Lecture 3: Modelling Techniques

Caden Hewlett

2024-03-21

Data Collection Mechanisms

Topics: Generalizing the censoring problem to other data collection mechanisms, and examples of other data collection mechanisms.

Last time we covered Censoring and custom distributions. Censoring is critical in survival limits. Today we cover some other data collection mechanisms, which is cool! It's important to recognize that it is simply a special (but important) example of probabilistic modelling and the first step of the "Bayesian Recipe."

Truncation

In censoring we knew how many H_is were above the detection limit.

In **truncation**, a different step, we now have even less information! We only observe the H_is that are below the limit! We don't know how many exceeded the limit (like in censoring.) So it's essentially Censoring, but with even less info. So instead of capping the observations, we throw them away.

Mathemtatically, when the H_is have a continuous distribution, this can be modelled as:

$$\begin{aligned} X &\sim \text{prior}(\dots) \\ H_i &\sim \text{likelihood}(X) \\ I_i &= \mathbb{1}[H_i \leq L] \\ Y &= \{H_i : I_i = 1\} \end{aligned}$$

Example: CRISPR-Cas9 unique molecular identifier family sizes. The "family of cells" that left zero progenies are not observed! More down-to-earth, a bad data collection process mis-identifies outliers and you still want to quantify the original data.

Non-Ignorabble Missingness

Truncation can be generalized as follows:

Instead of a deterministic criterion based on H_i to decide if to include it in hte set of observations or not, we make the decision based on some probability model p that could depend on h_i, and x for $p(x, h_i) \in [0, 1]$.

$$\begin{aligned} X &\sim \text{prior}(\dots) \\ H_i &\sim \text{likelihood}(X) \\ I_i &= \text{bern}(p(X, H_i)) \\ Y &= \{H_i : I_i = 1\} \end{aligned}$$

Question: How would you set $p(x, h)$ to recover truncation as a special case of non-ignorable missingness? This week's reading (Ch.8 of Gelman et al.) has some cool examples on this.

Rao-Blackwellization

It can speed up MCMC considerably! In languages that do not support discrete latent variables (such as Stan) this is the only way to implement certain models (such as mixture models.)

Stan Demo

We revisit the Chernobyl example from last class, but this time we implement it in Stan **differently** to support **Rao-Blackwellization**.

```
require(rstan, quietly = TRUE)
```

We see in lecture (didn't want to run it here) that this model gives essentially the same result, but is about 4x faster! What is `exponential_lccdf`?

Mathematical Underpinnings

Now, we consider a simplified example where there is only one observation.

$$
\begin{aligned}
X &\sim \exp(1/100) \\
H &\sim \exp(X) \\
C &= \mathbb{1}[H \geq L] \\
Y &= CL + (1 - C)H
\end{aligned}
$$

Suppose our one observation is censored ($C = 1$)

The first stan model we implemented on Tuesday targets a distribution over X and H, i.e. $\gamma(x, h) = p(x, h, y)$.

The **KEY IDEA** of Rao-Blackwellization is to a target over x only, i.e. $\gamma(x)$, such that the results are the same. This is because we don't care about h - it's some "nuisance variable."

So, the best approach to define $\gamma(x)$ from $\gamma(x, h)$ is to compute the marginal density, i.e. $\int \gamma(x, h) \mathrm{d}h$. This guarantees consistency.

$$
\int \gamma(x, h) \mathrm{d}h = \int \underbrace{f(x, h, y)}_{\text{joint density}} \mathrm{d}h = f(x, y) = \gamma(h)
$$

So in our specific eample, we can actually write out γ.

First, write out $\gamma(x, h)$. Since $y = L$, the detection limit:

$$
\begin{aligned}
\gamma(x) &= \int \gamma(x, h) \mathrm{d}h \\
\gamma(x) &= \int f(x, h, y) \mathrm{d}h \\
\gamma(x) &= \int f(x) f(h \mid x) f(y \mid h) \mathrm{d}h \\
\gamma(x) &= f(x) \int f(h \mid x) \mathbb{1}[L \leq h] \mathrm{d}h \\
\gamma(x) &= f(x) \big(1 - F(L; x)\big)
\end{aligned}
$$

Where we have one *minus* the CDF, because we consider all L below h. In Stan, we take the logarithm. But also, this is the **complement** of the cdf (below), which is `exponential_lccdf` standing for "exponential log-complement CDF." Makes sense!

If you have multiple workers (or units), we need to add it that many times. This is why we have target defined as `+= n_above_limit * exponential_lccdf(limit | rate)`.

A bit more detail on the above calculation: relies on **conditional independence**: If V and W are conditionally independent given $Z\ldots$

$$f(y \mid h) \stackrel{?}{=} f(y \mid h, x)$$

Recall… if $y \perp x$,

$$\begin{aligned} f(x, h, y) &= f(x)f(h \mid x)f(y \mid x, h) \\ &= f(x)f(h \mid x)f(y \mid h) \end{aligned}$$

In a graphical sense, we have $x \to h \to y$