Readme

Each of the folders labelled `Model#` contains an R file and a stan model file. The R file simulates data according to hazard rate with known parameters, and then the stan file tries to fit the model and recover these parameters.

Coding up complicated survival models in the MCMC languages is error-prone. Even if the code executes successfully, it may be giving erroneous output. Establishing the source these errors is time-consuming, as you have to inspect the code line by line, usually by building smaller, simpler models to see where the code fails. After locating and fixing the error, you reintroduce the desired complexity until you arrive at the model you desire.

Another approach is to build a selection of simpler models, while validating them using simulated data. These models can then be used as a stepping stone for more complicated models.

The model files in this repository serve as a proof of concept to get survival models running in Stan. They start with a simple model and progress to more complicated models thereafter. They also serve as an audit trial of the model development process.

Running the code

Each of the `Model#` folders contain the files `simulate_and_estimate.R` and `mystanmodel.stan`. The depend on the R library `rstan`, which can be installed from CRAN. To run the model 1, for instance, we would start R and make the `Model1/` our working directory. Then, either run `source("simulate_and_estimate.R")` or run the code in this file line by line. This will produce an object called `fit`, containing all the MCMC samples. This object will be of class `stanfit-class`, and the `rstan` library contains useful functions for inspection. The `fit` object can also be inspected by calling the usual R functions such as `summary`, `print`, or `plot`, with the optional argument `pars` to specify the parameters of interest. Output of the following functions is usually of interest:

```r
print(fit, pars=c("gompertz","beta"))
plot(fit, pars=c("gompertz","beta"))
stan_ac(fit, pars=c("gompertz","beta"))
stan_hist(fit, pars=c("gompertz","beta"))
```

To see a list of all the functions that can act on the `stanfit-class`, see `help("stanfit-class")`.

Description of models

Notes on Readme and mathematics

This readme is written in markdown, and if you are viewing it as rendered by GitHub, the mathematics won't work. For this reason there is also a `readme.pdf` file included in the repository, which was generated using `pandoc`[1]:

```
pandoc -f markdown -t latex -o readme.pdf readme.md
```

In future versions, the bulk of this readme will be moved into a LaTeX document, and the readme will just contain basic information.

Model 1

This is a survival model, where events happen according to the the hazard rate:

$$h(t) = Be^{\theta t} \tag{1}$$

$$H(t) = \frac{B}{\theta}\left(\exp(\theta t) - 1)\right) \tag{2}$$

The events are simulated using $H^{-1}(t)$, and then subjected to random, uninformative censoring.

The stan file constructs the likelihood using model blocks.

Model 2

This is the same as model 1, but we have multiple gompertz parameters. We try to use a `group` variable to assign the right gompertz parameters within Stan.

Thus, this model fits the hazard rate

$$h(t) = B_j e^{\theta_j t} \tag{3}$$

where j is the group number of the subject in question. It is a covariate in our model.

[1]Version `1.15.0.6`. Pandoc renders the pdf using LaTeX. Information and source code available at http://pandoc.org.

Model 3

This is the same as model 2, but we add a “linear predictor”:

$$\eta = \beta x \tag{4}$$

where x is a covariate.

The hazard rate then becomes

$$\begin{aligned} h(t) &= B_j e^{\theta_j t} e^{\eta} \\ &= B_j e^{\theta_j t} e^{\beta x} \end{aligned}$$

where j is the group number of the subject in question. This is similar to the cox proportional hazards model. Notice that the linear predictor does not contain an intercept term. This is since the gompertz parameter B_j already acts as an intercept for that group. The β parameter measures the influence of of covariate x measured against the baseline. That is, we can rewrite the hazard rate as:

$$h(t) = e^{\theta_j t + \beta x + log(B_j)}$$

Thus, $log\,(B_j)$ is the intercept for group j, θ_j measures the influence of time on the hazard rate for group j, and β measures the influence of covariate x, assuming the influence is global over all groups.

Likelihood, estimation, and choice of prior parameters

All of our models assume that survival times come from a process with hazard rate $h(t)$, and we can use a Poisson process to show the likelihood contribution of each observation. Let T be the event-time of a subject in our model, then

$$T > t \quad \longleftrightarrow \quad N(t) < 1$$

where $N(t)$ is the number of events at time t, which will always be 0 or 1 in our model, then we have

$$N(t) \sim Poisson\left(\int_0^t h(x)dx\right).$$

We don’t observe T_i directly, since our observations are subject to censoring. We observe $V_i = min(T_i, C_i)$ where C_i is a censoring time, assumed to be random and independent of T (non-informative censoring). Our observations are the pairs (v_i, δ_i) where δ_i is an indicator taking 1 if $T_i \leq C_i$ and 0 otherwise. Thus, δ_i is an event indicator, and it will be equal to the number of events at time t.

The Poisson process (and some algebra) provides the the likelihood contribution of the observation pair (v_i, δ_i):

$$\begin{aligned}
\mathcal{L}\left(\theta|(v_i, \delta_i)\right) &= S(v_i)^{\delta_i} f(v_i)^{(1-\delta_i)} \\
&= h(v_i)^{\delta_i} \exp\left(-\int_0^{v_i} h(x)dx\right) \\
&= h(v_i)^{\delta_i} \exp\left(-H(v_i)\right)
\end{aligned}$$

where θ is the set of all parameters. This is the likelihood we need to add to MCMC software if we want to fit survival models.[2] The point of this repository is to show how this can done using Stan's `functions` block.

Stan was able to recover the chosen parameters for all of the above models. For model 3, the more complicated of the three models in terms of number of parameters, convergence was very sensitive to the choice of priors assumed on the gompertz parameters, B and θ.

Add discussion here about the nuance of the prior parameters, how it effected the convergence of model 2 vs model 3, and how to avoid this problem in general by thinking carefully about prior parameters.

[2]MCMC software like Stan and JAGS do provide Poisson densities, but we can only use them if the hazard rate is constant. For more complicated hazard rate functions, we have to resort to more tedious measures. The likelihood contains both $h(t)$ and $H(t)$, and the software needs to know both.