

# Survival Analysis

## General Principles

Survival analysis studies the time until an event of interest (e.g., death, recovery, information acquisition) occurs. When analyzing binary survival outcomes (e.g., alive or dead), we can use models such as Cox proportional hazards to evaluate the effect of predictors on survival probabilities.

Key concepts include:

1. **Hazard Function:** The instantaneous risk of the event occurring at a given time.
2. **Survival Function:** The probability of surviving beyond a given time.
3. **Covariates:** Variables (e.g., age, treatment) that may affect survival probabilities.
4. **Baseline Hazard:** The hazard when all covariates are zero, which forms the reference for comparing different conditions.

## Considerations

### Note

- Bayesian models provide a framework to account for uncertainty in parameter estimates through posterior distributions. You will need to define prior distributions for all model parameters, such as baseline hazard, covariate effects, and variance terms.
  - In survival analysis:
    - The **baseline hazard** can follow distributions like Exponential, Weibull, or Gompertz, depending on the data.
    - Censoring (when the event is not observed for some subjects) must be accounted for in the likelihood function. Proper handling is essential for unbiased results.
  - Bayesian survival models allow flexible handling of time-dependent covariates, ran-

dom effects, and incorporate uncertainty more naturally than Frequentist methods.

## Example

Here's an example of a Bayesian survival analysis using the **Bayesian Inference (BI)** package. The data come from a clinical trial of mastectomy for breast cancer. The goal is to estimate the effect of the `metastasized` covariate, coded as 0 (no metastasis) and 1 (metastasis), on the survival outcome `event` for each patient. Time is continuous and censoring is indicated by the `event` variable.

## Python

```
from BI import bi
import numpy as np
import jax.numpy as jnp
# Setup device-----
m = bi(platform='cpu')

# Import Data & Data Manipulation -----
# Import
from importlib.resources import files
data_path = files('BI.resources.data') / 'mastectomy.csv'
m.data(data_path, sep=',',)

m.df.metastasized = (m.df.metastasized == "yes").astype(np.int64)
m.df.event = jnp.array(m.df.event.values, dtype=jnp.int32)

## Create survival object
m.models.survival.surv_object(time='time', event='event', cov='metastasized', interval_length=1)

# Plot censoring -----
m.models.survival.plot_censoring(cov='metastasized')

# Model -----
def model(intervals, death, metastasized, exposure):
    # Parameter prior distributions-----
    ## Base hazard distribution
    lambda0 = m.dist.gamma(0.01, 0.01, shape= intervals.shape, name = 'lambda0')
    ## Covariate effect distribution
    beta = m.dist.normal(0, 1000, shape = (1,), name='beta')
```

```

#### Likelihood
##### Compute hazard rate based on covariate effect
lambda_ = m.models.survival.hazard_rate(cov = metastasized, beta = beta, lambda0 = lambda0)
##### Compute exposure rates
mu = exposure * lambda_

# Likelihood calculation
y = m.dist.poisson(mu + jnp.finfo(mu.dtype).tiny, obs = death)

# Run mcmc -----
m.fit(model, num_samples=500)

# Summary -----
print(m.summary())

# Plot hazards and survival function -----
m.models.survival.plot_surv()

jax.local_device_count 16

0%|          0/1000 [00:00<?, ?it/s] warmup: 0%|          1/1000 [00:01<19:57, 1.20s]

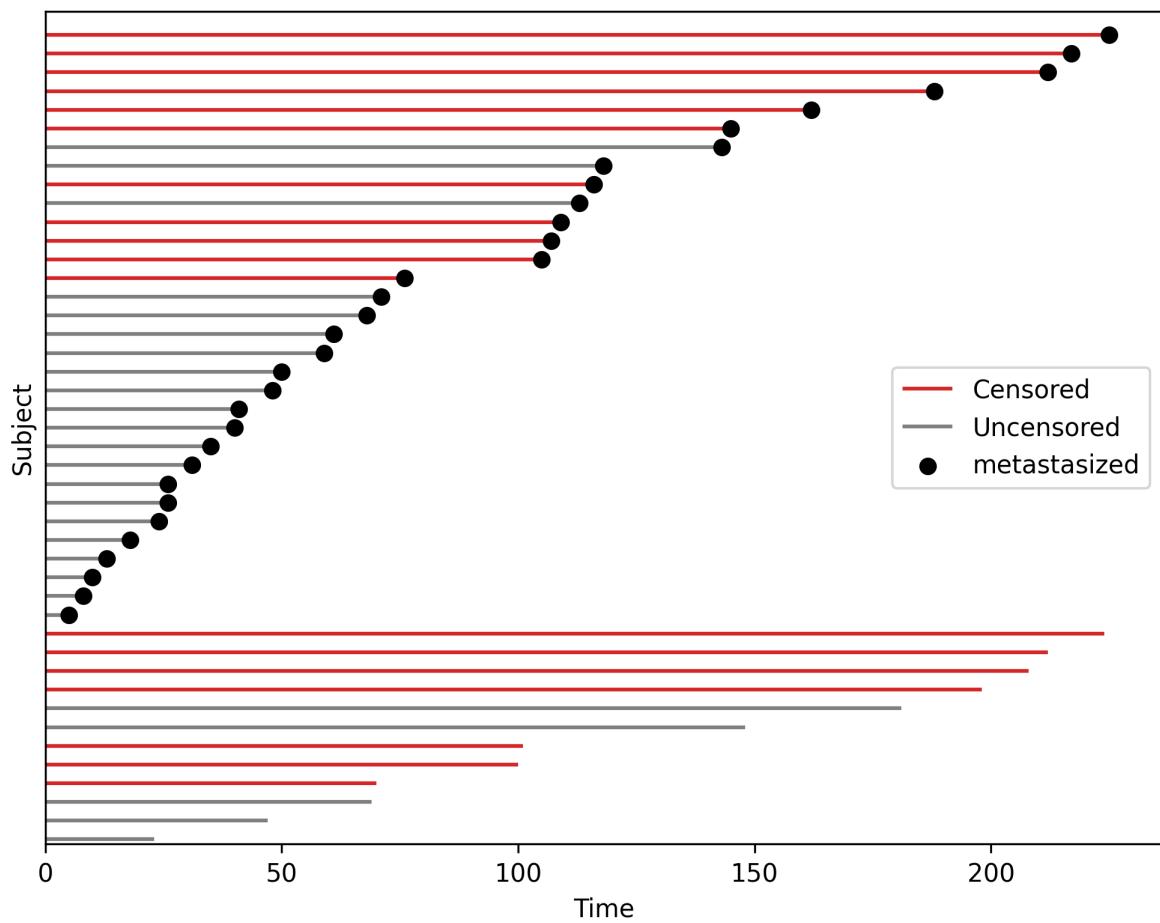
          mean      sd  hdi_5.5%  hdi_94.5%
beta[0]    0.77    0.48      0.0     1.49
lambda0[0]  0.00    0.00      0.0     0.00
lambda0[1]  0.00    0.00      0.0     0.01
lambda0[2]  0.00    0.01      0.0     0.01
lambda0[3]  0.00    0.01      0.0     0.01
...
lambda_[43, 71] 0.00    0.01      0.0     0.00
lambda_[43, 72] 0.00    0.02      0.0     0.00
lambda_[43, 73] 0.00    0.02      0.0     0.00
lambda_[43, 74] 0.00    0.01      0.0     0.00
lambda_[43, 75] 0.00    0.03      0.0     0.00

[3421 rows x 4 columns]

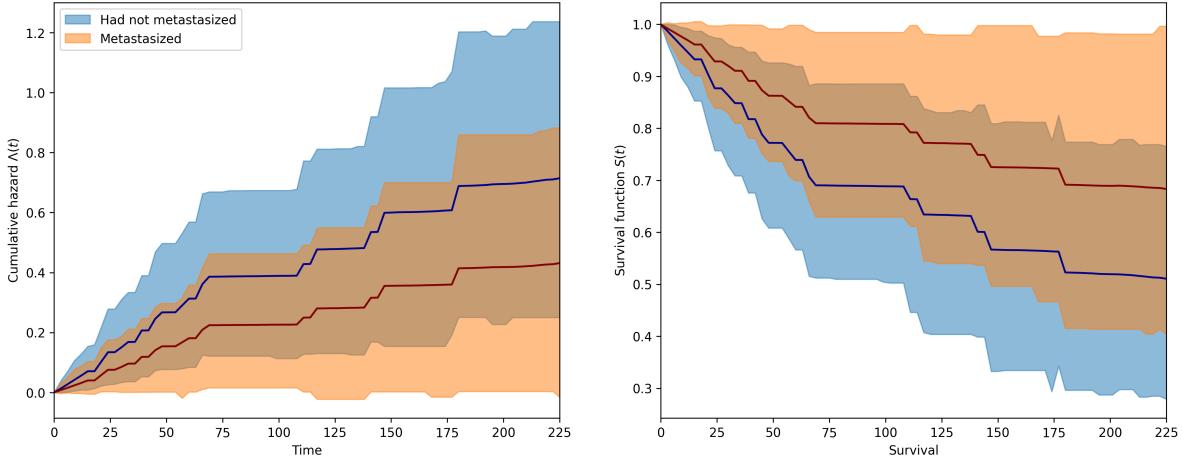
/home/sosa/work/.venv/lib/python3.12/site-packages/arviz/plots/hdiplot.py:166: FutureWarning
  hdi_data = hdi(y, hdi_prob=hdi_prob, circular=circular, multimodal=False, **hdi_kwargs)
/home/sosa/work/.venv/lib/python3.12/site-packages/arviz/plots/hdiplot.py:166: FutureWarning
  hdi_data = hdi(y, hdi_prob=hdi_prob, circular=circular, multimodal=False, **hdi_kwargs)

```

```
/home/sosa/work/.venv/lib/python3.12/site-packages/arviz/plots/hdiplot.py:166: FutureWarning  
    hdi_data = hdi(y, hdi_prob=hdi_prob, circular=circular, multimodal=False, **hdi_kwargs)  
/home/sosa/work/.venv/lib/python3.12/site-packages/arviz/plots/hdiplot.py:166: FutureWarning  
    hdi_data = hdi(y, hdi_prob=hdi_prob, circular=circular, multimodal=False, **hdi_kwargs)
```



Bayesian survival model



## Mathematical Details

The model is defined as follows:

$$dN_i(t) \sim \text{Poisson}(\lambda_i(t)Y_i(t)dt)$$

$$\lambda_i(t) = \lambda_0(t) \exp(X_i\beta)$$

The hierarchical priors for the regression coefficients are:

$$\beta \sim \text{Normal}(\mu_\beta, \sigma_\beta^2)$$

$$\mu_\beta \sim \text{Normal}(0, 100)$$

$$\sigma_\beta^2 \sim \text{InverseGamma}(0.1, 0.1)$$

- Where:

- $N_i(t)$  is the counting process for subject  $i$ , which counts the number of observed events up to time  $t$ . For survival analysis, this is typically 0 or 1.
- $dN_i(t)$  is the increment of the process over a small interval  $dt$ , indicating if an event occurred for subject  $i$  at time  $t$ .

- $Y_i(t)$  is the **at-risk indicator**, taking a value of 1 if subject  $i$  is under observation and has not yet experienced an event just prior to time  $t$ , and 0 otherwise. This indicator is the mechanism that handles censoring:
  - \* If a subject is **censored** at time  $t'$ , their at-risk indicator  $Y_i(t)$  remains 1 up to  $t'$ , signifying they were at risk during this period.
  - \* At the moment of censoring  $t'$ , no event is recorded (the counting process  $N_i(t)$  does not increment).
  - \* For all subsequent times  $t > t'$ , the indicator  $Y_i(t)$  switches to 0, effectively removing the individual from the risk set for any future calculations.
- $\lambda_i(t)$  is the hazard rate for subject  $i$  at time  $t$ .
- $\lambda_0(t)$  is the baseline hazard rate function at time  $t$ . A key assumption is that this baseline hazard is the same for all subjects.
- $X_i$  is the covariates for subject  $i$ .
- $\beta$  is the regression coefficients.

- **Priors:**

- We assign prior distributions to the unknown parameters. The regression coefficients  $\beta$  are given a Normal prior.
- The hyperparameters of the  $\beta$  prior,  $\mu_\beta$  and  $\sigma_\beta^2$ , are themselves given vague priors to be learned from the data.
- The baseline hazard  $\lambda_0(t)$  is also treated as an unknown parameter and is often modeled non-parametrically, for instance, using a gamma process or as a piecewise constant function, where priors are placed on the hazard level in each time interval.

The key assumption of this model is that the hazard ratios are constant over time. Censoring is typically handled in the likelihood function used for estimation, not by multiplying the hazard function by a factor. The data for each subject  $i$  is represented by a tuple  $(t_i, \delta_i, X_i)$ , where  $t_i$  is the observed time (either event or censoring time),  $\delta_i$  is an event indicator (1 if the event was observed, 0 if censored), and  $X_i$  is the vector of covariates.

## Reference(s)

[https://en.wikipedia.org/wiki/Proportional\\_hazards\\_model](https://en.wikipedia.org/wiki/Proportional_hazards_model) <https://www.mathworks.com/help/stats/cox-proportional-hazard-regression.html> [https://www.pymc.io/projects/examples/en/latest/survival\\_analysis/surv.html](https://www.pymc.io/projects/examples/en/latest/survival_analysis/surv.html)  
[https://vflores-io.github.io/posts/20240924\\_numpyro\\_logreg\\_surv\\_analysis/](https://vflores-io.github.io/posts/20240924_numpyro_logreg_surv_analysis/)