maintained 8 9 1 8 1

## 4 2 Maintained 0 5 0 0 5

## 5 2 Maintained 5 8 0 5 3

## 6 2 Maintained 8 9 0 8 1

## 7 2 Maintained 9 12 0 9 3

## 8 2 Maintained 12 13 1 12 1

## 9 3 Maintained 0 5 0 0 5

## 10 3 Maintained 5 8 0 5 3

## # ... with 170 more rows

## Split at event and censoring times

leukemia\_times <- survival::survSplit(formula = Surv(time, status) ~ ., data = leukemia, cut = cut\_times) %>%

mutate(interval = factor(tstart),

interval\_length = time - tstart) %>%

as\_data\_frame

leukemia\_times

## # A tibble: 203 x 7

## id x tstart time status interval interval\_length

## <int> <fct> <dbl> <dbl> <dbl> <fct> <dbl>

## 1 1 Maintained 0 5 0 0 5

## 2 1 Maintained 5 8 0 5 3

## 3 1 Maintained 8 9 1 8 1

## 4 2 Maintained 0 5 0 0 5

## 5 2 Maintained 5 8 0 5 3

## 6 2 Maintained 8 9 0 8 1

## 7 2 Maintained 9 12 0 9 3

## 8 2 Maintained 12 13 1 12 1

## 9 3 Maintained 0 5 0 0 5

## 10 3 Maintained 5 8 0 5 3

## # ... with 193 more rows

**Sanity check with frequentist methods**

coxph(formula = Surv(time, status) ~ x,

data = leukemia,

ties = c("efron","breslow","exact")[1]) %>% summary

## Call:

## coxph(formula = Surv(time, status) ~ x, data = leukemia, ties = c("efron",

## "breslow", "exact")[1])

##

## n= 23, number of events= 18

##

## coef exp(coef) se(coef) z Pr(>|z|)

## xNonmaintained 0.9155 2.4981 0.5119 1.788 0.0737 .

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## exp(coef) exp(-coef) lower .95 upper .95

## xNonmaintained 2.498 0.4003 0.9159 6.813

##

## Concordance= 0.619 (se = 0.063 )

## Rsquare= 0.137 (max possible= 0.976 )

## Likelihood ratio test= 3.38 on 1 df, p=0.07

## Wald test = 3.2 on 1 df, p=0.07

## Score (logrank) test = 3.42 on 1 df, p=0.06

glm(formula = status ~ x + offset(log(interval\_length)),

family = poisson(link = "log"),

data = leukemia\_one) %>% summary

##

## Call:

## glm(formula = status ~ x + offset(log(interval\_length)), family = poisson(link = "log"),

## data = leukemia\_one)

##

## Deviance Residuals:

## Min 1Q Median 3Q Max

## -2.3084 -0.6723 0.2218 0.9045 1.4513

##

## Coefficients:

## Estimate Std. Error z value Pr(>|z|)

## (Intercept) -4.1015 0.3780 -10.851 <2e-16 \*\*\*

## xNonmaintained 0.9581 0.4835 1.982 0.0475 \*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## (Dispersion parameter for poisson family taken to be 1)

##

## Null deviance: 25.878 on 22 degrees of freedom

## Residual deviance: 21.816 on 21 degrees of freedom

## AIC: 61.816

##

## Number of Fisher Scoring iterations: 6

## Drop intercept to show all interval-specific log rates

glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

data = leukemia\_two,

family = poisson(link = "log")) %>% summary

##

## Call:

## glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

## family = poisson(link = "log"), data = leukemia\_two)

##

## Deviance Residuals:

## Min 1Q Median 3Q Max

## -2.3680 -0.7694 -0.2321 1.0099 1.5947

##

## Coefficients:

## Estimate Std. Error z value Pr(>|z|)

## interval0 -4.3530 0.4813 -9.044 <2e-16 \*\*\*

## interval23 -3.8963 0.4245 -9.178 <2e-16 \*\*\*

## xNonmaintained 1.0760 0.5008 2.149 0.0317 \*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## (Dispersion parameter for poisson family taken to be 1)

##

## Null deviance: 1233.592 on 34 degrees of freedom

## Residual deviance: 39.322 on 31 degrees of freedom

## AIC: 81.322

##

## Number of Fisher Scoring iterations: 6

glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

data = leukemia\_three,

family = poisson(link = "log")) %>% summary

##

## Call:

## glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

## family = poisson(link = "log"), data = leukemia\_three)

##

## Deviance Residuals:

## Min 1Q Median 3Q Max

## -2.2992 -0.7545 -0.5998 0.9033 1.9277

##

## Coefficients:

## Estimate Std. Error z value Pr(>|z|)

## interval0 -4.3546 0.5342 -8.151 3.60e-16 \*\*\*

## interval14 -4.1437 0.5469 -7.577 3.55e-14 \*\*\*

## interval31 -3.8955 0.4792 -8.129 4.32e-16 \*\*\*

## xNonmaintained 1.0734 0.5170 2.076 0.0379 \*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## (Dispersion parameter for poisson family taken to be 1)

##

## Null deviance: 1243.349 on 45 degrees of freedom

## Residual deviance: 49.378 on 41 degrees of freedom

## AIC: 93.378

##

## Number of Fisher Scoring iterations: 6

glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

data = leukemia\_events,

family = poisson(link = "log")) %>% summary

##

## Call:

## glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

## family = poisson(link = "log"), data = leukemia\_events)

##

## Deviance Residuals:

## Min 1Q Median 3Q Max

## -1.0000 -0.4782 -0.3817 -0.2709 2.2170

##

## Coefficients:

## Estimate Std. Error z value Pr(>|z|)

## interval0 -4.6197 0.7997 -5.777 0.0000000076 \*\*\*

## interval5 -3.9794 0.7909 -5.032 0.0000004863 \*\*\*

## interval8 -3.4251 1.0527 -3.254 0.001139 \*\*

## interval9 -4.4906 1.0562 -4.252 0.0000212196 \*\*\*

## interval12 -3.3054 1.0513 -3.144 0.001666 \*\*

## interval13 -4.7988 1.0573 -4.539 0.0000056629 \*\*\*

## interval18 -3.9979 0.7880 -5.074 0.0000003904 \*\*\*

## interval23 -4.2947 1.0577 -4.060 0.0000489812 \*\*\*

## interval27 -3.8196 1.0538 -3.625 0.000289 \*\*\*

## interval30 -2.5174 1.0457 -2.407 0.016063 \*

## interval31 -3.1265 1.0538 -2.967 0.003009 \*\*

## interval33 -2.1895 1.0392 -2.107 0.035127 \*

## interval34 -4.2680 1.0495 -4.067 0.0000476765 \*\*\*

## interval43 -2.3916 1.0264 -2.330 0.019795 \*

## interval45 -1.7918 1.0000 -1.792 0.073172 .

## interval48 -21.0300 2103.3625 -0.010 0.992023

## xNonmaintained 0.9024 0.5123 1.762 0.078133 .

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## (Dispersion parameter for poisson family taken to be 1)

##

## Null deviance: 1282.980 on 180 degrees of freedom

## Residual deviance: 74.353 on 163 degrees of freedom

## AIC: 144.35

##

## Number of Fisher Scoring iterations: 14

glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

data = leukemia\_times,

family = poisson(link = "log")) %>% summary

##

## Call:

## glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

## family = poisson(link = "log"), data = leukemia\_times)

##

## Deviance Residuals:

## Min 1Q Median 3Q Max

## -1.0000 -0.4653 -0.3347 -0.2550 2.2175

##

## Coefficients:

## Estimate Std. Error z value Pr(>|z|)

## interval0 -4.6210 0.7997 -5.778 0.00000000755 \*\*\*

## interval5 -3.9806 0.7910 -5.033 0.00000048368 \*\*\*

## interval8 -3.4262 1.0528 -3.255 0.001136 \*\*

## interval9 -4.4918 1.0563 -4.253 0.00002113636 \*\*\*

## interval12 -3.3065 1.0513 -3.145 0.001661 \*\*

## interval13 -20.8476 2343.2634 -0.009 0.992901

## interval16 -3.8208 1.0539 -3.625 0.000288 \*\*\*

## interval18 -3.9991 0.7881 -5.075 0.00000038829 \*\*\*

## interval23 -4.2959 1.0578 -4.061 0.00004879153 \*\*\*

## interval27 -19.6892 2869.9764 -0.007 0.994526

## interval28 -3.3932 1.0563 -3.212 0.001316 \*\*

## interval30 -2.5185 1.0457 -2.408 0.016024 \*

## interval31 -3.1276 1.0539 -2.968 0.003000 \*\*

## interval33 -2.1905 1.0393 -2.108 0.035055 \*

## interval34 -4.2691 1.0496 -4.068 0.00004750821 \*\*\*

## interval43 -2.3924 1.0264 -2.331 0.019759 \*

## interval45 -1.7918 1.0000 -1.792 0.073172 .

## interval48 -24.0300 9426.6168 -0.003 0.997966

## xNonmaintained 0.9042 0.5122 1.765 0.077530 .

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## (Dispersion parameter for poisson family taken to be 1)

##

## Null deviance: 1285.623 on 203 degrees of freedom

## Residual deviance: 74.182 on 184 degrees of freedom

## AIC: 148.18

##

## Number of Fisher Scoring iterations: 17

Some log baseline hazard estimates are -22.8454 with a standard error of 8693.5021. This non-convergence of the MLE algorithm occurs if the interval (a,b] does not contain any event, such as (13,16] in the model that defined intervals by all observed times. There is only one censoring at time 16 in this interval.

**Model with rstanarm**

Here we will use normal priors because they are readily available in rstanarm. Centering the prior for a log hazard ratio parameter at 0 (hazard ratio 1) sounds reasonable.

Where to center the prior for a log baseline hazard is unclear, so we will center it around 0, but make it vague. Here we will use *N*(0,42)

, which means 95% of the prior probability is in [-7.839856, +7.839856] on the natural log scale.

We will remove the intercept term to make all interval terms parametrized as log baseline hazard (rather than change from the initial log baseline hazard). rstanarm does not seem to allow a different prior for each regression coefficient. So we will just use *N*(0,42)

for all.

fit\_leukemia\_one <- rstanarm::stan\_glm(formula = status ~ x + offset(log(interval\_length)),

data = leukemia\_one,

family = poisson(link = "log"),

prior\_intercept = normal(location = 0, scale = 4),

prior = normal(location = 0, scale = 4))

prior\_summary(fit\_leukemia\_one)

## Priors for model 'fit\_leukemia\_one'

## ------

## Intercept (after predictors centered)

## ~ normal(location = 0, scale = 4)

##

## Coefficients

## ~ normal(location = 0, scale = 4)

## ------

## See help('prior\_summary.stanreg') for more details

summary(fit\_leukemia\_one)

##

## Model Info:

##

## function: stan\_glm

## family: poisson [log]

## formula: status ~ x + offset(log(interval\_length))

## algorithm: sampling

## priors: see help('prior\_summary')

## sample: 4000 (posterior sample size)

## observations: 23

## predictors: 2

##

## Estimates:

## mean sd 2.5% 25% 50% 75% 97.5%

## (Intercept) -4.1 0.4 -5.0 -4.4 -4.1 -3.9 -3.5

## xNonmaintained 1.0 0.5 0.1 0.6 1.0 1.3 1.9

## mean\_PPD 0.8 0.3 0.3 0.6 0.8 1.0 1.3

## log-posterior -33.6 1.0 -36.2 -33.9 -33.3 -32.9 -32.6

##

## Diagnostics:

## mcse Rhat n\_eff

## (Intercept) 0.0 1.0 2165

## xNonmaintained 0.0 1.0 2253

## mean\_PPD 0.0 1.0 3144

## log-posterior 0.0 1.0 1620

##

## For each parameter, mcse is Monte Carlo standard error, n\_eff is a crude measure of effective sample size, and Rhat is the potential scale reduction factor on split chains (at convergence Rhat=1).

fit\_leukemia\_two <- rstanarm::stan\_glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

data = leukemia\_two,

family = poisson(link = "log"),

prior = normal(location = 0, scale = 4))

prior\_summary(fit\_leukemia\_two)

## Priors for model 'fit\_leukemia\_two'

## ------

##

## Coefficients

## ~ normal(location = [0,0,0], scale = [4,4,4])

## ------

## See help('prior\_summary.stanreg') for more details

summary(fit\_leukemia\_two)

##

## Model Info:

##

## function: stan\_glm

## family: poisson [log]

## formula: status ~ -1 + interval + x + offset(log(interval\_length))

## algorithm: sampling

## priors: see help('prior\_summary')

## sample: 4000 (posterior sample size)

## observations: 34

## predictors: 3

##

## Estimates:

## mean sd 2.5% 25% 50% 75% 97.5%

## interval0 -4.4 0.5 -5.3 -4.7 -4.3 -4.0 -3.5

## interval23 -3.9 0.4 -4.8 -4.2 -3.9 -3.6 -3.2

## xNonmaintained 1.0 0.5 0.0 0.7 1.0 1.4 2.0

## mean\_PPD 0.5 0.2 0.2 0.4 0.5 0.6 0.9

## log-posterior -43.1 1.3 -46.4 -43.7 -42.8 -42.1 -41.6

##

## Diagnostics:

## mcse Rhat n\_eff

## interval0 0.0 1.0 1366

## interval23 0.0 1.0 1562

## xNonmaintained 0.0 1.0 1318

## mean\_PPD 0.0 1.0 3627

## log-posterior 0.0 1.0 1445

##

## For each parameter, mcse is Monte Carlo standard error, n\_eff is a crude measure of effective sample size, and Rhat is the potential scale reduction factor on split chains (at convergence Rhat=1).

fit\_leukemia\_three <- rstanarm::stan\_glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

data = leukemia\_three,

family = poisson(link = "log"),

prior = normal(location = 0, scale = 4))

prior\_summary(fit\_leukemia\_three)

## Priors for model 'fit\_leukemia\_three'

## ------

##

## Coefficients

## ~ normal(location = [0,0,0,...], scale = [4,4,4,...])

## ------

## See help('prior\_summary.stanreg') for more details

summary(fit\_leukemia\_three)

##

## Model Info:

##

## function: stan\_glm

## family: poisson [log]

## formula: status ~ -1 + interval + x + offset(log(interval\_length))

## algorithm: sampling

## priors: see help('prior\_summary')

## sample: 4000 (posterior sample size)

## observations: 45

## predictors: 4

##

## Estimates:

## mean sd 2.5% 25% 50% 75% 97.5%

## interval0 -4.3 0.5 -5.4 -4.7 -4.3 -4.0 -3.4

## interval14 -4.1 0.5 -5.3 -4.5 -4.1 -3.8 -3.2

## interval31 -3.9 0.5 -4.9 -4.2 -3.9 -3.6 -3.1

## xNonmaintained 1.0 0.5 0.0 0.6 1.0 1.3 2.0

## mean\_PPD 0.4 0.1 0.2 0.3 0.4 0.5 0.7

## log-posterior -49.9 1.4 -53.6 -50.6 -49.6 -48.9 -48.2

##

## Diagnostics:

## mcse Rhat n\_eff

## interval0 0.0 1.0 1622

## interval14 0.0 1.0 1738

## interval31 0.0 1.0 1793

## xNonmaintained 0.0 1.0 1417

## mean\_PPD 0.0 1.0 4123

## log-posterior 0.0 1.0 1694

##

## For each parameter, mcse is Monte Carlo standard error, n\_eff is a crude measure of effective sample size, and Rhat is the potential scale reduction factor on split chains (at convergence Rhat=1).

fit\_leukemia\_events <- rstanarm::stan\_glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

data = leukemia\_events,

family = poisson(link = "log"),

prior = normal(location = 0, scale = 4))

prior\_summary(fit\_leukemia\_events)

## Priors for model 'fit\_leukemia\_events'

## ------

##

## Coefficients

## ~ normal(location = [0,0,0,...], scale = [4,4,4,...])

## ------

## See help('prior\_summary.stanreg') for more details

summary(fit\_leukemia\_events)

##

## Model Info:

##

## function: stan\_glm

## family: poisson [log]

## formula: status ~ -1 + interval + x + offset(log(interval\_length))

## algorithm: sampling

## priors: see help('prior\_summary')

## sample: 4000 (posterior sample size)

## observations: 180

## predictors: 17

##

## Estimates:

## mean sd 2.5% 25% 50% 75% 97.5%

## interval0 -4.5 0.8 -6.2 -4.9 -4.4 -3.9 -3.2

## interval5 -3.9 0.8 -5.6 -4.3 -3.8 -3.3 -2.6

## interval8 -3.4 1.1 -6.0 -4.0 -3.3 -2.7 -1.7

## interval9 -4.4 1.1 -6.8 -5.0 -4.3 -3.7 -2.7

## interval12 -3.3 1.1 -5.8 -3.9 -3.2 -2.6 -1.6

## interval13 -4.7 1.0 -7.1 -5.3 -4.6 -4.0 -3.1

## interval18 -3.9 0.8 -5.6 -4.3 -3.8 -3.3 -2.6

## interval23 -4.2 1.1 -6.7 -4.8 -4.1 -3.5 -2.6

## interval27 -3.8 1.1 -6.2 -4.4 -3.7 -3.0 -2.1

## interval30 -2.6 1.1 -5.2 -3.2 -2.5 -1.8 -0.9

## interval31 -3.1 1.0 -5.5 -3.7 -3.0 -2.4 -1.4

## interval33 -2.3 1.1 -4.9 -2.9 -2.2 -1.5 -0.6

## interval34 -4.2 1.0 -6.5 -4.8 -4.1 -3.5 -2.5

## interval43 -2.6 1.1 -5.2 -3.2 -2.4 -1.8 -0.8

## interval45 -2.1 1.1 -4.6 -2.8 -2.0 -1.4 -0.4

## interval48 -6.7 1.9 -11.0 -7.8 -6.4 -5.3 -4.0

## xNonmaintained 0.5 0.4 -0.4 0.2 0.5 0.8 1.3

## mean\_PPD 0.1 0.0 0.1 0.1 0.1 0.1 0.2

## log-posterior -87.0 3.1 -93.9 -88.9 -86.7 -84.8 -81.9

##

## Diagnostics:

## mcse Rhat n\_eff

## interval0 0.0 1.0 3576

## interval5 0.0 1.0 4107

## interval8 0.0 1.0 3723

## interval9 0.0 1.0 3297

## interval12 0.0 1.0 3544

## interval13 0.0 1.0 3279

## interval18 0.0 1.0 4133

## interval23 0.0 1.0 3880

## interval27 0.0 1.0 3935

## interval30 0.0 1.0 3871

## interval31 0.0 1.0 3589

## interval33 0.0 1.0 3291

## interval34 0.0 1.0 3765

## interval43 0.0 1.0 3575

## interval45 0.0 1.0 4413

## interval48 0.0 1.0 3276

## xNonmaintained 0.0 1.0 3104

## mean\_PPD 0.0 1.0 4077

## log-posterior 0.1 1.0 1385

##

## For each parameter, mcse is Monte Carlo standard error, n\_eff is a crude measure of effective sample size, and Rhat is the potential scale reduction factor on split chains (at convergence Rhat=1).

fit\_leukemia\_times <- rstanarm::stan\_glm(formula = status ~ -1 + interval + x + offset(log(interval\_length)),

data = leukemia\_times,

family = poisson(link = "log"),

prior = normal(location = 0, scale = 4))

prior\_summary(fit\_leukemia\_times)

## Priors for model 'fit\_leukemia\_times'

## ------

##

## Coefficients

## ~ normal(location = [0,0,0,...], scale = [4,4,4,...])

## ------

## See help('prior\_summary.stanreg') for more details

summary(fit\_leukemia\_times)

##

## Model Info:

##

## function: stan\_glm

## family: poisson [log]

## formula: status ~ -1 + interval + x + offset(log(interval\_length))

## algorithm: sampling

## priors: see help('prior\_summary')

## sample: 4000 (posterior sample size)

## observations: 203

## predictors: 19

##

## Estimates:

## mean sd 2.5% 25% 50% 75% 97.5%

## interval0 -4.4 0.8 -6.1 -4.9 -4.3 -3.9 -3.1

## interval5 -3.8 0.8 -5.5 -4.2 -3.7 -3.3 -2.5

## interval8 -3.4 1.1 -5.9 -4.0 -3.3 -2.6 -1.6

## interval9 -4.4 1.0 -6.7 -4.9 -4.2 -3.7 -2.7

## interval12 -3.3 1.0 -5.7 -3.9 -3.2 -2.6 -1.6

## interval13 -6.2 2.0 -11.0 -7.4 -5.9 -4.7 -3.2

## interval16 -3.8 1.0 -6.2 -4.4 -3.6 -3.0 -2.1

## interval18 -3.8 0.8 -5.5 -4.3 -3.8 -3.3 -2.5

## interval23 -4.2 1.0 -6.5 -4.7 -4.1 -3.4 -2.5

## interval27 -5.1 2.2 -10.3 -6.3 -4.7 -3.5 -1.8

## interval28 -3.4 1.1 -5.9 -4.0 -3.2 -2.6 -1.7

## interval30 -2.5 1.1 -5.0 -3.2 -2.4 -1.8 -0.9

## interval31 -3.1 1.1 -5.6 -3.7 -3.0 -2.4 -1.4

## interval33 -2.3 1.1 -4.9 -2.9 -2.2 -1.5 -0.6

## interval34 -4.2 1.0 -6.6 -4.8 -4.1 -3.5 -2.6

## interval43 -2.5 1.1 -5.0 -3.1 -2.4 -1.8 -0.8

## interval45 -2.1 1.1 -4.5 -2.8 -2.0 -1.3 -0.4

## interval48 -6.7 1.9 -11.2 -7.7 -6.4 -5.4 -4.0

## xNonmaintained 0.4 0.4 -0.5 0.1 0.4 0.7 1.3

## mean\_PPD 0.1 0.0 0.1 0.1 0.1 0.1 0.2

## log-posterior -91.3 3.3 -98.9 -93.3 -90.9 -88.9 -86.0

##

## Diagnostics:

## mcse Rhat n\_eff

## interval0 0.0 1.0 3432

## interval5 0.0 1.0 4021

## interval8 0.0 1.0 3538

## interval9 0.0 1.0 3551

## interval12 0.0 1.0 3500

## interval13 0.0 1.0 3275

## interval16 0.0 1.0 3421

## interval18 0.0 1.0 3125

## interval23 0.0 1.0 3474

## interval27 0.0 1.0 3305

## interval28 0.0 1.0 3081

## interval30 0.0 1.0 2785

## interval31 0.0 1.0 2986

## interval33 0.0 1.0 3649

## interval34 0.0 1.0 3070

## interval43 0.0 1.0 3527

## interval45 0.0 1.0 3956

## interval48 0.0 1.0 2958

## xNonmaintained 0.0 1.0 2751

## mean\_PPD 0.0 1.0 4476

## log-posterior 0.1 1.0 1829

##

## For each parameter, mcse is Monte Carlo standard error, n\_eff is a crude measure of effective sample size, and Rhat is the potential scale reduction factor on split chains (at convergence Rhat=1).

By the virtue of regularization by the prior, even the log baseline hazards for intervals without events does not break the model, but has lower posterior means.

Examine model fit with loo. Note @avehtari ([tweet](https://twitter.com/avehtari/status/1063048345118273536), BDA3 co-author, developer of Stan, GPy, etc!) pointed out that in the case of sub-patient level dataset (multiple rows representing intervals for each patient), leave-one-observation-out method is tricky. In this instance, what each observation represents in terms of the duration of the interval is different across models. This likely requires manually coded cross-validation with patient-level nested data frame (each row represents one patient with multiple time points nested within it). The evaluation was suppressed as the results are misleading.

list\_fits\_leukemia <- list(fit\_leukemia\_one = fit\_leukemia\_one,

fit\_leukemia\_two = fit\_leukemia\_two,

fit\_leukemia\_three = fit\_leukemia\_three,

fit\_leukemia\_events = fit\_leukemia\_events,

fit\_leukemia\_times = fit\_leukemia\_times)

lapply(list\_fits\_leukemia, loo::loo)

**Survival curve prediction**

Now we have posterior samples of interval-specific log baseline hazards as well as log hazard ratio for treatment. We can construct treatment group-specific posterior survival curves. Since we have piecewise constant hazards, it is easier to first construct a cumulative hazard function. Here is the constructor.

cunstruct\_cumulative\_hazard\_function <- function(cutpoints, log\_baseline\_hazards, group\_effect) {

## t is a vector of time points. group is {0,1} scalar

cumulative\_hazard\_function <- function(t, group) {

## Boolean for any exposed time in each interval

## length(cutpoints) x length(t)

interval\_exposed <- outer(cutpoints, t, `<`)

## t - cutpoint. Multiply by interval exposed to avoid negative times.

time\_exposed <- -outer(cutpoints, t, `-`) \* interval\_exposed

## Last interval is of width Inf

interval\_widths <- c(diff(cutpoints), Inf)

## For each interval, time exposed cannot exceed interval width.

time\_exposed\_correct <- sweep(x = time\_exposed,

MARGIN = 1,

STATS = interval\_widths,

FUN = pmin)

## Multiply by corresponding baseline hazards to get interval specific cumulative baseline hazards.

interval\_baseline\_cumulative\_hazards <- sweep(x = time\_exposed\_correct,

MARGIN = 1,

STATS = exp(log\_baseline\_hazards),

FUN = `\*`)

## Cumulative baseline hazard vector length(t)

baseline\_cumulative\_hazards <- colSums(interval\_baseline\_cumulative\_hazards)

## return after applying group effect

return(baseline\_cumulative\_hazards \* exp(group\_effect \* group))

}

return(cumulative\_hazard\_function)

}

We can use the tidybayes package to obtain each posterior sample as a row of a data frame. The columns named interval are interval-specific constant log hazards.

tidybayes::tidy\_draws(fit\_leukemia\_three)

## # A tibble: 4,000 x 13

## .chain .iteration .draw interval0 interval14 interval31 xNonmaintained accept\_stat\_\_ stepsize\_\_ treedepth\_\_

## <int> <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>

## 1 1 1 1 -4.50 -4.33 -4.89 1.07 0.951 0.346 2

## 2 1 2 2 -4.83 -4.37 -3.28 0.913 1 0.346 3

## 3 1 3 3 -4.63 -3.40 -3.87 0.940 0.980 0.346 3

## 4 1 4 4 -5.32 -4.86 -3.93 1.50 0.965 0.346 3

## 5 1 5 5 -4.33 -4.02 -4.29 1.42 0.991 0.346 3

## 6 1 6 6 -4.14 -4.41 -4.44 0.702 0.996 0.346 3

## 7 1 7 7 -3.81 -4.09 -3.94 0.561 0.998 0.346 3

## 8 1 8 8 -4.55 -3.99 -3.97 1.14 0.991 0.346 3

## 9 1 9 9 -4.34 -4.47 -3.75 0.806 0.939 0.346 3

## 10 1 10 10 -3.33 -3.70 -3.58 0.175 0.964 0.346 4

## # ... with 3,990 more rows, and 3 more variables: n\_leapfrog\_\_ <dbl>, divergent\_\_ <dbl>, energy\_\_ <dbl>

Then we construct posterior samples of cumulative hazard functions based on the interval-specific constant log hazards. No-so-tidy tidyverse code follows.

cum\_haz\_leukemia\_one <- tidybayes::tidy\_draws(fit\_leukemia\_one) %>%

mutate(`H(t|x)` = pmap(list(`(Intercept)`, xNonmaintained),

function(`(Intercept)`, xNonmaintained){

cunstruct\_cumulative\_hazard\_function(

cutpoints = c(0),

log\_baseline\_hazards = c(`(Intercept)`),

group\_effect = xNonmaintained)

})) %>%

select(.chain, .iteration, .draw, `H(t|x)`)

cum\_haz\_leukemia\_two <- tidybayes::tidy\_draws(fit\_leukemia\_two) %>%

mutate(`H(t|x)` = pmap(list(interval0, interval23, xNonmaintained),

function(interval0, interval23, xNonmaintained){

cunstruct\_cumulative\_hazard\_function(

cutpoints = c(0,23),

log\_baseline\_hazards = c(interval0, interval23),

group\_effect = xNonmaintained)

})) %>%

select(.chain, .iteration, .draw, `H(t|x)`)

cum\_haz\_leukemia\_three <- tidybayes::tidy\_draws(fit\_leukemia\_three) %>%

mutate(`H(t|x)` = pmap(list(interval0, interval14, interval31, xNonmaintained),

function(interval0, interval14, interval31, xNonmaintained){

cunstruct\_cumulative\_hazard\_function(

cutpoints = c(0,14,31),

log\_baseline\_hazards = c(interval0, interval14, interval31),

group\_effect = xNonmaintained)

})) %>%

select(.chain, .iteration, .draw, `H(t|x)`)

cum\_haz\_leukemia\_events <- tidybayes::tidy\_draws(fit\_leukemia\_events) %>%

mutate(`H(t|x)` = pmap(list(interval0, interval5, interval8, interval9, interval12,

interval13, interval18, interval23, interval27, interval30,

interval31, interval33, interval34, interval43, interval45,

interval48, xNonmaintained),

function(interval0, interval5, interval8, interval9, interval12,

interval13, interval18, interval23, interval27, interval30,

interval31, interval33, interval34, interval43, interval45,

interval48, xNonmaintained){

cunstruct\_cumulative\_hazard\_function(

cutpoints = c(0, 5, 8, 9, 12, 13, 18, 23, 27, 30, 31, 33, 34, 43, 45, 48),

log\_baseline\_hazards = c(interval0, interval5, interval8, interval9,

interval12, interval13, interval18, interval23,

interval27, interval30, interval31, interval33,

interval34, interval43, interval45, interval48),

group\_effect = xNonmaintained)

})) %>%

select(.chain, .iteration, .draw, `H(t|x)`)

cum\_haz\_leukemia\_times <- tidybayes::tidy\_draws(fit\_leukemia\_times) %>%

mutate(`H(t|x)` = pmap(list(interval0, interval5, interval8, interval9, interval12, interval13,

interval16, interval18, interval23, interval27, interval28, interval30,

interval31, interval33, interval34, interval43, interval45, interval48,

xNonmaintained),

function(interval0, interval5, interval8, interval9, interval12, interval13,

interval16, interval18, interval23, interval27, interval28, interval30,

interval31, interval33, interval34, interval43, interval45, interval48,

xNonmaintained){

cunstruct\_cumulative\_hazard\_function(

cutpoints = c(0, 5, 8, 9, 12, 13, 16, 18, 23, 27, 28, 30, 31, 33,

34, 43, 45, 48),

log\_baseline\_hazards = c(interval0, interval5, interval8, interval9,

interval12, interval13, interval16, interval18,

interval23, interval27, interval28, interval30,

interval31, interval33, interval34, interval43,

interval45, interval48),

group\_effect = xNonmaintained)

})) %>%

select(.chain, .iteration, .draw, `H(t|x)`)

As you can see, each row corresponds to a posterior sample of a cumulative hazard function that can be evaluated at an arbitrary time point *t*≥0

.

cum\_haz\_leukemia\_times

## # A tibble: 4,000 x 4

## .chain .iteration .draw `H(t|x)`

## <int> <int> <int> <list>

## 1 1 1 1 <fn>

## 2 1 2 2 <fn>

## 3 1 3 3 <fn>

## 4 1 4 4 <fn>

## 5 1 5 5 <fn>

## 6 1 6 6 <fn>

## 7 1 7 7 <fn>

## 8 1 8 8 <fn>

## 9 1 9 9 <fn>

## 10 1 10 10 <fn>

## # ... with 3,990 more rows

Here we define functions to create plotting data and to plot.

create\_plot\_df <- function(cum\_haz\_leukemia\_df) {

## Evaluation time points

times\_df <- data\_frame(t = seq(from = 0, to = cut\_one, by = 1))

cum\_haz\_leukemia\_df %>%

mutate(times\_df = list(times\_df)) %>%

mutate(times\_df = pmap(list(times\_df, `H(t|x)`),

function(times\_df, `H(t|x)`) {

times\_df %>%

mutate(`H(t|1)` = `H(t|x)`(t, 1),

`H(t|0)` = `H(t|x)`(t, 0)) %>%

mutate(`S(t|1)` = exp(-`H(t|1)`),

`S(t|0)` = exp(-`H(t|0)`)) %>%

select(-`H(t|1)`, -`H(t|0)`)

}

)

) %>%

select(-`H(t|x)`) %>%

unnest() %>%

gather(key = treatment,

value = survival,

`S(t|1)`, `S(t|0)`) %>%

mutate(treatment = factor(treatment,

levels = c("S(t|0)", "S(t|1)"),

labels = c("Maintained","Nonmaintained")))

}

summarize\_df <- function(df) {

df %>%

group\_by(treatment, t) %>%

summarize(survival\_mean = mean(survival),

survival\_95upper = quantile(survival, probs = 0.975),

survival\_95lower = quantile(survival, probs = 0.025))

}

plot\_df <- function(df) {

df\_summary <- summarize\_df(df)

df %>%

ggplot(mapping = aes(x = t, y = survival,

group = interaction(.chain, .iteration, .draw, treatment))) +

geom\_line(size = 0.1, alpha = 0.025) +

geom\_line(data = df\_summary,

mapping = aes(y = survival\_mean, group = treatment)) +

geom\_line(data = df\_summary,

mapping = aes(y = survival\_95upper, group = treatment),

linetype = "dotted") +

geom\_line(data = df\_summary,

mapping = aes(y = survival\_95lower, group = treatment),

linetype = "dotted") +

facet\_grid(. ~ treatment) +

theme\_bw() +

theme(axis.text.x = element\_text(angle = 90, vjust = 0.5),

legend.key = element\_blank(),

plot.title = element\_text(hjust = 0.5),

strip.background = element\_blank())

}

Plot different models.

survival\_leukemia\_one <-

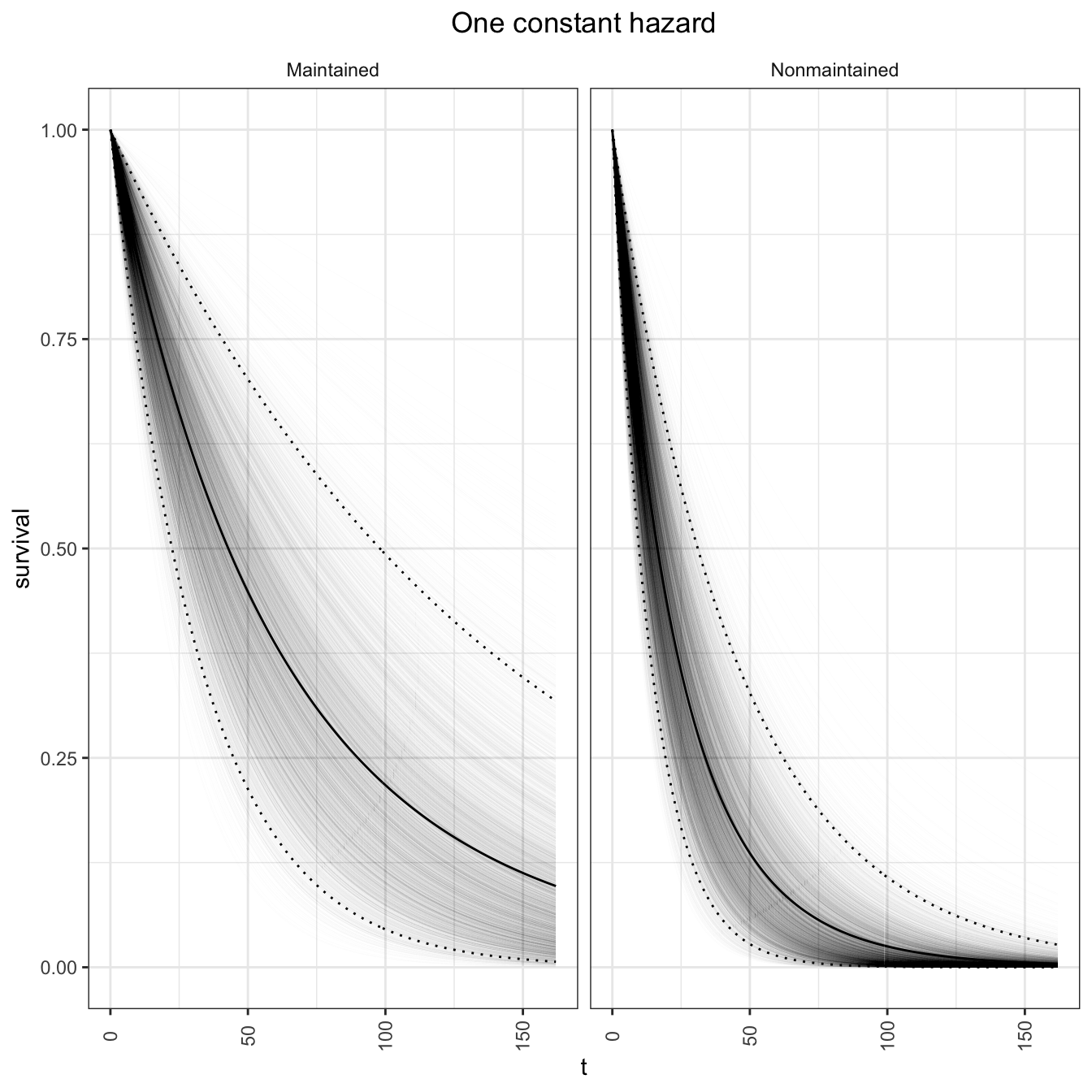
cum\_haz\_leukemia\_one %>%

create\_plot\_df()

survival\_leukemia\_one %>%

plot\_df() +

labs(title = "One constant hazard")



survival\_leukemia\_two <-

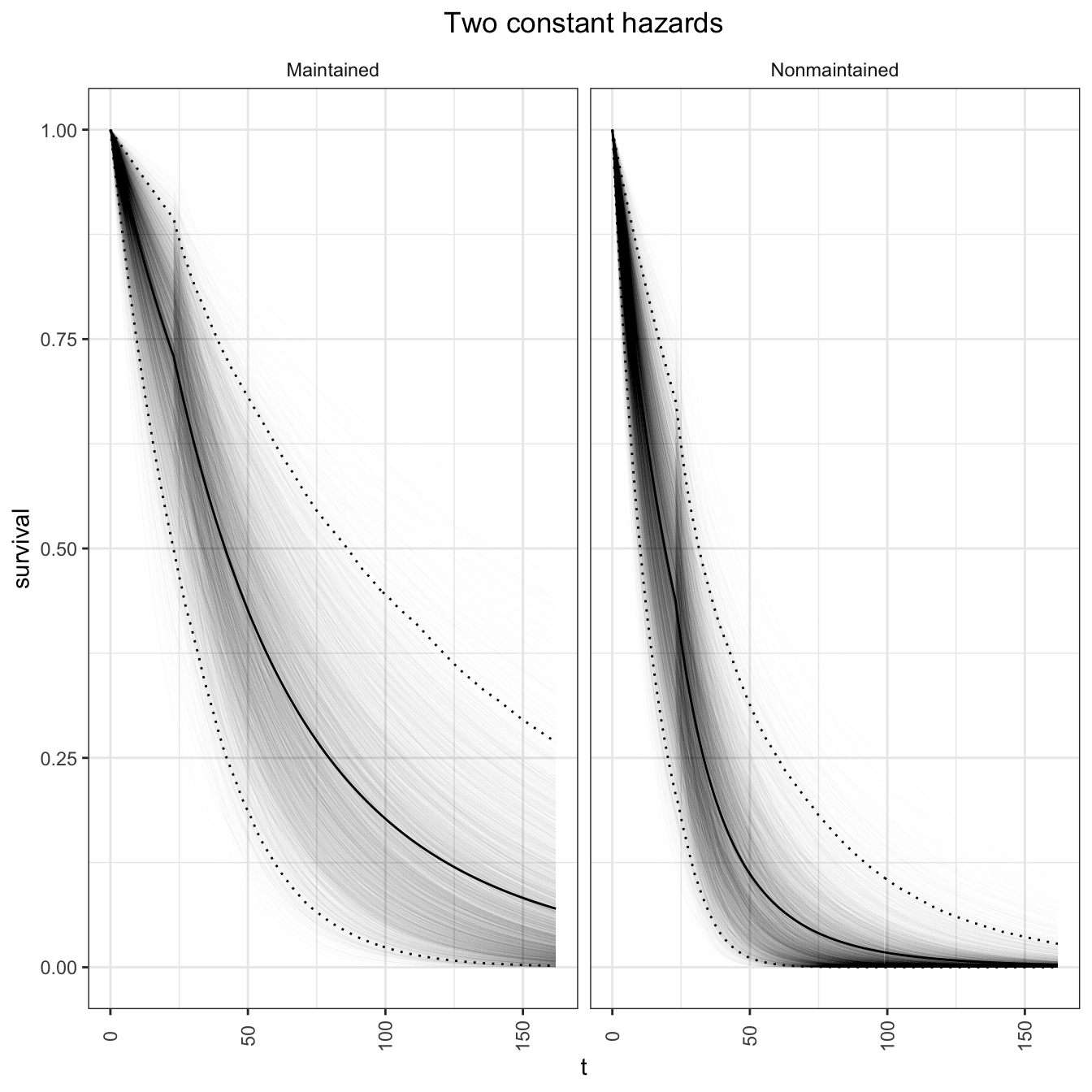
cum\_haz\_leukemia\_two %>%

create\_plot\_df()

survival\_leukemia\_two %>%

plot\_df() +

labs(title = "Two constant hazards")



survival\_leukemia\_three <-

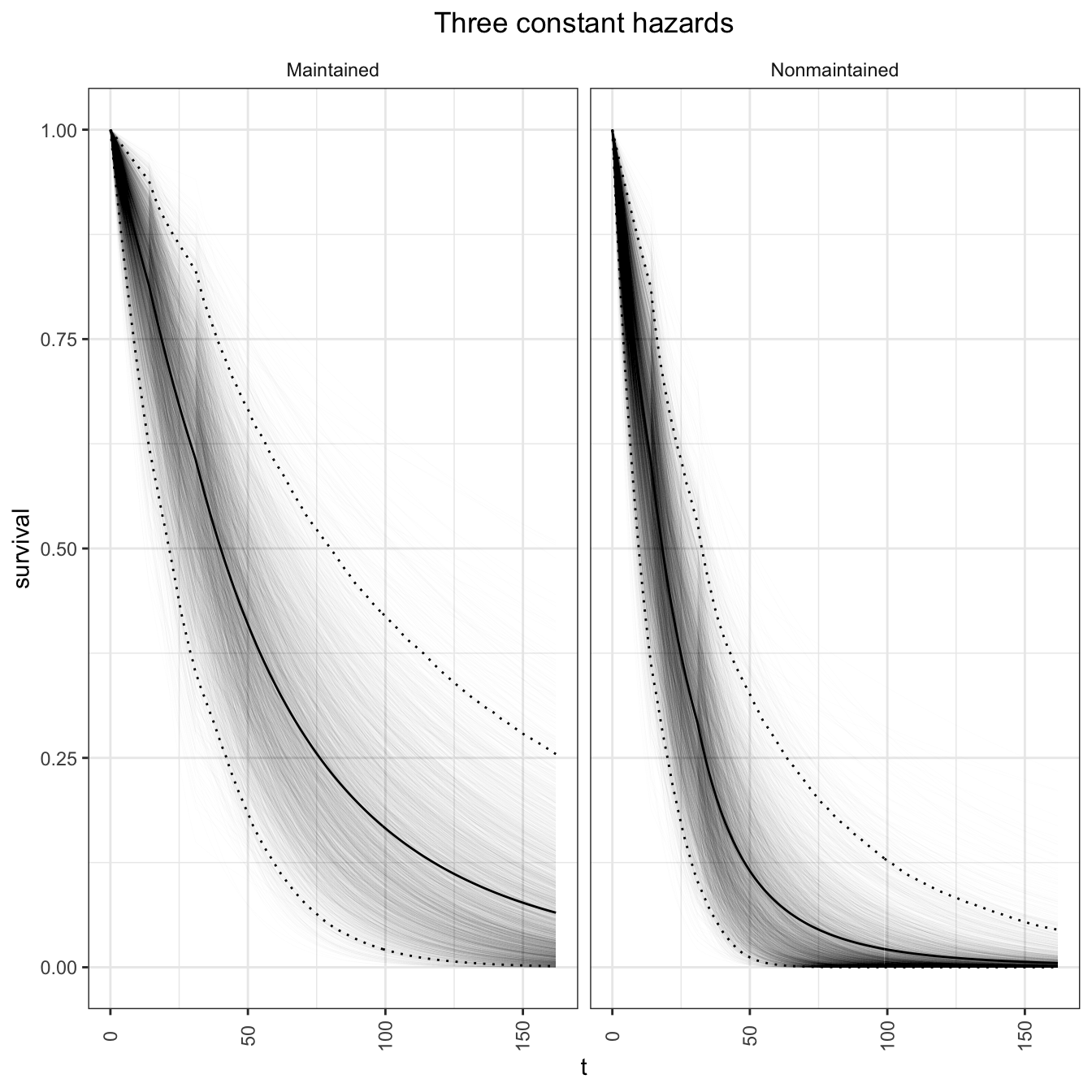
cum\_haz\_leukemia\_three %>%

create\_plot\_df()

survival\_leukemia\_three %>%

plot\_df() +

labs(title = "Three constant hazards")



survival\_leukemia\_events <-

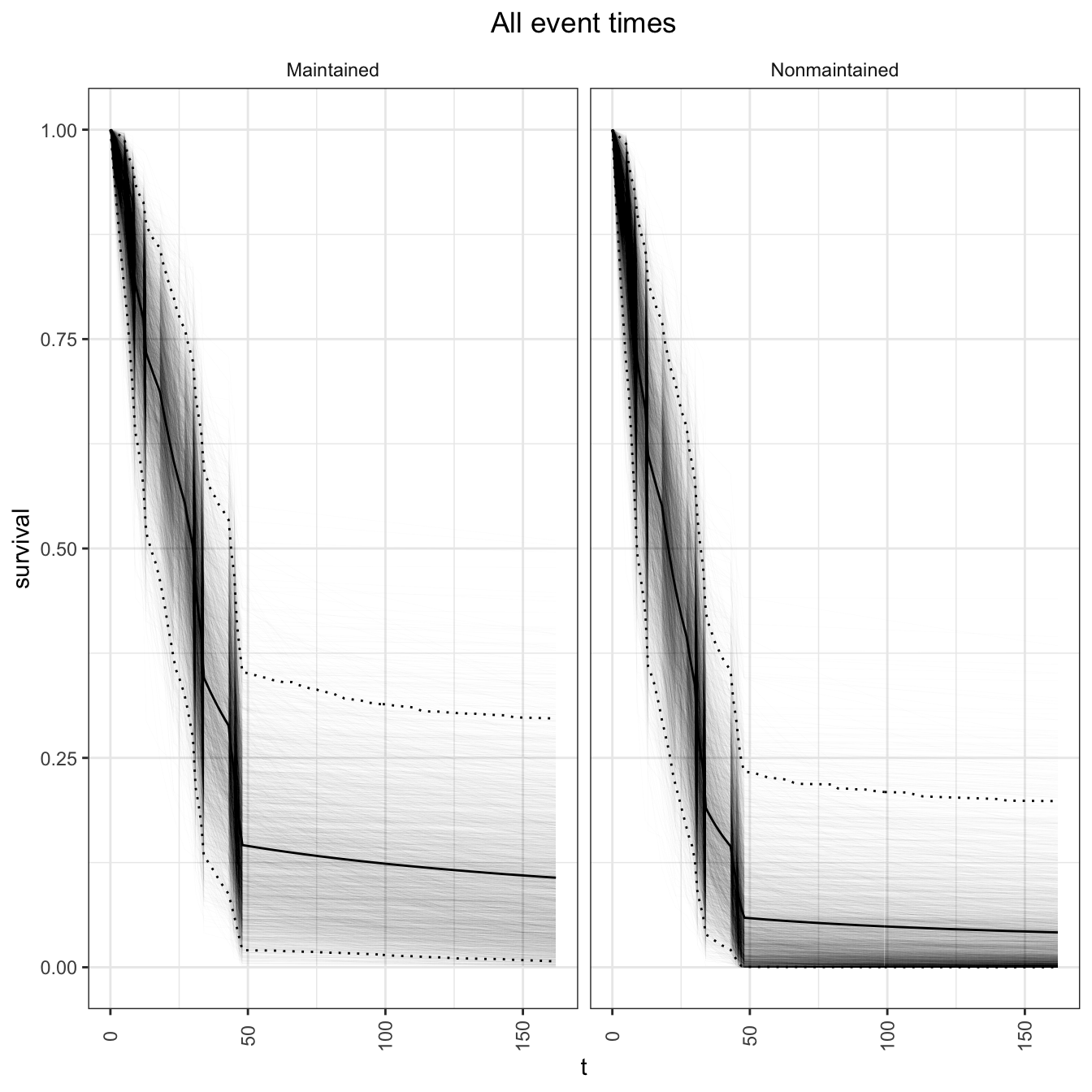
cum\_haz\_leukemia\_events %>%

create\_plot\_df()

survival\_leukemia\_events %>%

plot\_df() +

labs(title = "All event times")



survival\_leukemia\_times <-

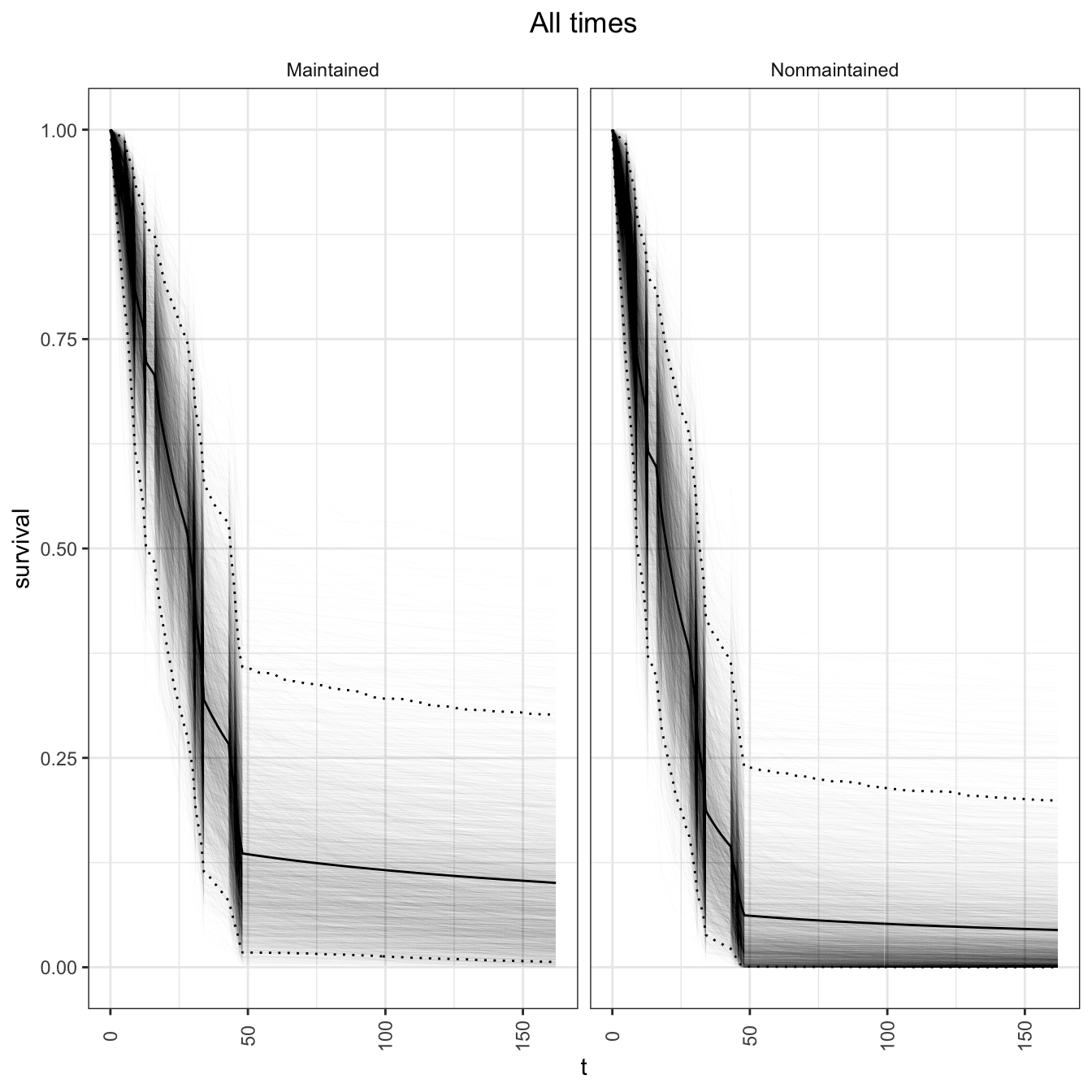
cum\_haz\_leukemia\_times %>%

create\_plot\_df()

survival\_leukemia\_times %>%

plot\_df() +

labs(title = "All times")



## Frequentist Kaplan-Meier estimates

km\_leukemia\_fit <- survival::survfit(formula = Surv(time, status) ~ x,

data = leukemia,

type = "kaplan-meier",

error = "greenwood",

conf.type = "log-log")

survival\_leukemia\_km <-

km\_leukemia\_fit %>%

broom::tidy() %>%

mutate(treatment = gsub("x=", "", strata),

model = "Kaplan-Meier") %>%

rename(t = time,

## These are misnomers, but for plotting purpuse.

survival\_mean = estimate,

survival\_95upper = conf.high,

survival\_95lower = conf.low) %>%

select(treatment, t, survival\_mean, survival\_95upper, survival\_95lower, model) %>%

right\_join(y = crossing(treatment = c("Maintained","Nonmaintained"),

t = seq(from = 0, to = cut\_one, by = 1),

model = "Kaplan-Meier")) %>%

mutate(treatment = factor(treatment,

levels = c("Maintained","Nonmaintained"))) %>%

arrange(treatment, t) %>%

mutate(survival\_mean = if\_else(t == 0, 1, survival\_mean),

survival\_95upper = if\_else(t == 0, 1, survival\_95upper),

survival\_95lower = if\_else(t == 0, 1, survival\_95lower)) %>%

## Need LOCF to recover step function

tidyr::fill(survival\_mean, survival\_95upper, survival\_95lower,

.direction = "down")

Now plot together for comparison. Note the Kaplan-Meier estimates are Frequentist estimates. The Kaplan-Meier interval is Frequentist 95% confidence interval. For other methods, the intervals are 95% credible intervals (posterior probability intervals).

## Summarize

survival\_leukemia\_together <-

bind\_rows(

survival\_leukemia\_one %>%

summarize\_df() %>%

mutate(model = "One constant hazard"),

survival\_leukemia\_two %>%

summarize\_df() %>%

mutate(model = "Two constant hazards"),

survival\_leukemia\_three %>%

summarize\_df() %>%

mutate(model = "Three constant hazards"),

survival\_leukemia\_events %>%

summarize\_df() %>%

mutate(model = "All event times"),

survival\_leukemia\_times %>%

summarize\_df() %>%

mutate(model = "All times"),

## KM

survival\_leukemia\_km)

gg\_together <- survival\_leukemia\_together %>%

ggplot(mapping = aes(x = t, y = survival\_mean,

group = interaction(treatment, model), color = model)) +

geom\_line() +

geom\_line(mapping = aes(y = survival\_95upper),

linetype = "dotted") +

geom\_line(mapping = aes(y = survival\_95lower),

linetype = "dotted") +

facet\_grid(. ~ treatment) +

labs(x = "Time in Days", y = "Survival") +

theme\_bw() +

theme(axis.text.x = element\_text(angle = 90, vjust = 0.5),

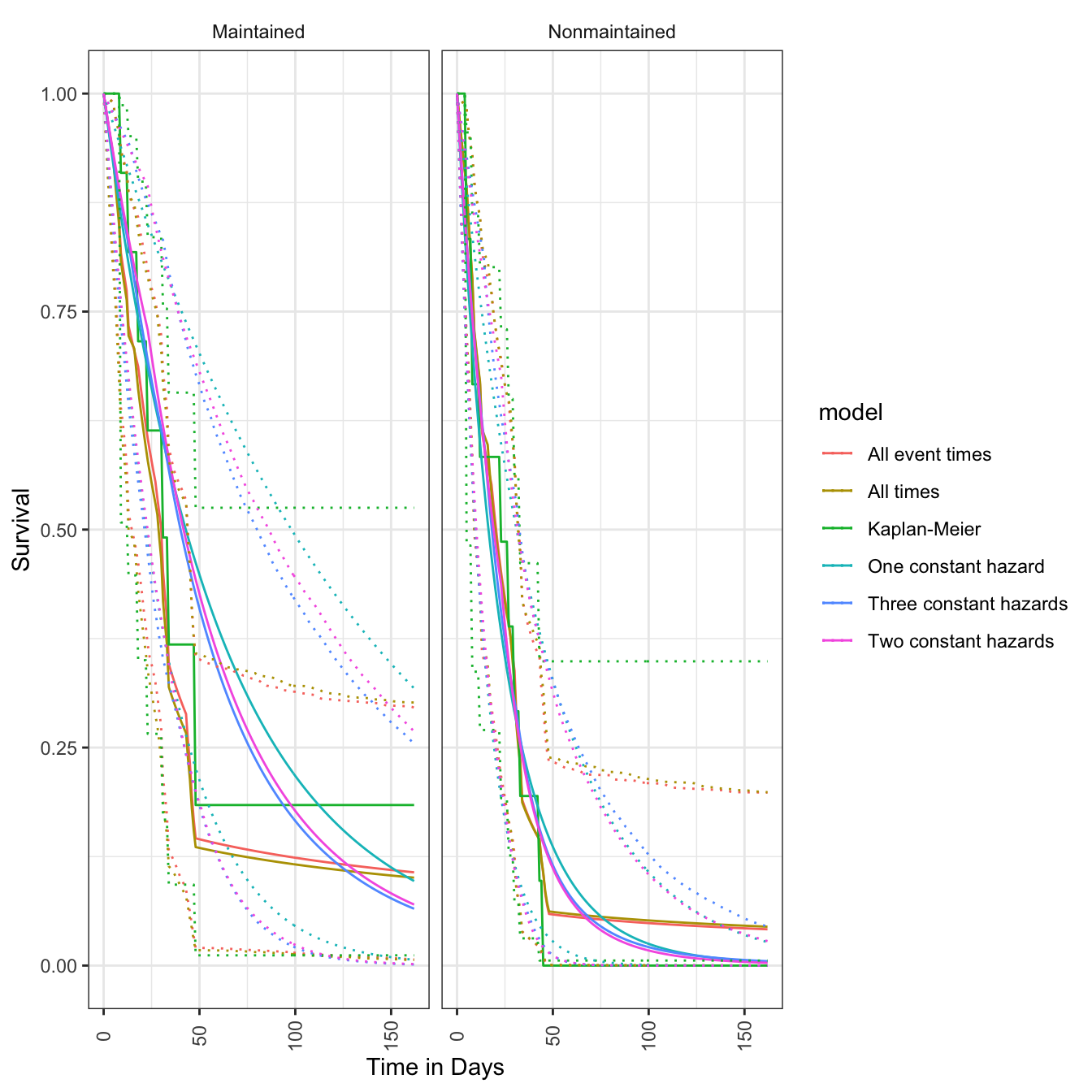
legend.key = element\_blank(),

plot.title = element\_text(hjust = 0.5),

strip.background = element\_blank())

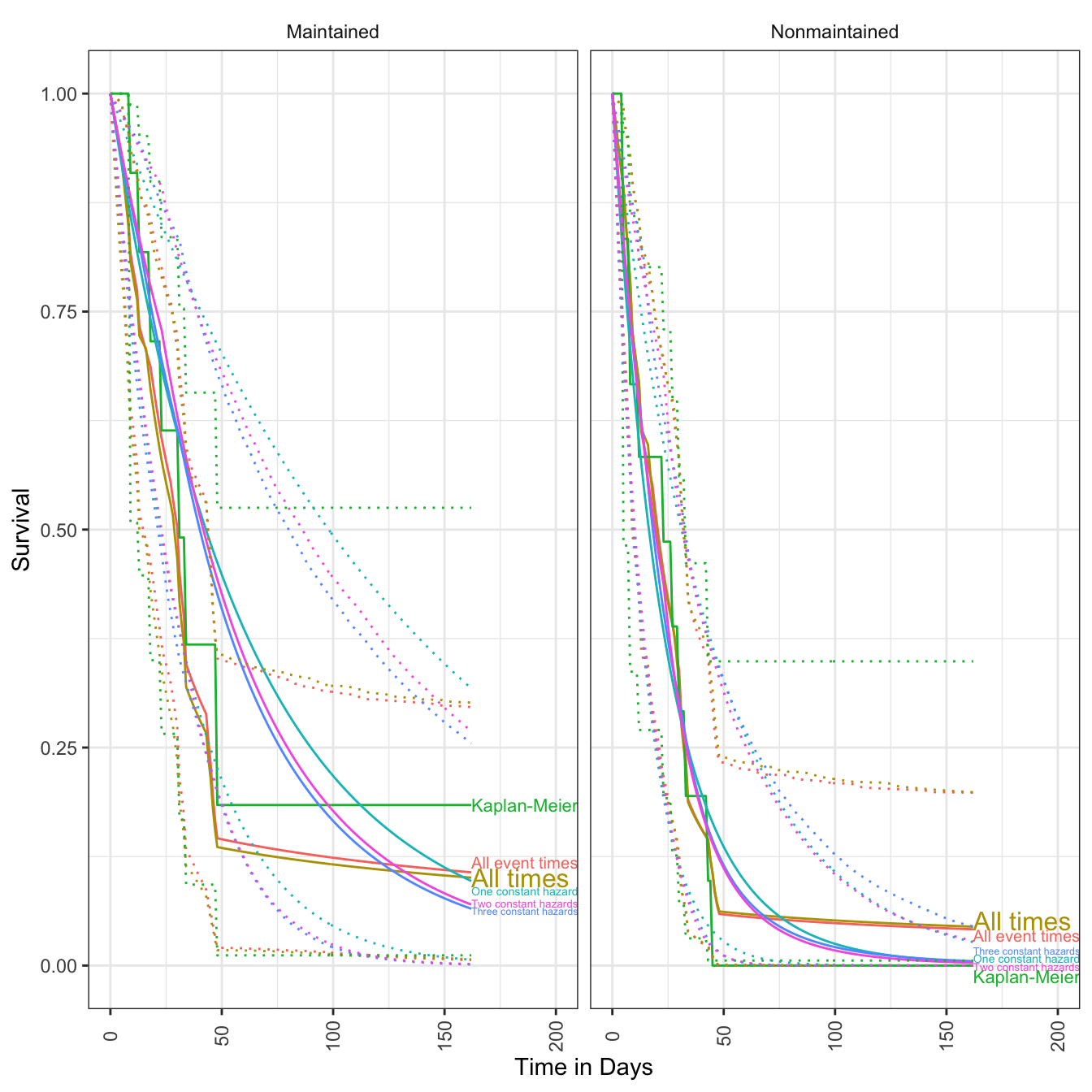
## Regular plot

gg\_together



## Direct labeling

directlabels::direct.label(gg\_together + scale\_x\_continuous(limit = c(0, 200)))



All methods are similar for the Nonmaintained group, whereas there are larger differences for the Maintained group. This is likely due to departure from the exponential model (One constant hazard).

**Record session information**

print(sessionInfo())

## R version 3.5.1 (2018-07-02)

## Platform: x86\_64-apple-darwin15.6.0 (64-bit)

## Running under: macOS 10.14.1

##

## Matrix products: default

## BLAS: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib

## LAPACK: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib

##

## locale:

## [1] en\_US.UTF-8/en\_US.UTF-8/en\_US.UTF-8/C/en\_US.UTF-8/en\_US.UTF-8

##

## attached base packages:

## [1] parallel stats graphics grDevices utils datasets methods base

##

## other attached packages:

## [1] bindrcpp\_0.2.2 directlabels\_2018.05.22 broom\_0.5.0 rstanarm\_2.18.1

## [5] Rcpp\_0.12.19 survival\_2.43-1 forcats\_0.3.0 stringr\_1.3.1

## [9] dplyr\_0.7.7 purrr\_0.2.5 readr\_1.1.1 tidyr\_0.8.2

## [13] tibble\_1.4.2 ggplot2\_3.1.0 tidyverse\_1.2.1 doRNG\_1.7.1

## [17] rngtools\_1.3.1 pkgmaker\_0.27 registry\_0.5 doParallel\_1.0.14

## [21] iterators\_1.0.10 foreach\_1.4.4 knitr\_1.20

##

## loaded via a namespace (and not attached):

## [1] minqa\_1.2.4 colorspace\_1.3-2 ggridges\_0.5.1 rsconnect\_0.8.8

## [5] rprojroot\_1.3-2 ggstance\_0.3.1 markdown\_0.8 base64enc\_0.1-3

## [9] rstudioapi\_0.8 rstan\_2.18.1 svUnit\_0.7-12 DT\_0.4

## [13] fansi\_0.4.0 lubridate\_1.7.4 xml2\_1.2.0 codetools\_0.2-15

## [17] splines\_3.5.1 shinythemes\_1.1.1 bayesplot\_1.6.0 jsonlite\_1.5

## [21] nloptr\_1.2.1 shiny\_1.1.0 compiler\_3.5.1 httr\_1.3.1

## [25] backports\_1.1.2 assertthat\_0.2.0 Matrix\_1.2-14 lazyeval\_0.2.1

## [29] cli\_1.0.1 later\_0.7.5 htmltools\_0.3.6 prettyunits\_1.0.2

## [33] tools\_3.5.1 igraph\_1.2.2 coda\_0.19-2 gtable\_0.2.0

## [37] glue\_1.3.0 reshape2\_1.4.3 cellranger\_1.1.0 nlme\_3.1-137

## [41] crosstalk\_1.0.0 ps\_1.2.0 lme4\_1.1-18-1 rvest\_0.3.2

## [45] mime\_0.6 miniUI\_0.1.1.1 tidybayes\_1.0.3 gtools\_3.8.1

## [49] MASS\_7.3-51 zoo\_1.8-4 scales\_1.0.0 colourpicker\_1.0

## [53] hms\_0.4.2 promises\_1.0.1 inline\_0.3.15 shinystan\_2.5.0

## [57] yaml\_2.2.0 gridExtra\_2.3 loo\_2.0.0 StanHeaders\_2.18.0

## [61] stringi\_1.2.4 dygraphs\_1.1.1.6 pkgbuild\_1.0.2 bibtex\_0.4.2

## [65] rlang\_0.3.0.1 pkgconfig\_2.0.2 matrixStats\_0.54.0 evaluate\_0.12

## [69] lattice\_0.20-35 bindr\_0.1.1 splines2\_0.2.8 labeling\_0.3

## [73] rstantools\_1.5.1 htmlwidgets\_1.3 tidyselect\_0.2.5 processx\_3.2.0

## [77] plyr\_1.8.4 magrittr\_1.5 R6\_2.3.0 pillar\_1.3.0

## [81] haven\_1.1.2 withr\_2.1.2 xts\_0.11-1 modelr\_0.1.2

## [85] crayon\_1.3.4 arrayhelpers\_1.0-20160527 utf8\_1.1.4 rmarkdown\_1.10

## [89] grid\_3.5.1 readxl\_1.1.0 callr\_3.0.0 threejs\_0.3.1

## [93] digest\_0.6.18 xtable\_1.8-3 httpuv\_1.4.5 stats4\_3.5.1

## [97] munsell\_0.5.0 quadprog\_1.5-5 shinyjs\_1.0

## Record execution time

end\_time <- Sys.time()

cat("Started ", as.character(start\_time), "\n")

## Started 2018-11-15 21:02:40

cat("Finished ", as.character(end\_time), "\n")

## Finished 2018-11-15 21:21:37

print(end\_time - start\_time)

## Time difference of 18.94245 mins

* Top Page: <http://rpubs.com/kaz_yos/>
* Github: <https://github.com/kaz-yos>