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 **BayesForge (BF)** 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 BayesForge import bf
import numpy as np
import jax.numpy as jnp
# Setup device-----------------------------------------------
m = bf(platform='cpu')

# Import Data & Data Manipulation ----------------------------------------------
# Import
from importlib.resources import files
data_path = m.load.mastectomy(only_path=True)
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, progress_bar=False)

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

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

R

Mathematical Details

Bayesian formulation

The BF survival model uses a **piecewise-constant hazard** (Poisson counting-process) approach. The continuous follow-up time is divided into K fixed intervals of length Δ. Each subject i in interval k contributes:

$$N_{ik} \sim \text{Poisson}(\mu_{ik})$$

$$\mu_{ik} = \lambda_{ik} \cdot e_{ik}$$

$$\lambda_{ik} = \lambda_0^{(k)} \exp(\beta X_i)$$

$$\lambda_0^{(k)} \sim \text{Gamma}(0.01,\ 0.01)$$

$$\beta \sim \text{Normal}(0,\ 1000)$$

Where:

- N_{ik} is the number of events (0 or 1) recorded for subject i in interval k.
- e_{ik} is the **exposure** (time at risk) for subject i in interval k. It equals Δ if the subject is fully observed in the interval, a fractional value if the subject exits (via event or censoring) mid-interval, and 0 if the subject has already left the risk set. This term is the mechanism that handles censoring: once a subject is censored or experiences the event, their exposure drops to 0 in all subsequent intervals.
- $\mu_{ik} = \lambda_{ik} \cdot e_{ik}$ is the expected number of events, i.e., the hazard rate scaled by time at risk.
- λ_{ik} is the hazard rate for subject i in interval k, decomposed into a **baseline hazard** $\lambda_0^{(k)}$ and a covariate-specific multiplicative shift.
- $\lambda_0^{(k)}$ is the **baseline hazard** in interval k, constant within each interval but free to vary across intervals. This gives the model non-parametric flexibility over time. A $\text{Gamma}(0.01, 0.01)$ prior places minimal information on the baseline hazard.
- X_i is the vector of covariates for subject i.
- β is the vector of regression coefficients. The coefficient $\exp(\beta)$ gives the **hazard ratio**: the multiplicative change in hazard for a unit increase in the corresponding covariate. A wide $\text{Normal}(0, 1000)$ prior reflects minimal prior knowledge.

Survival and hazard functions

From the posterior samples of $\lambda_0^{(k)}$ and β, two key quantities can be derived:

Cumulative hazard up to interval K:

$$\Lambda_i(t_K) = \sum_{k=1}^{K} \lambda_0^{(k)} \exp(\beta X_i)\, \Delta_k$$

Survival function:

$$S_i(t_K) = \exp(-\Lambda_i(t_K))$$

Where Δ_k is the width of interval k. ## Reference(s) https://en.wikipedia.org/wiki/Proportional_hazards_mod
https://www.mathworks.com/help/stats/cox-proportional-hazard-regression.html https://www.pymc.io/projec
https://vflores-io.github.io/posts/20240924_numpyro_logreg_surv_analysis/np01_logreg_surv_analysis/