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 until 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**
>
> - 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, random 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_lengt

# 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 = lambda
```

```
    #### 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()
```

R

Mathematical Details

Reference(s) https://en.wikipedia.org/wiki/Proportional_hazards_model https://www.mathworks.com/he
proportional-hazard-regression.html https://www.pymc.io/projects/examples/en/latest/survival_analysis/surv
https://vflores-io.github.io/posts/20240924_numpyro_logreg_surv_analysis/np01_logreg_surv_analysis/