Bayesian Data Analysis Project

1 Introduction

In this project we wish to model the survival time of lung patients. The table in Figure 1 shows data for 10 of the 137 patients in the trial. The column of interest is the column, **t**, which is the number of days until death, or in some cases the number of days until the patients is censored, which is the case in row number 9 (see column **dead**). This is called right-censoring and corresponds to the more vague statement that the survival time is larger than t. Figure 2 shows the observed durations and which observations are censored. All other columns are covariates and initially we will only focus on the covariate called **therapy**. *Therapy* can be either *standard* or *test*, and we wish to do determine the effect of the test treatment on the survival probability, relative to the standard treatment. To do so, we first outline the main idea of what is called the *equivalent poisson model* in Section 1. In Section 2 we present models of increasingly "cleverer" choices of priors. STAN-code is presented in Section 4, followed by diagnostics, model selection and a final conclusion.

	therapy	cell	t	dead	kps	diagtime	age	prior
0	standard	Squamous	72	dead	60	7	69	no
1	standard	Squamous	411	dead	70	5	64	yes
2	standard	Squamous	228	dead	60	3	38	no
3	standard	Squamous	126	dead	60	9	63	yes
4	standard	Squamous	118	dead	70	11	65	yes
5	standard	Squamous	10	dead	20	5	49	no
6	standard	Squamous	82	dead	40	10	69	yes
7	standard	Squamous	110	dead	80	29	68	no
8	standard	Squamous	314	dead	50	18	43	no
9	standard	Squamous	100	censored	70	6	70	no

Figure 1: Data of 10 patients in the trial. The column **t** represents the survival time for a lung cancer patient.

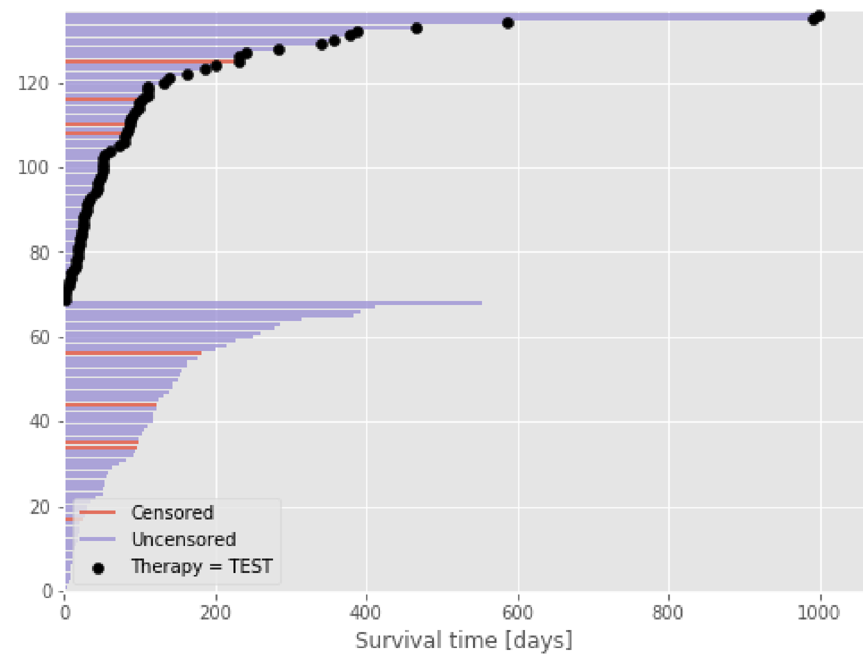


Figure 2: Survival time in days. *Censored* indicates that the individual did not die during the observation period, so we only know the total number of days when death did not occur.

In Figure 3, the curves for the survival probability for the standard and the test treatment are plotted. The most notable in the plot is that the curves cross each other after about 200 days. Thus, we see that the probability of survival for individuals who have survived about 200 days is greatest if these have completed the test treatment.

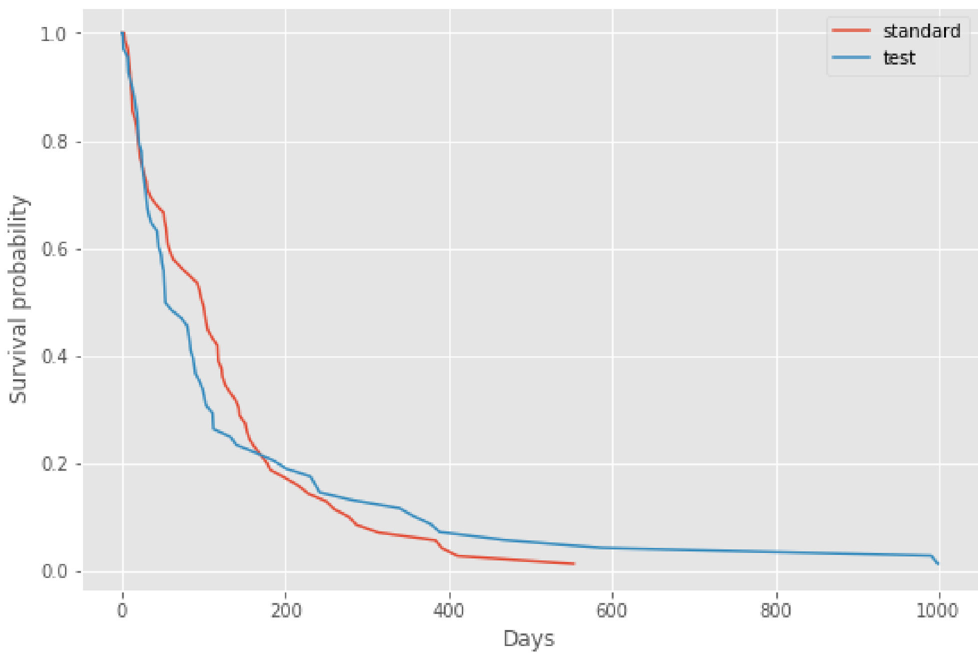


Figure 3: Survival probability for standard and test treatment.

2 The model

2.1 Quick survival analysis recap

Let the survival time be represented as a continuous non-negative random variable, T, with density $f(t)$. The survivor function

$$P(T > t) \equiv S(t) = \exp\left(-\int_0^t \lambda(u)du\right) \tag{1}$$

is a function of the cummulative hazard, $\int_0^t \lambda(u)du$, experienced up until time t.
In the *proportional hazard model* the hazard function is given by

$$\lambda(t) = \lambda_0(t) \exp\left(G(\boldsymbol{x}, \boldsymbol{\beta})\right), \tag{2}$$

with $\lambda_0(t)$ is a so-called baseline hazard function identical for all individuals, $\boldsymbol{\beta}$ is a vector of regression coefficients and $G(\boldsymbol{x}, \boldsymbol{\beta})$ is a function. The second term is expressed using the exponential function because the hazard must be positive. With constant covariates this model implies that ratio between the hazards of two individuals remains contant over time. It is most common to have $G(\boldsymbol{x}, \boldsymbol{\beta}) = \boldsymbol{x}^T\boldsymbol{\beta}$.

2.2 The equivalent poisson model

In the semi-parametric *Cox proportional hazard model* which will be implemented as Model 1 in the next section, we divide the time interval (which contains all survival times in the dataset) into T intervals, $(0, s_1], (s_1, s_2], .., (s_{T_1}, s_T]$, and assume the baseline hazard to be constant throughout each interval, i.e. $\lambda_0(t) = \lambda_j$ if $t \in (s_{j-1}, s_j]$. The hazard of the i'th individual at time interval j is then

$$\lambda_{ij} = \lambda_j \exp(\mathbf{x}_i^T\boldsymbol{\beta}). \tag{3}$$

The likelihood contribution from i'th individual with survival time, y_i, and binary censoring variable

$$\nu_i = \begin{cases} 1 & \text{for death,} \\ 0 & \text{for censoring,} \end{cases} \tag{4}$$

is given by

$$L_i(\boldsymbol{\beta}, \boldsymbol{\lambda}) = f_i(y_i)^{\nu_i} S_i(y_i)^{(1-\nu_i)}. \tag{5}$$

Let

$$\delta_{ij} = \begin{cases} 1 & \text{if } y_i \in I_j = [s_{j-1}, s_j), \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Then, by using the piecewise constant hazard function of equation (3) in equation (5),

$$L_i(\boldsymbol{\beta}, \boldsymbol{\lambda}) = (\lambda_j \exp(\mathbf{x}_i^T\boldsymbol{\beta}))^{\delta_{ij}\nu_i} \exp\left\{-\delta_{ij}\left[\lambda_j(y_i - s_{j-1}) + \sum_{g=1}^{j-1} \lambda_g(s_g - s_{g-1})\right] \exp(\mathbf{x}_i^T\boldsymbol{\beta})\right\}. \tag{7}$$

This likelihood can be rewritten[1]

$$L_i(\boldsymbol{\beta}, \boldsymbol{\lambda}) = \prod_{g=1}^{j} \exp\left\{-\lambda_g t_{i,g} \exp(\mathbf{x}_i^T\boldsymbol{\beta})\right\} (\lambda_g t_{i,g} \exp(\mathbf{x}_i^T\boldsymbol{\beta}))^{\delta_{i,g}\nu_i} / (\delta_{i,g}\nu_i)!, \tag{8}$$

where $t_{i,g}$ is the amount of time individual i is under exposure of its hazard, $\lambda_j \exp(\mathbf{x}_i^T \boldsymbol{\beta})$. We can recognize the likelihood in equation (8) as being equivalent to the likelihood of j independent poisson random variables (recall the parametric form of the probability mass function of a poisson random variable, $\frac{\mu e^{-\mu}}{k!}$). Thus, for each individual, i, we consider for each interval from $g_i \in \{1, ..., j(i)\}$ (here $j(i)$ denotes the index of the interval in which individual i died or was censored) the observation, $\delta_{i,g}\nu_i$, as drawn from a poisson random variable with mean (and variance) given by $\mu_{i,g} = \lambda_g t_{i,g} \exp(\mathbf{x}_i^T \boldsymbol{\beta})$. This is obviously a rather strange model, since a poisson variable can take any non-negative integer value and not just one and zero, as is the case for $\delta_{i,g}\nu_i$. The smart trick here is that if we do inference on this pseudo model, we are simultaneously doing inference on the model in equation (7). To summarize, we have $\sum_i^N j(i)$ poisson observations with

$$\log(\mu_{i,g}) = \log(t_{i,g}) + \log(\lambda_g) + \mathbf{x}_i^T \boldsymbol{\beta},$$

where we are only considering non-zero exposure intervals, i.e. we exclude all observations with $g_i > j(i)$ (corresponding to the entire grey area of Figure 2) since these can cause numerical issues, although these observations shouldn't theoretically interfere with the parameter estimation.

3 Choices for priors

3.1 Model 1

The first model we investigate is the Cox proportional hazard model presented in Section 2.2. Here $beta \sim N(0, 2)$. Recall that $\beta = 0$ corresponds to no effect of the test therapy. This is therefore a very vague prior. The baseline hazard priors, λ_j, $j = \{1, .., T\}$, are all independent (we start simple!) and drawn as $\lambda_j \sim gamma(0.1, 0.1)$. We do this to constrain them to be reasonably small. Intuitively, this Independence between is neighbouring baseline hazards is rather unrealistic and we actually intended to build a third model where we would have put additional structure on the baseline hazard prior and on the time dependent regression coefficients that we introduce in Model 2 below.

3.2 Model 2

Model 1 can be improved by allowing the test therapy to have a time varying effect on the hazard rate, i.e. $\beta(t)$. This is achieved by estimating a beta coefficient for each time interval, β_j. To keep things simple, no structure is put on these β_j. Again each coefficient is drawn as $beta_j \sim N(0, 2)$.

4 Stan Code

Model 1:
The STAN model (see below) for Model 1 is run using the command:

```
fit = sm.sampling(data=death_dat3, algorithm="HMC",
                  seed=1, iter=600, chains=4, warmup=200).
```

We use 4 Markov chains, each drawing 600 samples using the Hamiltonian MC sampler. We discard the first 200 samples since these will probably contain some transient behavior non-representative of the actual stationary target posterior distribution.
Notice, that we are actually fitting a model consisting of 745 possion variables. There is one observation for each individual in each time interval the individual is alive. In the generated quantities we have build a 137-dimensional (the number of unique individuals in the trial) log-likelihood vector by grouping all pseudo observations belonging to the same individual. We do this in order to later do leave-one-individual-out cross validation.

```
sudel is a rvival_model="""
data {
  int<lower=1> N;          // number of individuals
  int<lower=1> N_tot;      // total number of pseudo poisson observations
  int<lower=1> T;          // number of time intervals
  //int<lower=0> M;        // number of covariates
  int<lower=0, upper=40> base_id[N_tot];  // time interval index for each pseudo obs.

  int<lower=0, upper=1> death_array[N_tot];    // 1 for observed death, 0 otherwise
  vector[N_tot] x;                             // covariates

  vector<lower=0>[N_tot] expo;  // exposure time (time alive) in each interval
}

transformed data {
  vector[N_tot] log_expo = log(expo);  // log-duration for each timepoint
}
parameters {
  real beta;                   // regression coefficient
  vector<lower=0>[T] lambda0;  // baseline hazard for each timepoint t
}
model {
  beta ~ normal(0, 2);
  lambda0 ~ gamma(0.1,0.1);
  for (n_tot in 1:N_tot) {
    death_array[n_tot] ~ poisson_log( log(lambda0[base_id[n_tot]])
                                    + log_expo[n_tot]   + x[n_tot] * beta );
  }

}
generated quantities {
  vector[N] log_lik;
  int n;
  n = 1;
  log_lik = rep_vector(0,N);

  // log_lik for loo-psis
  for (n_tot in 1:N_tot) {
```

```
        log_lik[n] += poisson_log_lpmf(death_array[n_tot]|
                          log(lambda0[base_id[n_tot]])+log_expo[n_tot]+x[n_tot]*beta );

        // increment individual count if next time interval comes before the current
        // only because of bad programming on our part
        if (n_tot > 1){
            if (base_id[n_tot] <= base_id[n_tot-1]){
                n += 1;
            }
        }
    }
}
"""
```

Model 2:
To make the model time dependent, we now define β to be a vector in the parameters block:

```
vector[T] beta;
```

Unfortunately, we run into two severe problems. We will need to chance the therapy covariate vector $x \leftarrow 1 - x$, since we can't estimate a baseline hazard for the standard therapy patients, when non of these survive past the first 500 days. We also run into another severe problem, since we are trying to estimate a local proportionate hazard model when there is no data from the standard therapy. This makes no sense. Putting autoregressive stucture on both the baseline and the now time dependent regression coefficient, $\beta(t)$, should offer sufficient regularization in order to do estimation.

5 Diagnostics

Model 1:
All $\hat{R}$-values are smaller than 1.03, and we seem to have collected a reasonable number of effective samples. This can be confirmed by inspecting the notebook below.
In order to evaluate model bias and the reliability of the PSIS-LOO estimates, we look at the distribution of k-values. Generally, the k values are below 0.7, which is the criterion for the PSIS-LOO estimates to be considered reliable [2]. However, it should be noted that some k-values exceed 0.7, so the PSIS-LOO estimates for Model 1 should not be trusted completely.

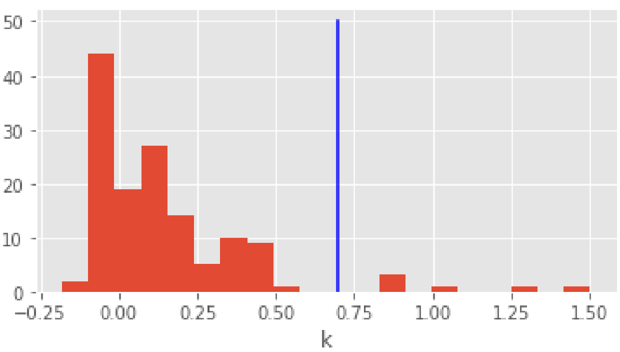


Figure 4: Distribution of k-values for Model 1.

Model 2: In the time-dependent model the $\hat{R}$-values were off the charts (i.e. infinite) due to data being so sparse, and for the same reason the PSIS-LOO analysis fails. A more in-depth description is given in the conclusion.

6 Posterior predictive checking

Model 1:
We note that the Cox proportional hazard model correctly estimates the magnitude of the hazard rate, as can be seen in Figure 5 (right).

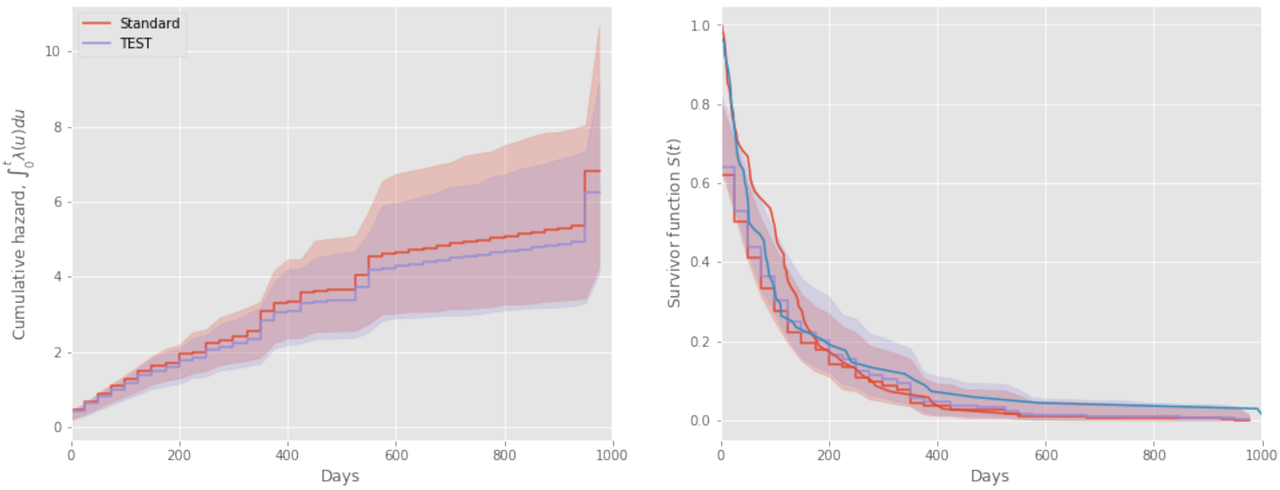


Figure 5: The standard and test treatment in Model 1. **Left:** Cumulative hazard and **right:** Survivor function $S(t)$ versus time in days.

Model 2:
From the survivor function (right) in Figure 6 we see that the time dependent model correctly detects the hazard rate change that occurs after approximately 200 days. Obviously, this model can't estimate the β coefficients for the whole period since there is no survivors past 500 days for the standard treatment.

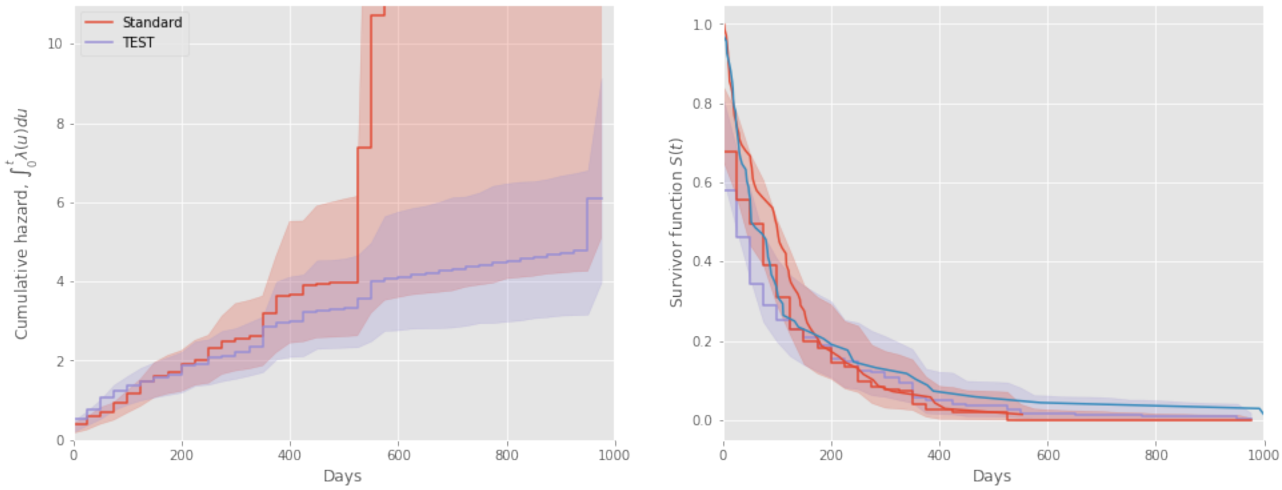


Figure 6: The standard and test treatment in Model 2. **Left:** Cumulative hazard and **right:** Survivor function $S(t)$ versus time in days.

7 Model comparison

Table 1 summarizes the PSIS-LOO values for the tested models. The table shows the expected log predictive density `elpd_loo`, the difference in expected log predictive density between the models `elpd_diff` and the effective number of parameters `eff_params` .

Model	Method	elpd_loo	elpd_diff	eff_params
Model 1	Cox P.H.	-454	-	31.6
Model 2	time dependent beta	not applicable		

Table 1: Comparison of Model 1 and Model 2 based on PSIS-LOO values

8 Conclusion

We have implemented the classic Cox proportional hazard model in the STAN and according to this model, the test therapy has an overall positive effect on the survivor function, i.e. the hazard is on average reduced by 8%. The time dependent model correctly detects the crossing of the hazard rates. Unfortunately, due to bad modelling choices, PSIS-LOO analysis fails for this model. The unstructured piecewise constant hazard rate model cannot be properly estimated when data is so sparse that some intervals have no data for both standard and test therapy patients. We can easily identify several low hanging fruits regarding improvements. First of all it would generally be advantageous to work with increasingly longer time intervals since data gets more and more sparse. It could be worth trying intervals bounds like [20, 40, 64, 92, 124, 160, 200, 244, 292, 344, 400, 460, 524, 592, 664, 740, 820, 904, 992, 1084]. Certainly it makes no sense to try to estimate $beta(t)$ after 500 since all patients with the standard therapy have passed away. In this sparse case study, a simple parametric model, such as a the Weibull model, would most likely have been superior to the semi-parametric model we tried to estimate.

References

[1] Adam Branscum Timothy E. Hanson Ronald Christensen, Wesley Johnson, “Bayesian ideas and data analysis: An introduction for scientists and statisticians,” *International Statistical Review*, vol. 79, no. 2, pp. 285–286, 2011.

[2] Aki Vehtari, Andrew Gelman, and Jonah Gabry, “Practical bayesian model evaluation using leave-one-out cross-validation and waic,” *Statistics and Computing*, vol. 27, no. 5, pp. 1413–1432, Sept. 2017.

Bayesian Semi-parametric Survival Analysis Notebook

by Michael Wamberg & Bjarke Hastrup

The equivalent poisson model

Survival function:

$$S(t) = exp\left(-\int_0^t \lambda(u)du\right)$$

Cox proportional hazard:

$$\lambda_{ij} = \lambda_j \exp(\mathbf{x}_i^T \boldsymbol{\beta})$$

Likelihood contribution from i'th individual, who dies or is censored in j'th time interval: $\nu_i = 1$ for death, 0 otherwise

$$L_i(\boldsymbol{\beta}, \lambda | D_i) = (\lambda_j \exp(\mathbf{x}_i^T \boldsymbol{\beta}))^{\delta_{ij}\nu_i} \exp\left\{-\delta_{ij}\left[\lambda_j(y_i - s_{j-1}) + \sum_{g=1}^{j-1} \lambda_g(s_g - s_{g-1})\right] \exp(\mathbf{x}_i^T \boldsymbol{\beta}\right.$$

$$= \prod_{g=1}^{j} \exp\left\{-\lambda_g t_{i,g} \exp(\mathbf{x}_i^T \boldsymbol{\beta})\right\}(\lambda_g t_{i,g} \exp(\mathbf{x}_i^T \boldsymbol{\beta}))^{\delta_{i,g}\nu_i} \Big/ (\delta_{i,g}\nu_i)!$$

We recognize this as poisson likelihood: $\frac{\mu e^{-\mu}}{k!}$.

$$\log(\mu_{i,g}) = \log(t_{i,g}) + \log(\lambda_g) + \mathbf{x}_i^T \boldsymbol{\beta}$$

In [449]:
```
%load_ext autoreload
%autoreload 2
%matplotlib inline
import random
random.seed(1100038344)
import survivalstan
import numpy as np
import pandas as pd
from stancache import stancache
from matplotlib import pyplot as plt
import statsmodels
import pystan
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

In [450]:
```
# matplotlib options
plt.style.use('ggplot')
%matplotlib inline
plt.rcParams['figure.figsize'] = (9, 6)
```

In [451]:
```
# matplotlib options
plt.style.use('ggplot')
%matplotlib inline
plt.rcParams['figure.figsize'] = (9, 6)
```

In [452]:
```
df = pd.read_csv('valung.csv')
df.head()
len(df)
```

Out[452]: 137

In [453]:
```
df.dead[df.dead == 'dead'] = 1;
df.dead[df.dead == 'censored'] = 0;
```

```
/home/bjarke/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy (http://pandas.pyd
ata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  """Entry point for launching an IPython kernel.
/home/bjarke/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy (http://pandas.pyd
ata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
```

In [454]:
```
df.dead = df.dead.astype(int)
```

In [455]:
```
df.head()
```

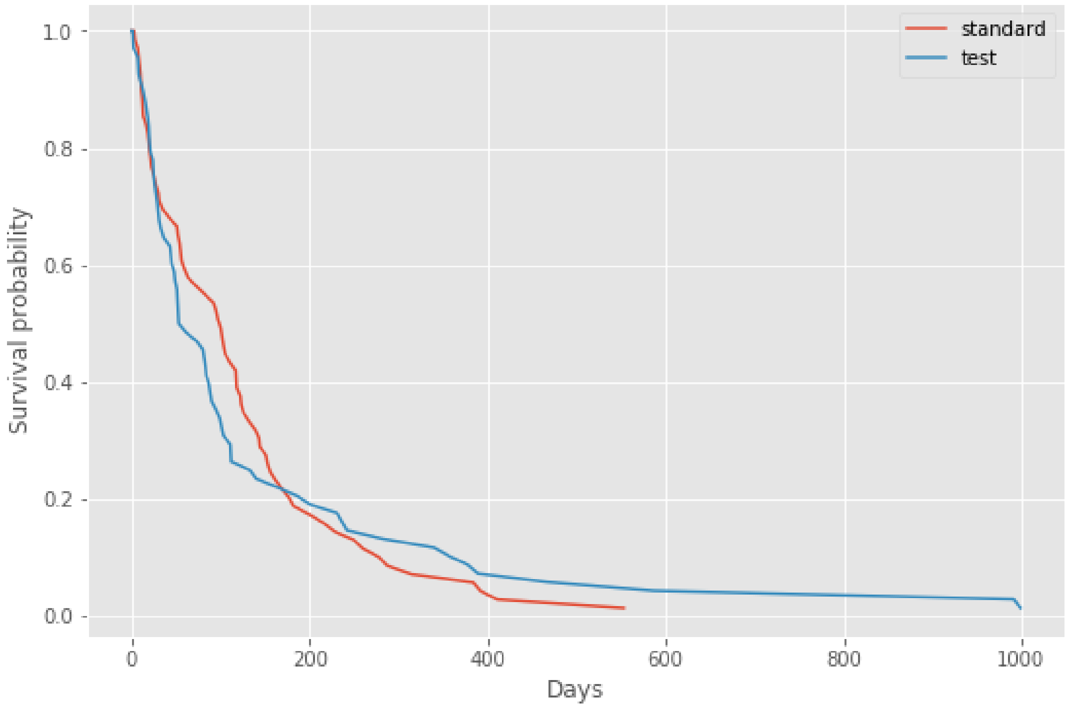
Out[455]:

	therapy	cell	t	dead	kps	diagtime	age	prior
0	standard	Squamous	72	1	60	7	69	no
1	standard	Squamous	411	1	70	5	64	yes
2	standard	Squamous	228	1	60	3	38	no
3	standard	Squamous	126	1	60	9	63	yes
4	standard	Squamous	118	1	70	11	65	yes

We can use the survivalstan package to generate survivor function plots:

In [456]:
```
survivalstan.utils.plot_observed_survival(df=df[df['therapy']=='standard
survivalstan.utils.plot_observed_survival(df=df[df['therapy']=='test'],
plt.legend();
plt.xlabel("Days");
plt.ylabel("Survival probability")
```

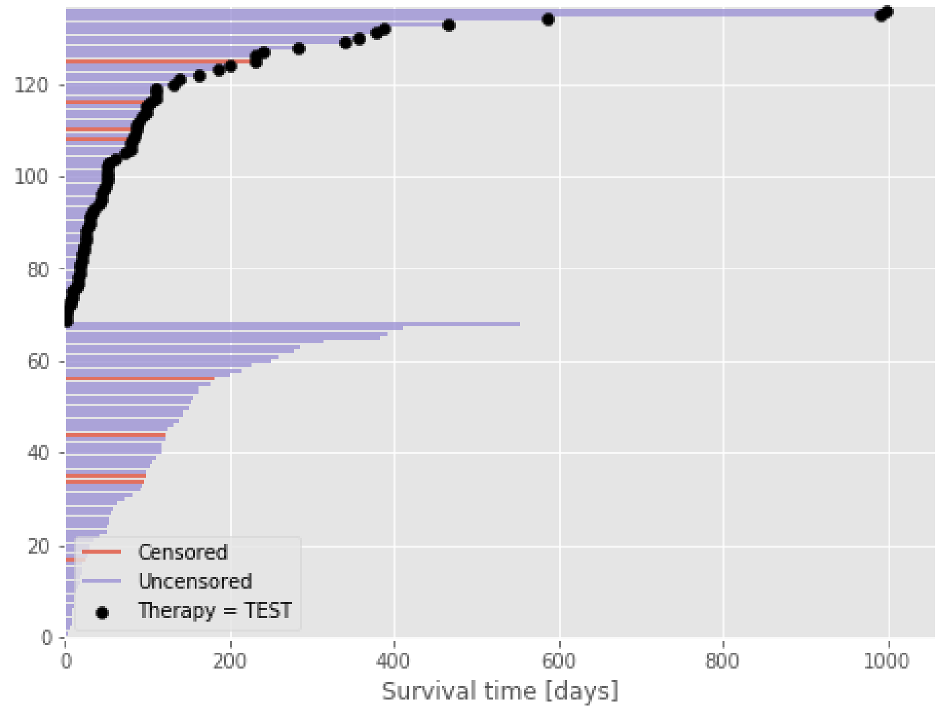
Out[456]:
```
Text(0, 0.5, 'Survival probability')
```



In [457]:
```
df.therapy = (df.therapy == 'test').astype(np.int64)
n_patients = df.shape[0]
patients = np.arange(n_patients)
```

In [458]:
```
df_sort = df.sort_values(by=['therapy', 't'])
```

In [459]:
```
import seaborn as sns
fig, ax = plt.subplots(figsize=(8, 6))
blue, _, red = sns.color_palette()[:3]
ax.hlines(patients[df_sort.dead.values == 0], 0, df_sort[df_sort.dead.va
          color=blue, label='Censored');
ax.hlines(patients[df_sort.dead.values == 1], 0, df_sort[df_sort.dead.va
          color=red, label='Uncensored');
ax.scatter(df_sort[df_sort.therapy.values == 1].t, patients[df_sort.the
           color='k', zorder=10, label='Therapy = TEST');
ax.set_xlim(left=0);
ax.set_xlabel('Survival time [days]');
ax.set_ylim(-0.25, n_patients + 0.25);
ax.legend(loc='top right');
```



In [460]:
```
interval_length = 25
interval_bounds = np.arange(0, df.t.max() + interval_length + 1, interva
n_intervals = interval_bounds.size - 1
intervals = np.arange(n_intervals)
```

It could be beneficial to have a increasingly longer time intervals like below. But we didn't have time to implement this unfortunately.

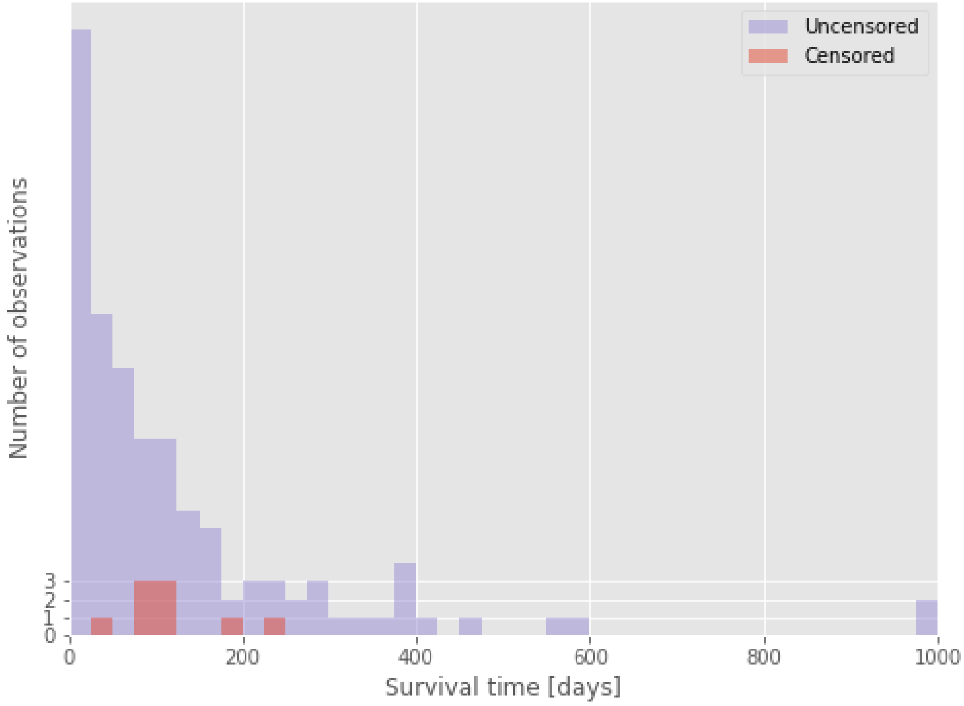
In [378]:
```
vec = []
ff=20
fff=20
for i in range(20):
    vec.append(fff)
    fff += ff
    ff  += 4
```

In [379]:
```
vec
```

Out[379]:
```
[20,
 40,
 64,
 92,
 124,
 160,
 200,
 244,
 292,
 344,
 400,
 460,
 524,
 592,
 664,
 740,
 820,
 904,
 992,
 1084]
```

In [461]:
```
interval_length = 25
interval_bounds = np.arange(0, df.t.max() + interval_length + 1, interva
n_intervals = interval_bounds.size - 1
intervals = np.arange(n_intervals)
```

In [462]:
```
fig, ax = plt.subplots(figsize=(8, 6))
ax.hist(df[df.dead == 1].t.values, bins=interval_bounds,
        color=red, alpha=0.5, lw=0,
        label='Uncensored');
ax.hist(df[df.dead == 0].t.values, bins=interval_bounds,
        color=blue, alpha=0.5, lw=0,
        label='Censored');
ax.set_xlim(0, interval_bounds[-1]);
ax.set_xlabel('Survival time [days]');
ax.set_yticks([0, 1, 2, 3]);
ax.set_ylabel('Number of observations');
ax.legend();
```



In [463]:
```
last_period = np.floor((df.t - 0.01) / interval_length)

death = np.zeros((n_patients, n_intervals))
death[patients, last_period.astype(int)] = df.dead
```

In [464]:
```
exposure = np.greater_equal.outer(df.t, interval_bounds[:-1]) * interva
exposure[patients, last_period.astype(int)] = df.t - interval_bounds[la
```

In [465]:
```
death_dat = {'N': len(df),
             'n_intervals':  n_intervals,
             #'death_array': death_array,
             #'meta_array':  meta_array,
             #'expo_array':  expo_array}
             'death':     death.astype(int),
             'therapy':   np.asarray(df.therapy),
             'exposure':  exposure}
```

In [466]:
```
#death_array = np.matrix.flatten(death)
#expo_array = np.matrix.flatten(exposure)
#therapy_array = np.asarray(df.therapy)
#meta_array = np.tile(meta_array, (n_intervals, 1))
#meta_array = meta_array.flatten('F')
```

In [467]:
```
death_array = np.matrix.flatten(death).astype(int)
```

In [468]:
```
expo_array = np.matrix.flatten(exposure)
```

In [469]:
```
therapy_array = np.asarray(df.therapy)
therapy_array = np.tile(therapy_array, (n_intervals, 1))
therapy_array = therapy_array.flatten('F')
```

In [470]:
```
base_id = np.arange(1,n_intervals+1)
print(str(base_id.shape))
base_id = np.tile(base_id, len(df))
base_id
```

```
(40,)
```

Out[470]:
```
array([ 1,  2,  3, ..., 38, 39, 40])
```

In [471]:
```
death_dat2 ={'N':      len(df),
             'N_tot':  len(death_array),
             'T':      n_intervals,
             'M': 1,
             'base_id': base_id,
             'death_array':   death_array,
             'x': therapy_array,
             'expo':    expo_array}
```

In [472]:
```
expo_new  = expo_array[expo_array > 0]
N_tot_new = len(expo_new)
base_id_new = base_id[expo_array > 0]
death_array_new = death_array[expo_array > 0]
therapy_array_new = therapy_array[expo_array > 0]
```

In [473]:
```
death_dat3 ={'N':      len(df),
             'N_tot':  N_tot_new,
             'T':      n_intervals,
             'M': 1,
             'base_id': base_id_new,
             'death_array':   death_array_new,
             'x': therapy_array_new,
             'expo':    expo_new}
```

In [ ]:

Stan code for equivalent poisson model using independent vague priors $\lambda_j \sim gamma(0.1, 0.1)$. For covariate coeffient we have chosen $beta \sim normal(0, 2)$.

In [281]:
```
survival_model="""
data {
  int<lower=1> N;          // number of individuals
  int<lower=1> N_tot;      // total number of pseudo poisson observatio
  int<lower=1> T;          // number of time intervals
  //int<lower=0> M;        // number of covariates
  int<lower=0, upper=40> base_id[N_tot];  // time interval index for ea

  int<lower=0, upper=1> death_array[N_tot];    // 1 for observed death,
  vector[N_tot] x;                             // covariates

  vector<lower=0>[N_tot] expo;  // exposure time (time alive) in each i
}

transformed data {
  vector[N_tot] log_expo = log(expo);  // log-duration for each timepoi
}
parameters {
  real beta;                       // regression coefficient
  vector<lower=0>[T] lambda0;  // baseline hazard for each timepoint t
}
model {
  beta ~ normal(0, 2);
  lambda0 ~ gamma(0.1,0.1);
  for (n_tot in 1:N_tot) {
    death_array[n_tot] ~ poisson_log(log(lambda0[base_id[n_tot]]) + log_
  }

}
generated quantities {
  vector[N] log_lik;
  int n;
  n = 1;
  log_lik = rep_vector(0,N);

  // log_lik for loo-psis
  for (n_tot in 1:N_tot) {
      log_lik[n] += poisson_log_lpmf(death_array[n_tot]| log(lambda0[ba

      // increment individual count if next time interval comes before
      // only because of bad programming
      if (n_tot > 1){
          if (base_id[n_tot] <= base_id[n_tot-1]){
              n += 1;
          }
      }
  }
}
"""
```

In [282]:
```
sm = pystan.StanModel(model_code=survival_model);
```

INFO:pystan:COMPILING THE C++ CODE FOR MODEL anon_model_0d21c826558aa8
026ccfdecb49f8491e NOW.

In [283]:
```
fit = sm.sampling(data=death_dat3, algorithm="HMC", seed=1, iter=600, cl
```

In [315]:
```
fit
```

Out[315]:
```
Inference for Stan model: anon_model_0d21c826558aa8026ccfdecb49f8491e.
4 chains, each with iter=600; warmup=200; thin=1;
post-warmup draws per chain=400, total post-warmup draws=1600.

               mean se_mean     sd   2.5%    25%    50%    75%  97.5%
n_eff   Rhat
beta          -0.08  4.5e-3   0.18  -0.44   -0.2  -0.08   0.04   0.27
1600   1.01
lambda0[0]     0.01  1.5e-4 2.4e-3 8.6e-3   0.01   0.01   0.01   0.02
262   1.01
lambda0[1]   6.5e-3  4.3e-5 1.7e-3 3.7e-3 5.1e-3 6.3e-3 7.7e-3   0.01
1600   1.01
lambda0[2]   8.4e-3  5.9e-5 2.4e-3 4.4e-3 6.7e-3 8.2e-3 9.8e-3   0.01
1600    1.0
lambda0[3]   8.0e-3  5.8e-5 2.3e-3 4.1e-3 6.4e-3 7.8e-3 9.5e-3   0.01
1600   1.01
lambda0[4]   8.1e-3  6.7e-5 2.7e-3 3.5e-3 6.2e-3 7.9e-3 9.7e-3   0.01
1600   1.03
lambda0[5]   7.9e-3  7.7e-5 3.1e-3 3.1e-3 5.6e-3 7.5e-3 9.6e-3   0.02
1600   1.01
lambda0[6]   8.6e-3  2.4e-4 3.3e-3 3.0e-3 6.2e-3 8.3e-3   0.01   0.02
193   1.01
lambda0[7]   4.8e-3  2.9e-4 2.9e-3 1.0e-3 2.8e-3 4.2e-3 6.2e-3   0.01
103   1.01
lambda0[8]   3.8e-3  6.9e-5 2.8e-3 4.0e-4 1.8e-3 3.1e-3 5.0e-3   0.01
1600   1.01
lambda0[9]   9.0e-3  2.3e-4 4.5e-3 2.2e-3 5.6e-3 8.3e-3   0.01   0.02
398    1.0
lambda0[10]  2.6e-3  1.2e-4 2.5e-3 1.1e-4 7.5e-4 1.8e-3 3.7e-3 9.2e-3
455    1.0
lambda0[11]  9.2e-3  1.3e-4 5.2e-3 1.8e-3 5.3e-3 8.2e-3   0.01   0.02
1600    1.0
lambda0[12]  3.7e-3  8.9e-5 3.6e-3 1.7e-4 1.1e-3 2.6e-3 5.2e-3   0.01
1600    1.0
lambda0[13]  4.0e-3  9.6e-5 3.8e-3 1.1e-4 1.2e-3 2.9e-3 5.7e-3   0.01
1600    1.0
lambda0[14]  4.6e-3  2.2e-4 4.3e-3 1.8e-4 1.5e-3 3.5e-3 6.3e-3   0.02
383    1.0
lambda0[15]    0.02  2.7e-4   0.01 5.8e-3   0.01   0.02   0.03   0.05
1600    1.0
lambda0[16]  8.6e-3  2.1e-4 8.5e-3 2.8e-4 2.2e-3 5.8e-3   0.01   0.03
1600    1.0
lambda0[17]  8.9e-4  6.9e-5 2.8e-38.8e-18 9.6e-9 7.4e-6 3.4e-4 9.2e-3
1600    1.0
lambda0[18]    0.01  2.7e-4   0.01 1.5e-4 2.6e-3 6.8e-3   0.01   0.04
1600    1.0
lambda0[19]  1.2e-3  9.7e-5 3.6e-31.8e-17 9.4e-9 5.9e-6 3.5e-4   0.01
1343    1.0
lambda0[20]  1.0e-3  9.1e-5 3.3e-31.4e-19 1.9e-8 5.6e-6 3.7e-4   0.01
1305    1.0
lambda0[21]  1.1e-3  8.5e-5 3.4e-39.4e-18 2.3e-8 1.4e-5 4.6e-4   0.01
1600    1.0
lambda0[22]    0.02  6.5e-4   0.02 5.3e-4 4.7e-3   0.01   0.02   0.06
568   1.01
lambda0[23]    0.02  4.4e-4   0.02 7.7e-4 6.5e-3   0.01   0.03   0.07
```

```
1600    1.0
lambda0[24]  2.3e-3  1.9e-4 7.4e-34.6e-19 5.5e-9 1.4e-5 7.8e-4   0.02
1452    1.0
lambda0[25]  2.2e-3  2.2e-4 8.0e-32.2e-20 1.1e-8 1.0e-5 5.3e-4   0.02
1261    1.0
lambda0[26]  2.5e-3  2.4e-4 8.3e-31.3e-19 2.0e-9 5.4e-6 6.0e-4   0.03
1166    1.0
lambda0[27]  2.3e-3  1.8e-4 7.0e-33.1e-18 1.3e-8 1.2e-5 6.6e-4   0.03
1512    1.0
lambda0[28]  2.1e-3  1.8e-4 6.5e-39.7e-21 1.3e-8 1.4e-5 8.6e-4   0.02
1239    1.0
lambda0[29]  2.3e-3  2.0e-4 6.8e-34.9e-17 4.2e-8 1.6e-5 7.6e-4   0.02
1224    1.0
lambda0[30]  2.0e-3  1.7e-4 6.5e-32.3e-20 1.7e-8 1.3e-5 6.4e-4   0.02
1490    1.0
lambda0[31]  2.1e-3  2.3e-4 7.0e-31.2e-19 5.2e-9 1.7e-5 8.8e-4   0.02
900    1.0
lambda0[32]  2.2e-3  2.5e-4 7.3e-33.5e-20 1.0e-8 1.9e-5 8.3e-4   0.03
865    1.0
lambda0[33]  1.9e-3  1.7e-4 6.0e-34.7e-16 5.4e-9 9.3e-6 5.5e-4   0.02
1274    1.0
lambda0[34]  2.2e-3  2.2e-4 7.7e-36.0e-19 2.3e-8 1.1e-5 6.9e-4   0.02
1225    1.0
lambda0[35]  2.0e-3  1.7e-4 6.6e-32.4e-20 7.6e-9 1.2e-5 9.6e-4   0.02
1475    1.0
lambda0[36]  2.0e-3  2.0e-4 6.1e-32.4e-17 1.2e-8 1.0e-5 7.1e-4   0.02
917    1.0
lambda0[37]  2.0e-3  1.7e-4 6.3e-35.3e-17 2.4e-8 1.9e-5 7.5e-4   0.02
1367    1.0
lambda0[38]  2.5e-3  2.1e-4 7.4e-36.2e-19 1.5e-8 2.2e-5 9.3e-4   0.02
1252    1.0
lambda0[39]    0.06  9.6e-4   0.04 8.2e-3   0.03   0.05   0.08   0.16
1600    1.0
log_lik[0]    -2.72    0.01   0.27  -3.33  -2.87  -2.69  -2.53  -2.26
329    1.0
log_lik[1]    -6.13    0.04   1.21  -8.97   -6.9  -5.89  -5.22  -4.28
819    1.0
log_lik[2]     -5.5    0.04   0.55  -6.71  -5.84  -5.42   -5.1  -4.62
239    1.0
log_lik[3]    -6.02    0.01   0.41  -6.91  -6.28   -6.0  -5.73  -5.27
1600   1.01
log_lik[4]    -4.91    0.03   0.37  -5.71  -5.13  -4.88  -4.66  -4.25
157   1.02
log_lik[5]    -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[6]    -3.67    0.01   0.28   -4.3  -3.84  -3.64  -3.47  -3.17
564    1.0
log_lik[7]    -3.54    0.02   0.34  -4.32  -3.74  -3.52   -3.3  -2.93
232   1.02
log_lik[8]    -5.76    0.03   1.14  -8.23  -6.45  -5.57  -4.95  -4.05
1600    1.0
log_lik[9]    -1.09  3.6e-3   0.14  -1.39  -1.19  -1.09  -0.99  -0.82
1600   1.03
log_lik[10]   -4.76  7.7e-3   0.31  -5.39  -4.97  -4.75  -4.54   -4.2
1600   1.01
log_lik[11]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
```

```
log_lik[12]   -3.22  9.0e-3   0.36  -4.02  -3.44  -3.19  -2.96  -2.57
1600   1.01
log_lik[13]   -2.29  9.9e-3   0.17  -2.63   -2.4  -2.28  -2.17  -1.98
286   1.01
log_lik[14]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[15]   -3.82  6.6e-3   0.26  -4.34  -4.01   -3.8  -3.63  -3.34
1600   1.01
log_lik[16]   -7.22    0.01   0.56  -8.46  -7.55  -7.17  -6.82  -6.22
1600    1.0
log_lik[17]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[18]   -5.62  8.4e-3   0.34   -6.3  -5.84   -5.6  -5.39  -4.98
1600    1.0
log_lik[19]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[20]   -1.08  3.6e-3   0.14  -1.36  -1.17  -1.07  -0.98  -0.81
1600   1.03
log_lik[21]   -0.87  3.0e-3   0.12  -1.13  -0.95  -0.86  -0.78  -0.65
1600   1.01
log_lik[22]   -5.06    0.04   0.42  -6.03  -5.31  -5.02  -4.78  -4.31
95   1.02
log_lik[23]   -3.18    0.01   0.28  -3.77  -3.35  -3.16   -3.0  -2.71
403    1.0
log_lik[24]   -4.56    0.03   0.36  -5.35  -4.79  -4.54  -4.32  -3.92
158   1.02
log_lik[25]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[26]   -7.39    0.03   0.45  -8.44  -7.66  -7.35  -7.08  -6.64
203   1.01
log_lik[27]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[28]   -4.85  8.1e-3   0.32  -5.51  -5.06  -4.83  -4.63  -4.25
1600    1.0
log_lik[29]   -1.72  9.2e-3   0.15  -2.02  -1.82  -1.71  -1.62  -1.46
258   1.01
log_lik[30]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[31]   -4.81  9.9e-3    0.4  -5.65  -5.06  -4.78  -4.52  -4.12
1600   1.01
log_lik[32]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[33]   -3.64  6.6e-3   0.26  -4.16  -3.84  -3.63  -3.45  -3.17
1600   1.01
log_lik[34]   -4.63    0.01    0.3  -5.24   -4.8  -4.61  -4.43  -4.12
407    1.0
log_lik[35]    -5.9    0.02   0.62  -7.27  -6.27  -5.85  -5.46  -4.88
1600    1.0
log_lik[36]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[37]   -5.31    0.01    0.3  -5.93  -5.49  -5.29  -5.11   -4.8
407    1.0
log_lik[38]   -2.85    0.02   0.31  -3.58  -3.03  -2.82  -2.63  -2.31
356   1.02
log_lik[39]   -4.71  6.7e-3   0.27  -5.24  -4.92   -4.7  -4.52  -4.23
1600   1.01
log_lik[40]   -6.16  8.6e-3   0.35  -6.85  -6.39  -6.15  -5.93  -5.51
```

```
1600    1.0
log_lik[41]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[42]   -2.85    0.01   0.27  -3.42  -3.01  -2.83  -2.66   -2.4
400    1.0
log_lik[43]   -5.92    0.01    0.5  -7.04  -6.21  -5.88  -5.56  -5.04
1600    1.0
log_lik[44]    -2.4    0.01   0.17  -2.75  -2.52   -2.4  -2.29  -2.09
258   1.01
log_lik[45]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[46]   -2.86    0.01   0.26  -3.45  -3.02  -2.84  -2.68   -2.4
568    1.0
log_lik[47]   -3.16  6.4e-3   0.26  -3.66  -3.35  -3.14  -2.97   -2.7
1600   1.01
log_lik[48]   -3.07    0.02   0.32  -3.82  -3.25  -3.04  -2.84  -2.49
296   1.02
log_lik[49]   -5.85    0.01   0.42  -6.73  -6.13  -5.83  -5.55   -5.1
1600   1.01
log_lik[50]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[51]   -6.75    0.03   0.44  -7.76  -7.01  -6.72  -6.44  -6.02
225   1.01
log_lik[52]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[53]   -2.72    0.01   0.26   -3.3  -2.88   -2.7  -2.54  -2.28
569    1.0
log_lik[54]   -6.34    0.07   0.66  -7.83  -6.72  -6.27  -5.87  -5.21
88   1.01
log_lik[55]   -3.75    0.04    0.4  -4.68  -3.98  -3.71  -3.49  -3.06
93   1.02
log_lik[56]   -4.77    0.02    0.8  -6.66  -5.21  -4.67  -4.22   -3.5
1600   1.02
log_lik[57]   -7.34    0.05   1.29 -10.28  -8.08  -7.15  -6.42  -5.23
751    1.0
log_lik[58]   -7.53    0.02   0.66  -8.98  -7.93   -7.5  -7.08   -6.4
1600    1.0
log_lik[59]   -0.32  3.8e-3   0.06  -0.44  -0.36  -0.31  -0.27  -0.21
262   1.01
log_lik[60]   -6.16    0.08   1.24  -8.81  -6.85  -5.99  -5.24  -4.35
243   1.02
log_lik[61]   -4.02    0.06   0.61  -5.42  -4.35  -3.95   -3.6  -3.02
89   1.01
log_lik[62]   -4.39    0.04   0.41  -5.35  -4.64  -4.35  -4.12  -3.66
94   1.02
log_lik[63]   -1.54  5.0e-3    0.2  -1.94  -1.67  -1.53   -1.4  -1.17
1600   1.01
log_lik[64]   -3.26  9.1e-3   0.36  -4.07  -3.48  -3.24   -3.0  -2.61
1600   1.01
log_lik[65]   -4.19    0.02   0.35  -4.99   -4.4  -4.17  -3.95  -3.56
230   1.02
log_lik[66]   -4.69    0.02   0.35   -5.5   -4.9  -4.66  -4.44  -4.04
230   1.02
log_lik[67]   -3.64    0.03   0.47  -4.73  -3.92  -3.56   -3.3  -2.89
242    1.0
log_lik[68]   -2.72    0.01   0.26  -3.28  -2.88  -2.69  -2.54  -2.28
535    1.0
```

```
log_lik[69]   -6.18    0.03   1.06  -8.64  -6.81  -6.05  -5.41  -4.47
1000    1.0
log_lik[70]   -3.38    0.02   0.33  -4.16  -3.58  -3.36  -3.15  -2.79
365   1.02
log_lik[71]   -0.73  4.7e-3   0.11  -0.96   -0.8  -0.72  -0.65  -0.52
591    1.0
log_lik[72]   -1.64    0.01   0.25  -2.17   -1.8  -1.62  -1.46   -1.2
576    1.0
log_lik[73]   -3.83    0.03   0.51  -4.98  -4.13  -3.76  -3.44  -3.01
231    1.0
log_lik[74]   -6.17    0.04   1.06  -8.62  -6.79  -6.05  -5.39  -4.45
864    1.0
log_lik[75]   -7.64    0.01   0.39  -8.48  -7.89  -7.62  -7.37  -6.93
824   1.01
log_lik[76]   -0.29  1.8e-3   0.06  -0.42  -0.33  -0.29  -0.25  -0.19
996    1.0
log_lik[77]   -5.96    0.04   1.19  -8.69  -6.59  -5.82  -5.13  -4.09
748    1.0
log_lik[78]   -4.02    0.01   0.53  -5.19  -4.35  -3.95  -3.62  -3.16
1600    1.0
log_lik[79]   -4.66  7.9e-3   0.32  -5.32  -4.87  -4.65  -4.44  -4.08
1600   1.01
log_lik[80]   -0.44  1.9e-3   0.08  -0.61  -0.49  -0.44  -0.39  -0.31
1600    1.0
log_lik[81]   -6.31    0.08   1.26  -9.09  -6.94  -6.09  -5.46  -4.58
247   1.01
log_lik[82]   -5.71    0.07   1.36  -9.18   -6.4  -5.44  -4.74  -3.81
370   1.01
log_lik[83]  -11.63    0.02   0.84 -13.61 -12.13 -11.53 -11.05 -10.24
1600   1.02
log_lik[84]   -0.29  1.8e-3   0.06  -0.42  -0.33  -0.29  -0.25  -0.19
996    1.0
log_lik[85]   -3.87  6.7e-3   0.27  -4.39  -4.05  -3.86  -3.68  -3.38
1600   1.01
log_lik[86]   -2.62  6.1e-3   0.25   -3.1  -2.78  -2.61  -2.45  -2.18
1600   1.01
log_lik[87]   -6.43    0.02   0.64  -7.84   -6.8  -6.38  -5.98  -5.37
1600    1.0
log_lik[88]   -1.54  4.6e-3   0.14  -1.86  -1.63  -1.53  -1.44  -1.29
976    1.0
log_lik[89]   -0.44  1.9e-3   0.08  -0.61  -0.49  -0.44  -0.39  -0.31
1600    1.0
log_lik[90]   -2.22  9.8e-3   0.16  -2.57  -2.33   -2.2   -2.1  -1.92
280   1.01
log_lik[91]   -2.05  5.5e-3   0.17  -2.42  -2.16  -2.04  -1.93  -1.74
978    1.0
log_lik[92]   -0.29  1.8e-3   0.06  -0.42  -0.33  -0.29  -0.25  -0.19
996    1.0
log_lik[93]   -6.69  8.7e-3   0.35   -7.4  -6.91  -6.68  -6.45  -6.05
1600    1.0
log_lik[94]    -1.7  4.9e-3   0.15  -2.04   -1.8  -1.69  -1.59  -1.43
977    1.0
log_lik[95]    -2.6  5.9e-3   0.18   -3.0  -2.72  -2.59  -2.47  -2.26
980    1.0
log_lik[96]   -1.57  4.7e-3   0.15  -1.89  -1.66  -1.56  -1.47  -1.31
976    1.0
log_lik[97]   -0.29  1.8e-3   0.06  -0.42  -0.33  -0.29  -0.25  -0.19
```

```
996     1.0
log_lik[98]    -4.77  7.9e-3   0.32  -5.43  -4.97  -4.75  -4.55  -4.19
1600     1.0
log_lik[99]    -0.29  1.8e-3   0.06  -0.42  -0.33  -0.29  -0.25  -0.19
996     1.0
log_lik[100]  -2.59     0.01   0.25  -3.17  -2.74  -2.57  -2.41  -2.17
558     1.0
log_lik[101]  -4.28  8.5e-3   0.34  -5.01  -4.49  -4.25  -4.04  -3.66
1600     1.0
log_lik[102]  -0.44  1.9e-3   0.08  -0.61  -0.49  -0.44  -0.39  -0.31
1600     1.0
log_lik[103]  -2.74     0.01   0.26  -3.34   -2.9  -2.72  -2.56   -2.3
556     1.0
log_lik[104]  -4.01     0.01   0.29  -4.66  -4.19  -3.99  -3.81  -3.51
550     1.0
log_lik[105]  -5.36     0.02   0.31  -6.02  -5.54  -5.33  -5.14  -4.82
394     1.0
log_lik[106]  -5.36  8.1e-3   0.32  -6.03  -5.57  -5.35  -5.13  -4.76
1600    1.01
log_lik[107]  -1.78  5.1e-3   0.16  -2.13  -1.89  -1.78  -1.67   -1.5
977     1.0
log_lik[108]  -0.29  1.8e-3   0.06  -0.42  -0.33  -0.29  -0.25  -0.19
996     1.0
log_lik[109]   -0.7  4.5e-3   0.11  -0.91  -0.77  -0.69  -0.62   -0.5
593     1.0
log_lik[110]  -3.69  6.6e-3   0.27  -4.21  -3.88  -3.69  -3.51   -3.2
1600    1.01
log_lik[111]  -5.36     0.02   0.31  -6.02  -5.54  -5.33  -5.14  -4.82
394     1.0
log_lik[112]  -2.99     0.01   0.27  -3.61  -3.15  -2.97   -2.8  -2.53
554     1.0
log_lik[113]  -4.67     0.02    0.3  -5.33  -4.85  -4.64  -4.46  -4.14
394     1.0
log_lik[114]  -4.57  8.5e-3   0.34  -5.32   -4.8  -4.55  -4.34  -3.96
1600     1.0
log_lik[115]  -0.29  1.8e-3   0.06  -0.42  -0.33  -0.29  -0.25  -0.19
996     1.0
log_lik[116]  -3.12  6.4e-3   0.26  -3.62   -3.3  -3.11  -2.94  -2.65
1600    1.01
log_lik[117]  -4.76  8.0e-3   0.32  -5.43  -4.96  -4.75  -4.53  -4.16
1600    1.01
log_lik[118]  -0.29  1.8e-3   0.06  -0.42  -0.33  -0.29  -0.25  -0.19
996     1.0
log_lik[119]  -3.41  9.3e-3   0.37  -4.23  -3.64  -3.39  -3.15  -2.76
1600    1.01
log_lik[120]  -4.64     0.07   0.64   -6.1  -4.99  -4.58  -4.18  -3.57
89    1.0
log_lik[121]   -4.9  8.3e-3   0.33   -5.6  -5.11  -4.89  -4.68  -4.29
1600     1.0
log_lik[122]  -0.29  1.8e-3   0.06  -0.42  -0.33  -0.29  -0.25  -0.19
996     1.0
log_lik[123]  -2.57  6.1e-3   0.24  -3.06  -2.74  -2.56   -2.4  -2.13
1600    1.01
log_lik[124]  -4.01     0.01   0.29  -4.66  -4.19  -3.99  -3.81  -3.51
550     1.0
log_lik[125]  -4.67     0.02    0.3  -5.33  -4.85  -4.64  -4.46  -4.14
394     1.0
```

```
log_lik[126]  -5.03    0.04   0.42  -5.98  -5.29   -5.0  -4.74  -4.32
112   1.01
log_lik[127]  -0.29  1.8e-3   0.06  -0.42  -0.33  -0.29  -0.25  -0.19
996    1.0
log_lik[128]  -5.91  9.3e-3   0.37   -6.7  -6.15  -5.89  -5.65  -5.23
1600    1.0
log_lik[129]  -0.29  1.8e-3   0.06  -0.42  -0.33  -0.29  -0.25  -0.19
996    1.0
log_lik[130]  -2.67  6.2e-3   0.25  -3.15  -2.83  -2.66  -2.49  -2.22
1600   1.01
log_lik[131]  -5.64    0.05   1.21  -8.66  -6.29  -5.41  -4.76  -3.89
716    1.0
log_lik[132]  -3.99  9.7e-3   0.39  -4.84  -4.23  -3.97  -3.71  -3.29
1600   1.01
log_lik[133]  -3.46    0.02   0.33  -4.24  -3.66  -3.44  -3.23  -2.86
365   1.02
log_lik[134]  -4.78    0.04   0.55  -5.99  -5.11   -4.7  -4.37  -3.87
234    1.0
log_lik[135]  -5.34    0.02   0.61  -6.67  -5.73  -5.29  -4.88  -4.31
1600    1.0
log_lik[136]  -2.12  6.0e-3   0.24   -2.6  -2.28  -2.12  -1.95  -1.69
1600   1.01
n             137.0     0.0    0.0  137.0  137.0  137.0  137.0  137.0
1600    nan
lp__         -503.1    0.38   5.38 -515.1 -506.5 -502.8 -499.3 -493.8
200   1.01

Samples were drawn using HMC at Fri May 31 16:38:36 2019.
For each parameter, 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).
```

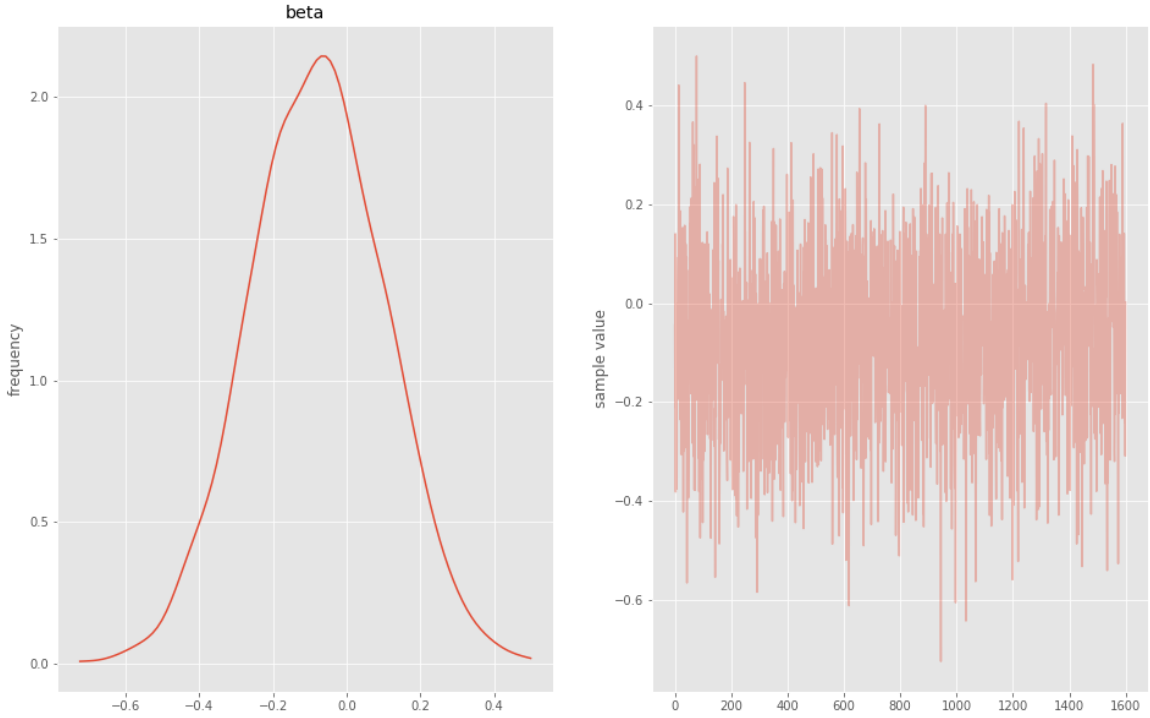
In [ ]:
```python
aa = fit.extract()
```

Conclusion: The test therapy reduces the average hazard by approximately 8%

In [318]:
```python
np.exp(aa['beta'].mean())
```

Out[318]:
```
0.9232192183330878
```

In [319]:
```
fit.traceplot('beta');
```



Psis-loo

In [320]:
```
from psis import psisloo
```

In [321]:
```
log_likelihood1 = aa['log_lik'].reshape(1600,-1)
```

In [322]:
```
loo, loos, ks = psisloo(log_likelihood1)
```

In [323]:
```
np.std(loos)
```

Out[323]: 2.4366107653962406

In [324]:
```
# estimated number of effective parameters
computed_lppd = np.sum(np.log(np.mean(np.exp(log_likelihood1),axis=0)))
pl00cv = computed_lppd - loo;
```

In [325]:
```
computed_lppd
```

Out[325]: -453.6168868124181

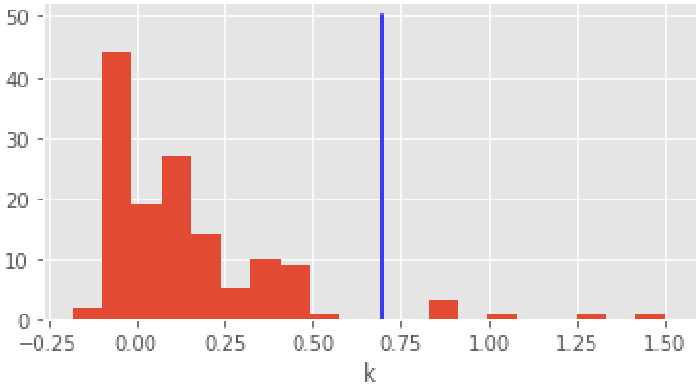
In [326]:
```
pl00cv
```

Out[326]: 31.570530932005227

31 effective parameters

In [403]:
```
plt.figure(figsize=(6,3))
plt.hist(ks, bins=20)
plt.plot([0.7, 0.7],[0,50],"b")
plt.xlabel("k")
```

Out[403]: Text(0.5, 0, 'k')



Plot survival function

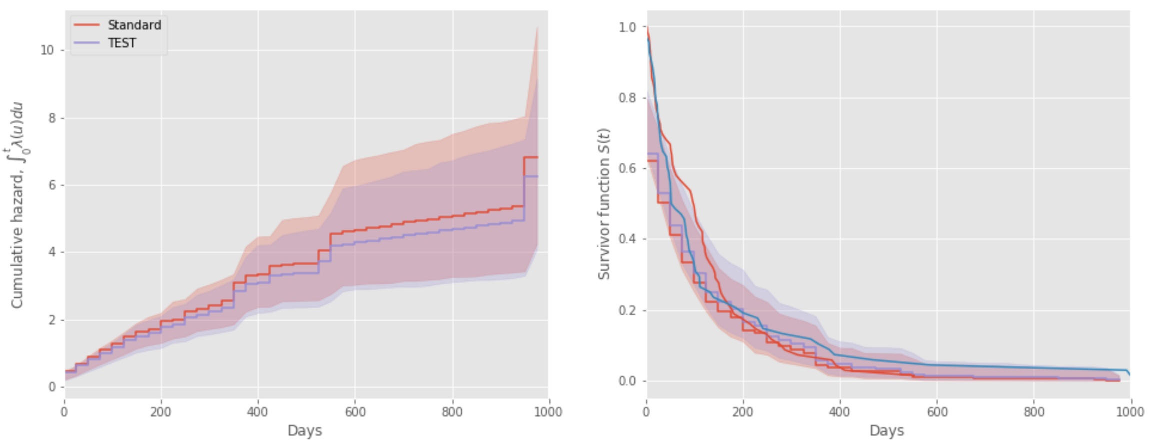
In [330]:
```
base_hazard = aa['lambda0']
base_hazard.shape

met_hazard = aa['lambda0'] * np.exp(np.atleast_2d(aa['beta']).T)
met_hazard.shape
```

Out[330]: (1600, 40)

In [331]:
```
def cum_hazard(hazard):
    return (interval_length * hazard).cumsum(axis=-1)

def survival(hazard):
    return np.exp(-cum_hazard(hazard))
```

In [332]:
```
def plot_with_hpd(x, hazard, f, ax, color=None, label=None, alpha=0.05):
    mean = f(hazard.mean(axis=0))

    percentiles = 100 * np.array([alpha / 2., 1. - alpha / 2.])
    hpd = np.percentile(f(hazard), percentiles, axis=0)

    ax.fill_between(x, hpd[0], hpd[1], color=color, alpha=0.25)
    ax.step(x, mean, color=color, label=label);
```

In [349]:
```
fig, (hazard_ax, surv_ax) = plt.subplots(ncols=2, sharex=True, sharey=Fa

plot_with_hpd(interval_bounds[:-1], base_hazard, cum_hazard,
              hazard_ax, color=blue, label='Standard')
plot_with_hpd(interval_bounds[:-1], met_hazard, cum_hazard,
              hazard_ax, color=red, label='TEST')

hazard_ax.set_xlim(0, df.t.max());
hazard_ax.set_xlabel('Days');

hazard_ax.set_ylabel(r'Cumulative hazard, $\int_0^t \lambda(u)du$');

hazard_ax.legend(loc=2);

plot_with_hpd(interval_bounds[:-1], base_hazard, survival,
              surv_ax, color=blue)
plot_with_hpd(interval_bounds[:-1], met_hazard, survival,
              surv_ax, color=red)
survivalstan.utils.plot_observed_survival(df=df[df['therapy']=='standar
survivalstan.utils.plot_observed_survival(df=df[df['therapy']=='test'],

surv_ax.set_xlim(0, df.t.max());
surv_ax.set_xlabel('Days');

surv_ax.set_ylabel('Survivor function $S(t)$');

#fig.suptitle('Bayesian survival model');
```



Time varying coefficient

In [409]:
```
survival_model_time_varying="""
data {
  int<lower=1> N;          // number of individuals
  int<lower=1> N_tot;      // total number of pseudo poisson observatio
  int<lower=1> T;          // number of time intervals
  //int<lower=0> M;        // number of covariates
  int<lower=0, upper=40> base_id[N_tot];  // time interval index for ea

  int<lower=0, upper=1> death_array[N_tot];    // 1 for observed death,
  vector[N_tot] x;                             // covariates

  vector<lower=0>[N_tot] expo;  // exposure time (time alive) in each i
}

transformed data {
  vector[N_tot] log_expo = log(expo);  // log-duration for each timepoi
  vector[N_tot] xx = 1-x;
}
parameters {
  vector[T] beta;               // regression coefficient
  vector<lower=0>[T] lambda0;  // baseline hazard for each timepoint t
}
model {
  beta ~ normal(0, 2);
  lambda0 ~ gamma(0.1,0.1);
  for (n_tot in 1:N_tot) {
    death_array[n_tot] ~ poisson_log(log(lambda0[base_id[n_tot]])+log_e
  }

}
generated quantities {
  vector[N] log_lik;
  int n;
  n = 1;
  log_lik = rep_vector(0,N);

  // log_lik for loo-psis
  for (n_tot in 1:N_tot) {
      log_lik[n] += poisson_log_lpmf(death_array[n_tot]| log(lambda0[ba

      // increment individual count if next time interval comes before 
      // only because of bad programming
      if (n_tot > 1){
          if (base_id[n_tot] <= base_id[n_tot-1]){
              n += 1;
          }
      }
  }
}
"""
```

In [410]:
```
sm_time = pystan.StanModel(model_code=survival_model_time_varying);
```

```
INFO:pystan:COMPILING THE C++ CODE FOR MODEL anon_model_616dd006a27a29
347e78e7dc99c4700d NOW.
```

In [411]:
```
fit_time = sm_time.sampling(data=death_dat3, algorithm="HMC", seed=1, i
```

In [412]:
```
fit_time
```

Out[412]:
```
Inference for Stan model: anon_model_616dd006a27a29347e78e7dc99c4700d.
4 chains, each with iter=600; warmup=200; thin=1;
post-warmup draws per chain=400, total post-warmup draws=1600.

               mean se_mean     sd   2.5%    25%    50%    75%  97.5%
n_eff   Rhat
beta[0]       -0.15    0.04   0.32  -0.77  -0.37  -0.15   0.07   0.49
76   1.06
beta[1]       -0.76    0.02   0.51  -1.81  -1.09  -0.75  -0.42   0.17
666    1.0
beta[2]       -0.11    0.01    0.5  -1.13  -0.44   -0.1   0.22   0.95
1600   1.01
beta[3]       -0.98    0.02   0.61  -2.21  -1.38  -0.96  -0.57   0.14
1600    1.0
beta[4]        0.38    0.04   0.69  -0.84  -0.11   0.35   0.82   1.79
332    1.0
beta[5]        0.58    0.02   0.79   -0.9   0.05   0.57   1.09   2.32
1600    1.0
beta[6]        1.34    0.03   0.91   -0.3    0.7   1.29   1.92   3.18
867   1.01
beta[7]        0.65    0.03   1.16  -1.69  -0.13   0.63    1.4   3.02
1600    1.0
beta[8]       -0.03    0.03   1.17  -2.33  -0.82  -0.01   0.71   2.34
1600    1.0
beta[9]        0.04    0.02   0.95  -1.85  -0.57   0.04   0.63   1.88
1600    1.0
beta[10]       1.06    0.07   1.62  -2.01  -0.01   1.02    2.1   4.48
538    1.0
beta[11]       0.79    0.03   1.04  -1.23    0.1   0.78   1.48   2.83
1600    1.0
beta[12]        1.3    0.04   1.48  -1.43    0.3   1.26   2.28   4.36
1600    1.0
beta[13]      -1.06    0.04   1.55  -4.44  -2.04  -1.03  -0.03   1.82
1600    1.0
beta[14]      -1.05    0.05   1.55  -4.18  -2.08  -1.01   0.04   1.82
911    1.0
beta[15]       0.17    0.08    1.0  -1.68  -0.56   0.22   0.86   2.21
159   1.02
beta[16]       1.52    0.04   1.56  -1.47   0.49    1.5    2.5   4.65
1600    1.0
beta[17]      -0.04    0.05    2.0  -3.81  -1.44  -0.07    1.4   3.68
1600    1.0
beta[18]      -0.73    0.04   1.55  -3.93  -1.77  -0.65   0.39   2.18
1600    1.0
beta[19]    -1.2e-3    0.05   1.94  -3.85  -1.25  -0.04   1.29   3.88
1600    1.0
beta[20]      -0.13    0.07   1.96   -4.1  -1.39   -0.1   1.17   3.74
721    1.0
beta[21]      -0.54    0.29   1.91  -4.44  -1.72  -0.53   0.63   3.27
43   1.08
beta[22]       2.34    0.06   1.53  -0.73   1.37   2.38   3.38   5.28
682   1.01
beta[23]       0.05     0.1    2.0  -3.72  -1.28   0.02   1.36   3.99
376   1.01
beta[24]       0.04    0.06   2.07  -4.02  -1.36   0.09   1.45    3.9
```

```
1127    1.0
beta[25]      -0.02    0.05   2.09   -4.0  -1.45  -0.03   1.49   3.92
1600    1.0
beta[26]        0.1    0.14   1.84  -3.48  -1.15    0.1   1.43    3.6
179   1.04
beta[27]       0.05    0.07   1.99  -3.89  -1.29   0.03   1.44   3.84
710    1.0
beta[28]       0.01    0.05   1.94  -3.75  -1.29   0.04   1.28   3.99
1600    1.0
beta[29]       0.12     0.1   2.11  -4.09  -1.36   0.13   1.61   4.08
437   1.01
beta[30]       0.04    0.05   1.95  -3.77  -1.26   0.07   1.34   3.96
1600    1.0
beta[31]       0.05    0.09   2.01   -3.9  -1.31   0.07   1.45   3.92
536    1.0
beta[32]       0.14    0.15   2.04  -3.94  -1.17    0.1   1.58   4.09
176    1.0
beta[33]       0.53    0.67   2.02   -3.4  -0.88   0.67   1.95    4.1
9   1.14
beta[34]      -0.17    0.13   1.89  -3.96  -1.36  -0.34   1.07   3.77
219   1.03
beta[35]       0.05    0.08   1.96  -3.87  -1.26   0.07   1.41   3.88
621   1.01
beta[36]       0.26    0.17   2.12  -3.71  -1.16   0.26   1.63   4.55
154   1.04
beta[37]     6.1e-3    0.06   2.28  -4.48  -1.56  -0.02   1.64   4.24
1600    1.0
beta[38]      -0.02    0.05   2.04  -3.93  -1.46  -0.12   1.35   4.13
1600    1.0
beta[39]      -0.03    0.05   1.95  -3.98  -1.29  -0.04   1.27   3.81
1600    1.0
lambda0[0]     0.01  3.4e-4 2.9e-3 8.1e-3   0.01   0.01   0.01   0.02
72   1.06
lambda0[1]   8.6e-3  8.9e-5 2.4e-3 4.5e-3 6.9e-3 8.5e-3   0.01   0.01
757    1.0
lambda0[2]   9.0e-3  8.4e-5 3.3e-3 3.7e-3 6.5e-3 8.6e-3   0.01   0.02
1600   1.01
lambda0[3]     0.01  1.0e-4 4.2e-3 5.0e-3 8.8e-3   0.01   0.01   0.02
1600    1.0
lambda0[4]   7.0e-3  1.9e-4 3.8e-3 1.5e-3 4.2e-3 6.4e-3 9.1e-3   0.02
398    1.0
lambda0[5]   5.6e-3  9.0e-5 3.6e-3 1.1e-3 3.0e-3 4.7e-3 7.4e-3   0.01
1600    1.0
lambda0[6]   3.9e-3  7.7e-5 3.1e-3 4.4e-4 1.7e-3 3.1e-3 5.3e-3   0.01
1600    1.0
lambda0[7]   3.7e-3  8.2e-5 3.3e-3 2.9e-4 1.3e-3 2.7e-3 5.0e-3   0.01
1600    1.0
lambda0[8]   3.7e-3  8.2e-5 3.3e-3 2.8e-4 1.4e-3 2.8e-3 5.0e-3   0.01
1600    1.0
lambda0[9]   8.5e-3  1.4e-4 5.6e-3 1.3e-3 4.4e-3 7.3e-3   0.01   0.02
1600    1.0
lambda0[10]  1.7e-3  1.1e-4 2.4e-3 1.6e-5 2.0e-4 7.5e-4 2.1e-3 8.9e-3
451    1.0
lambda0[11]  6.3e-3  1.3e-4 5.3e-3 5.8e-4 2.5e-3 4.9e-3 8.5e-3   0.02
1600    1.0
lambda0[12]  2.0e-3  7.9e-5 2.7e-3 2.4e-5 3.2e-4 9.7e-4 2.7e-3 9.5e-3
1176    1.0
```

```
lambda0[13]  4.4e-3  1.1e-4 4.3e-3 1.3e-4 1.4e-3 3.1e-3 6.1e-3   0.02
1600    1.0
lambda0[14]  4.9e-3  1.3e-4 5.2e-3 1.4e-4 1.3e-3 3.3e-3 6.5e-3   0.02
1600    1.0
lambda0[15]    0.02  8.4e-4   0.01 2.9e-3   0.01   0.02   0.03   0.05
237   1.01
lambda0[16]  4.5e-3  1.6e-4 6.3e-3 6.8e-5 7.3e-4 2.2e-3 6.0e-3   0.02
1600    1.0
lambda0[17]  6.8e-4  6.3e-5 2.4e-37.4e-19 5.7e-9 4.3e-6 2.7e-4 6.2e-3
1419    1.0
lambda0[18]    0.01  3.0e-4 9.4e-3 5.5e-4 3.5e-3 7.4e-3   0.01   0.03
957   1.01
lambda0[19]  9.0e-4  8.5e-5 3.1e-37.6e-19 1.2e-8 4.7e-6 2.5e-4 9.2e-3
1336    1.0
lambda0[20]  8.6e-4  8.0e-5 3.2e-31.1e-19 8.0e-9 4.4e-6 2.3e-4 8.3e-3
1600    1.0
lambda0[21]  1.1e-3  8.6e-5 3.4e-31.1e-18 2.8e-9 9.5e-6 4.2e-4   0.01
1600    1.0
lambda0[22]    0.01  3.9e-4   0.01 2.9e-4 2.7e-3 6.5e-3   0.01   0.04
753    1.0
lambda0[23]    0.02  4.1e-4   0.02 5.2e-4 5.5e-3   0.01   0.03   0.06
1600    1.0
lambda0[24]  2.0e-3  1.5e-4 6.1e-38.0e-19 2.0e-8 1.6e-5 8.5e-4   0.02
1600    1.0
lambda0[25]  2.1e-3  1.5e-4 6.1e-35.1e-17 1.8e-8 1.6e-5 8.4e-4   0.02
1600    1.0
lambda0[26]  1.9e-3  1.6e-4 6.3e-33.5e-19 2.1e-9 5.8e-6 6.4e-4   0.02
1600    1.0
lambda0[27]  1.9e-3  1.6e-4 5.8e-35.1e-17 5.7e-8 1.5e-5 7.4e-4   0.02
1327    1.0
lambda0[28]  2.2e-3  1.7e-4 6.7e-39.9e-18 1.2e-8 1.4e-5 9.1e-4   0.02
1600    1.0
lambda0[29]  1.9e-3  1.4e-4 5.6e-39.6e-20 2.7e-8 1.9e-5 1.0e-3   0.02
1600    1.0
lambda0[30]  2.2e-3  1.8e-4 6.8e-33.9e-21 4.4e-9 1.4e-5 7.9e-4   0.02
1440    1.0
lambda0[31]  2.1e-3  1.6e-4 6.6e-33.1e-18 2.2e-8 1.3e-5 7.2e-4   0.02
1600    1.0
lambda0[32]  2.3e-3  1.7e-4 7.0e-33.9e-19 8.4e-9 5.7e-6 6.6e-4   0.02
1600    1.0
lambda0[33]  1.8e-3  1.3e-4 5.3e-34.4e-18 2.3e-8 1.5e-5 6.9e-4   0.02
1600    1.0
lambda0[34]  2.0e-3  1.4e-4 5.7e-32.5e-18 3.2e-8 1.8e-5 1.0e-3   0.02
1600    1.0
lambda0[35]  1.8e-3  1.6e-4 5.7e-36.8e-20 6.8e-9 5.6e-6 6.4e-4   0.02
1223    1.0
lambda0[36]  2.3e-3  1.7e-4 6.9e-37.1e-17 9.1e-910.0e-6 9.0e-4   0.02
1600    1.0
lambda0[37]  2.1e-3  1.8e-4 7.0e-31.2e-18 9.5e-8 2.8e-5 1.0e-3   0.02
1600    1.0
lambda0[38]  1.9e-3  2.0e-4 6.5e-33.5e-16 2.1e-8 1.0e-5 6.7e-4   0.02
1074    1.0
lambda0[39]    0.05  1.6e-3   0.03 7.6e-3   0.03   0.05   0.07   0.14
454    1.0
log_lik[0]   -110.4    0.35  13.85 -139.7 -119.4 -109.3 -100.4 -86.14
1600   1.01
log_lik[1]   -271.1    2.25  62.62 -420.4 -307.5 -263.3 -226.1 -169.9
```

```
777    1.0
log_lik[2]   -223.5    0.76  30.32 -290.9 -241.5 -220.9 -202.2 -171.7
1600    1.0
log_lik[3]   -265.3    0.69  27.51 -326.3 -282.3 -263.6 -245.3 -215.3
1600    1.0
log_lik[4]   -211.1    1.26  23.16 -264.6 -224.9 -208.4 -194.9 -172.8
336   1.01
log_lik[5]   -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[6]   -136.4    0.37  14.63 -167.8 -145.3 -135.7 -126.1 -110.6
1600    1.0
log_lik[7]   -158.1    1.33   23.7 -211.2 -171.7 -155.3 -141.2 -119.4
318    1.0
log_lik[8]   -262.0    2.04   63.1 -403.5 -299.4 -253.1 -215.5 -161.4
952    1.0
log_lik[9]   -49.62    0.19   7.46  -64.6 -54.52 -49.46 -44.34 -35.83
1600   1.01
log_lik[10]  -179.5     1.1  12.86 -207.5 -187.4 -178.6 -170.7 -157.3
137   1.03
log_lik[11]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[12]  -151.5    0.63  25.27 -209.4 -166.6 -149.5 -132.5 -107.4
1600    1.0
log_lik[13]  -92.77    0.94   8.03 -110.0 -97.81 -92.25  -87.0 -78.32
73   1.06
log_lik[14]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[15]  -142.1    0.43  11.71 -167.1 -149.4 -141.2 -133.9 -120.8
727    1.0
log_lik[16]  -286.7    1.97  29.69 -357.0 -304.9 -283.5 -265.6 -237.4
227   1.01
log_lik[17]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[18]  -224.3    0.42  16.74 -260.1 -235.2 -223.6 -212.6 -194.2
1600   1.02
log_lik[19]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[20]  -49.06    0.18   7.32 -63.97 -53.87 -48.97  -43.9 -35.52
1600   1.01
log_lik[21]  -41.19    0.16   6.25 -53.67 -45.47 -41.09 -36.91 -29.45
1600   1.01
log_lik[22]  -245.9    1.18  34.35 -320.4 -268.1 -241.7 -221.5 -186.4
845   1.01
log_lik[23]  -128.3    0.36  14.57 -158.5 -138.1 -127.2 -117.6 -103.1
1600    1.0
log_lik[24]  -197.4    1.26  23.11 -250.9 -210.9 -194.6 -181.4 -159.6
336   1.01
log_lik[25]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[26]  -338.6    0.88  35.02 -410.7 -361.1 -335.7 -314.1 -277.6
1600   1.01
log_lik[27]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[28]  -193.8     0.4  16.12 -228.6 -204.3 -192.9 -182.4 -164.9
1600   1.02
log_lik[29]   -68.1    0.85   6.83 -83.06 -72.18 -67.45 -63.19 -56.25
65   1.06
```

```
log_lik[30]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[31]  -214.4    0.67  26.91 -273.7 -230.9 -213.1 -195.8 -166.4
1600   1.01
log_lik[32]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[33]  -135.2    0.43  11.61 -160.0 -142.4 -134.3 -127.0 -114.0
727    1.0
log_lik[34]  -185.9    0.39  15.45 -217.6 -196.4 -185.0 -174.7 -158.7
1600    1.0
log_lik[35]  -251.9    0.91  36.28 -333.9 -274.3 -248.4 -226.3 -191.7
1600    1.0
log_lik[36]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[37]  -213.3    0.39  15.57 -245.2 -223.9 -212.4 -201.9 -185.8
1600    1.0
log_lik[38]  -129.9    1.24  22.11 -180.4 -142.2 -126.9 -113.8 -95.14
316    1.0
log_lik[39]  -177.7    0.44  11.99 -203.3 -185.2 -176.8 -169.3 -155.7
727    1.0
log_lik[40]  -245.9    0.43  17.05 -282.5 -257.0 -245.2 -233.9 -214.9
1600   1.02
log_lik[41]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[42]  -115.0    0.35  14.07 -144.3 -124.4 -113.9 -104.7  -91.0
1600    1.0
log_lik[43]  -234.2    1.73   26.7 -299.3 -250.0 -230.9 -215.4 -191.3
237   1.01
log_lik[44]  -95.28    0.98   7.95 -112.2 -100.1 -94.69 -89.63 -80.98
66   1.06
log_lik[45]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[46]  -105.6    0.33  13.12 -134.4 -113.5 -104.9 -96.37 -82.86
1600   1.01
log_lik[47]  -116.1    0.42  11.25 -140.2 -123.0 -115.2 -108.2 -95.86
726    1.0
log_lik[48]  -138.8    1.28  22.77 -190.4 -151.8 -135.9 -122.4 -102.5
317    1.0
log_lik[49]  -256.8     0.7  28.02 -317.8 -274.0 -255.7 -237.7 -205.3
1600   1.01
log_lik[50]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[51]  -310.6    0.86  34.33 -381.6 -332.5 -307.9 -286.4 -252.0
1600   1.01
log_lik[52]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[53]  -100.6    0.32  12.68 -128.6 -108.2 -99.77 -91.64 -78.71
1600   1.01
log_lik[54]  -271.6    0.97  38.86 -356.9 -296.1 -268.0 -243.4 -205.6
1600    1.0
log_lik[55]  -191.9    1.15  33.41 -265.1 -213.2 -187.3 -167.9 -135.4
839   1.01
log_lik[56]  -194.4    0.97  38.69 -283.7 -218.4 -189.1 -165.8 -133.9
1600    1.0
log_lik[57]  -298.2    2.73  55.52 -424.0 -332.0 -291.9 -259.2 -205.2
415   1.01
log_lik[58]  -318.1    0.95  37.91 -402.5 -341.8 -315.2 -291.7 -252.6
```

```
1600    1.0
log_lik[59]  -13.14    0.34   2.87 -19.26 -14.99  -13.0 -11.18  -8.05
72   1.06
log_lik[60]  -280.3    3.99  67.31 -428.6 -322.9 -270.9 -229.8 -174.1
285   1.01
log_lik[61]  -177.6    0.92  36.63 -260.3 -200.4 -173.9 -150.8 -119.5
1600    1.0
log_lik[62]  -218.7    1.17  34.04 -292.7 -240.6 -214.3 -194.3 -160.2
843   1.01
log_lik[63]  -60.18    0.23   9.04 -79.23 -66.02 -59.66 -54.16 -44.05
1600   1.01
log_lik[64]  -153.5    0.63   25.4 -211.5 -168.7 -151.5 -134.4 -108.8
1600    1.0
log_lik[65]  -184.4    1.36  24.37 -238.6 -198.5 -181.8 -167.1 -144.3
319    1.0
log_lik[66]  -204.3    1.38  24.64 -258.9 -218.6 -201.8 -186.8 -163.5
319    1.0
log_lik[67]  -147.8    0.66  26.52 -209.4 -162.2 -144.8 -129.0 -106.8
1600    1.0
log_lik[68]  -174.3    3.33  92.74 -346.2 -187.8 -155.5 -133.6 -104.6
774    1.0
log_lik[69]   -1569   55.99 1459.0  -4711  -1791  -1204 -854.9 -472.8
679    1.0
log_lik[70]  -404.4   11.52 332.89 -965.8 -438.9 -334.0 -269.2 -192.1
835    1.0
log_lik[71]  -241.7    9.28 265.32 -683.8 -268.7 -188.0 -132.2  -71.9
817    1.0
log_lik[72]  -453.1   17.21 482.91  -1386 -500.9 -347.0 -249.0 -139.4
787    1.0
log_lik[73]  -558.9    18.0 500.02  -1563 -604.9 -446.5 -347.5 -232.7
772    1.0
log_lik[74]   -1478   53.19 1371.7  -4372  -1661  -1150 -814.7 -455.2
665    1.0
log_lik[75]  -491.5     8.2 244.63 -896.8 -515.3 -440.0 -395.3 -339.4
889    1.0
log_lik[76]   -86.4    3.32  92.17 -254.0  -97.6 -66.49  -46.6 -24.09
770    1.0
log_lik[77]   -1065   40.02 939.39  -3135  -1167 -834.6 -630.7 -387.1
551    1.0
log_lik[78]  -751.2   27.66 717.03  -2272 -814.2 -586.1 -436.6 -283.3
672    1.0
log_lik[79]  -247.3    3.39 106.69 -437.1 -255.8 -225.8 -206.1 -180.1
989    1.0
log_lik[80]  -144.3    5.85 156.29 -413.1 -161.3 -110.3 -77.94 -42.06
715    1.0
log_lik[81]  -753.1   22.99 637.17  -2059 -817.9 -615.8 -478.0 -319.0
768    1.0
log_lik[82]  -915.1   34.19 830.43  -2697 -999.3 -716.7 -540.6 -346.2
590    1.0
log_lik[83]  -736.9   12.82 369.66  -1469 -768.4 -662.4 -586.7 -495.2
831    1.0
log_lik[84]   -86.4    3.32  92.17 -254.0  -97.6 -66.49  -46.6 -24.09
770    1.0
log_lik[85]  -219.4    3.46 101.95 -399.5 -231.6 -198.5 -177.7 -150.3
869    1.0
log_lik[86]  -198.5    4.77 136.95 -435.5 -211.8 -169.0 -143.0 -113.1
824    1.0
```

```
log_lik[87]  -674.0   18.44  510.3  -1776 -712.6 -556.2 -461.2 -334.5
766    1.0
log_lik[88]  -126.1    2.97  88.54 -284.5 -133.9 -106.1 -90.88  -75.0
888    1.0
log_lik[89]  -144.3    5.85 156.29 -413.1 -161.3 -110.3 -77.94 -42.06
715    1.0
log_lik[90]  -320.9   10.27  299.2 -822.7 -348.1 -257.1 -198.6 -137.8
849    1.0
log_lik[91]  -110.7    1.42  45.18 -189.1 -114.1 -102.0 -93.62 -81.89
1014    1.0
log_lik[92]   -86.4    3.32  92.17 -254.0  -97.6 -66.49  -46.6 -24.09
770    1.0
log_lik[93]  -377.9     6.0 179.69 -667.7 -389.9 -340.2 -309.8 -270.1
897    1.0
log_lik[94]  -117.7    2.32  70.34 -241.7 -123.3 -102.8 -90.54 -76.68
916    1.0
log_lik[95]  -114.8     0.8  24.72 -155.9 -119.9 -111.1 -104.0 -92.73
961    1.0
log_lik[96]  -124.2    2.84  84.89 -275.7 -131.6 -105.3 -90.76 -75.35
893    1.0
log_lik[97]   -86.4    3.32  92.17 -254.0  -97.6 -66.49  -46.6 -24.09
770    1.0
log_lik[98]  -353.1    8.07 240.28 -746.8 -369.7 -302.1 -260.3 -212.4
887    1.0
log_lik[99]   -86.4    3.32  92.17 -254.0  -97.6 -66.49  -46.6 -24.09
770    1.0
log_lik[100] -326.6   10.42 302.43 -842.6 -353.0 -262.9 -203.3 -140.8
842    1.0
log_lik[101] -299.8    6.18 180.31 -595.7 -315.1 -260.7 -229.7 -194.9
851    1.0
log_lik[102] -144.3    5.85 156.29 -413.1 -161.3 -110.3 -77.94 -42.06
715    1.0
log_lik[103] -321.4    9.96 288.83 -809.2 -346.1 -261.1 -204.6 -142.8
841    1.0
log_lik[104] -329.8    8.24 238.72 -726.9 -350.4 -279.0 -233.8 -180.0
839    1.0
log_lik[105] -332.0    5.71 156.68 -598.7 -349.5 -298.5 -266.7 -230.8
753    1.0
log_lik[106] -263.9    2.93  93.43 -422.7 -271.8 -246.1 -228.0 -201.9
1016    1.0
log_lik[107] -115.0    2.07   63.1 -226.2 -120.0 -101.7 -90.89 -77.63
929    1.0
log_lik[108]  -86.4    3.32  92.17 -254.0  -97.6 -66.49  -46.6 -24.09
770    1.0
log_lik[109] -229.2    8.82 251.79 -649.4 -254.1 -177.8 -125.7 -68.04
815    1.0
log_lik[110] -214.5    3.55 104.38 -400.5 -226.7 -193.1 -171.7 -144.8
865    1.0
log_lik[111] -332.0    5.71 156.68 -598.7 -349.5 -298.5 -266.7 -230.8
753    1.0
log_lik[112] -317.2    9.38 271.95 -777.6 -339.5 -261.2 -207.2 -148.3
840    1.0
log_lik[113] -306.7    5.79 159.26 -576.1 -324.6 -272.5 -241.0 -204.2
757    1.0
log_lik[114] -285.8    5.14 153.94 -528.7 -295.5 -253.8 -227.8 -198.9
897    1.0
log_lik[115]  -86.4    3.32  92.17 -254.0  -97.6 -66.49  -46.6 -24.09
```

```
770    1.0
log_lik[116] -201.8    4.01 116.71 -407.5 -213.7 -177.5 -154.3 -127.1
847    1.0
log_lik[117] -228.5    2.62  81.59 -367.0 -234.8 -213.7 -197.7 -175.9
973    1.0
log_lik[118]  -86.4    3.32  92.17 -254.0  -97.6 -66.49  -46.6 -24.09
770    1.0
log_lik[119] -452.5   13.64 386.04  -1101 -487.7 -373.0 -295.9 -209.2
801    1.0
log_lik[120] -540.7   15.71 440.83  -1347 -580.1 -450.5 -357.7 -252.3
787    1.0
log_lik[121] -348.7     7.7 228.19 -718.1 -364.4 -298.9 -261.2 -214.2
879    1.0
log_lik[122]  -86.4    3.32  92.17 -254.0  -97.6 -66.49  -46.6 -24.09
770    1.0
log_lik[123] -198.7    4.87 139.52 -441.0 -212.6 -168.7 -142.2 -112.1
822    1.0
log_lik[124] -329.8    8.24 238.72 -726.9 -350.4 -279.0 -233.8 -180.0
839    1.0
log_lik[125] -306.7    5.79 159.26 -576.1 -324.6 -272.5 -241.0 -204.2
757    1.0
log_lik[126] -533.2   13.97 394.89  -1203 -564.8 -450.2 -375.9 -284.3
799    1.0
log_lik[127]  -86.4    3.32  92.17 -254.0  -97.6 -66.49  -46.6 -24.09
770    1.0
log_lik[128] -318.5    4.27 124.01 -527.5 -329.3 -293.3 -269.5 -237.2
845    1.0
log_lik[129]  -86.4    3.32  92.17 -254.0  -97.6 -66.49  -46.6 -24.09
770    1.0
log_lik[130] -198.3    4.67  134.4 -431.2 -211.7 -169.6 -143.9 -114.4
827    1.0
log_lik[131] -702.9   22.36 613.28  -1945 -754.7 -571.7 -443.8 -296.5
752    1.0
log_lik[132] -467.3   13.23 375.31  -1111 -503.4 -389.8 -313.4 -228.1
805    1.0
log_lik[133] -406.0   11.43 330.76 -965.5 -441.5 -335.3 -271.2 -195.0
837    1.0
log_lik[134] -575.6   17.11  480.2  -1491 -620.7 -471.1 -372.7 -264.3
788    1.0
log_lik[135] -754.1   24.52 666.66  -2065 -814.7 -606.1 -464.7 -320.9
739    1.0
log_lik[136] -114.3    2.01  61.27 -219.2 -117.5 -102.1  -92.5 -79.08
929    1.0
n             137.0     0.0    0.0  137.0  137.0  137.0  137.0  137.0
1600    nan
lp__         -515.4    0.46   6.94 -530.0 -520.2 -515.0 -510.5 -503.0
229   1.02

Samples were drawn using HMC at Fri May 31 18:57:43 2019.
For each parameter, 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).
```

In [413]:

```
aa_time = fit_time.extract()
```

In [414]:
```
log_likelihood2 = aa_time['log_lik'].reshape(1600,-1)
```

In [415]:
```
loo2, loos2, ks2 = psisloo(log_likelihood2)
```

k-value are larger than 0.7 and psis-loo estimate is therefore not applicable.

In [420]:
```
ks2[0:10]
```

Out[420]:
```
array([ 6.29725778, 27.93357033, 12.88923513, 10.90342463, 10.2895348
8,
        1.59305007,  6.98627851, 10.77669931, 31.45997688,  2.6041877
5])
```

In [417]:
```
base_hazard_t = aa_time['lambda0']
base_hazard_t.shape

met_hazard_t = aa_time['lambda0'] * np.exp(np.atleast_2d(aa_time['beta'
met_hazard.shape
```

Out[417]:
```
(1600, 40)
```

In [419]:

```
fig, (hazard_ax, surv_ax) = plt.subplots(ncols=2, sharex=True, sharey=Fa

plot_with_hpd(interval_bounds[:-1], met_hazard_t, cum_hazard,
              hazard_ax, color=blue, label='Standard')
plot_with_hpd(interval_bounds[:-1], base_hazard_t, cum_hazard,
              hazard_ax, color=red, label='TEST')


hazard_ax.set_xlim(0, df.t.max());
hazard_ax.set_ylim(0, 11);

hazard_ax.set_xlabel('Days');

hazard_ax.set_ylabel(r'Cumulative hazard, $\int_0^t \lambda(u)du$');

hazard_ax.legend(loc=2);

plot_with_hpd(interval_bounds[:-1], base_hazard_t, survival,
              surv_ax, color=red)
plot_with_hpd(interval_bounds[:-1], met_hazard_t, survival,
              surv_ax, color=blue)
survivalstan.utils.plot_observed_survival(df=df[df['therapy']=='standar
survivalstan.utils.plot_observed_survival(df=df[df['therapy']=='test'],

surv_ax.set_xlim(0, df.t.max());
surv_ax.set_xlabel('Days');

surv_ax.set_ylabel('Survivor function $S(t)$');

#fig.suptitle('Bayesian survival model');
```