Building a Survival Model in Stan

Materials:
https://github.com/bayesianops/stan-survival-model-workshop

Daniel Lee
October 16, 2023
R/Pharma workshop

daniel@bayesianops.com
dlee@zelusanalytics.com

Outline

4 Examples

1. Constant Hazard Model (Simple)
2. Add Covariates
3. Add Censoring
4. Hierarchical Model

Notes

- My (honest) goal: at least one person learns something useful

- Techniques apply beyond survival models.
- Ask questions. Please, make this time yours.
- Slack invite: https://join.slack.com/t/rinpharmaconf/shared_invite/zt-24w18rs6f-l7Y3PM4eCiIF6VIDYTlnYA
- Slack channel: https://rinpharmaconf.slack.com/?redir=%2Farchives%2FC02JRJL2C82

- Thank you, R/Pharma! Thanks, Phil Bowsher!

Contact

Feel free to reach out

• LinkedIn:	https://www.linkedin.com/in/syclik/
• Sports:	dlee@zelusanalytics.com
• All other:	daniel@bayesianops.com

Setup

Stan + R

- Install CmdStanR

```
install.packages("cmdstanr", repos = c("https://mc-
stan.org/r-packages/", getOption("repos")))
```

- Install survival, ggplot

```
install.packages(c("survival", "ggplot2"))
```

- Git Clone / download materials from: https://github.com/bayesianops/stan-survival-model-workshop

```
script.R
```

What is Stan?

Simplified Overview

1. **Language** for writing statistical models.

2. Provides Bayesian estimates, (penalized) maximum likelihood, approximate Bayesian inference.
   Best known for Bayesian estimates.

3. Built for **complicated models**.
   Effort high for simple models.

What is Stan?

1. **Language**

2. **Algorithms**

3. **Interfaces**

Language for Statistical Models

- **Goal:** specify statistical models

$$\underbrace{x}_{\text{data}}$$

Language for Statistical Models

- **Goal:** specify statistical models

$$\underset{\substack{\uparrow \\ \text{parameters}}}{\theta}, \underset{\substack{\nwarrow \\ \text{data}}}{x}$$

Language for Statistical Models

- **Goal:** specify statistical models

$$\underset{\text{model}}{p}(\underset{\text{parameters}}{\theta}, \underset{\text{data}}{x})$$

Language for Statistical Models

- **Goal:** specify statistical models

$$\underset{\text{model}}{p}(\underset{\text{parameters}}{\theta}, \underset{\text{data}}{x})$$

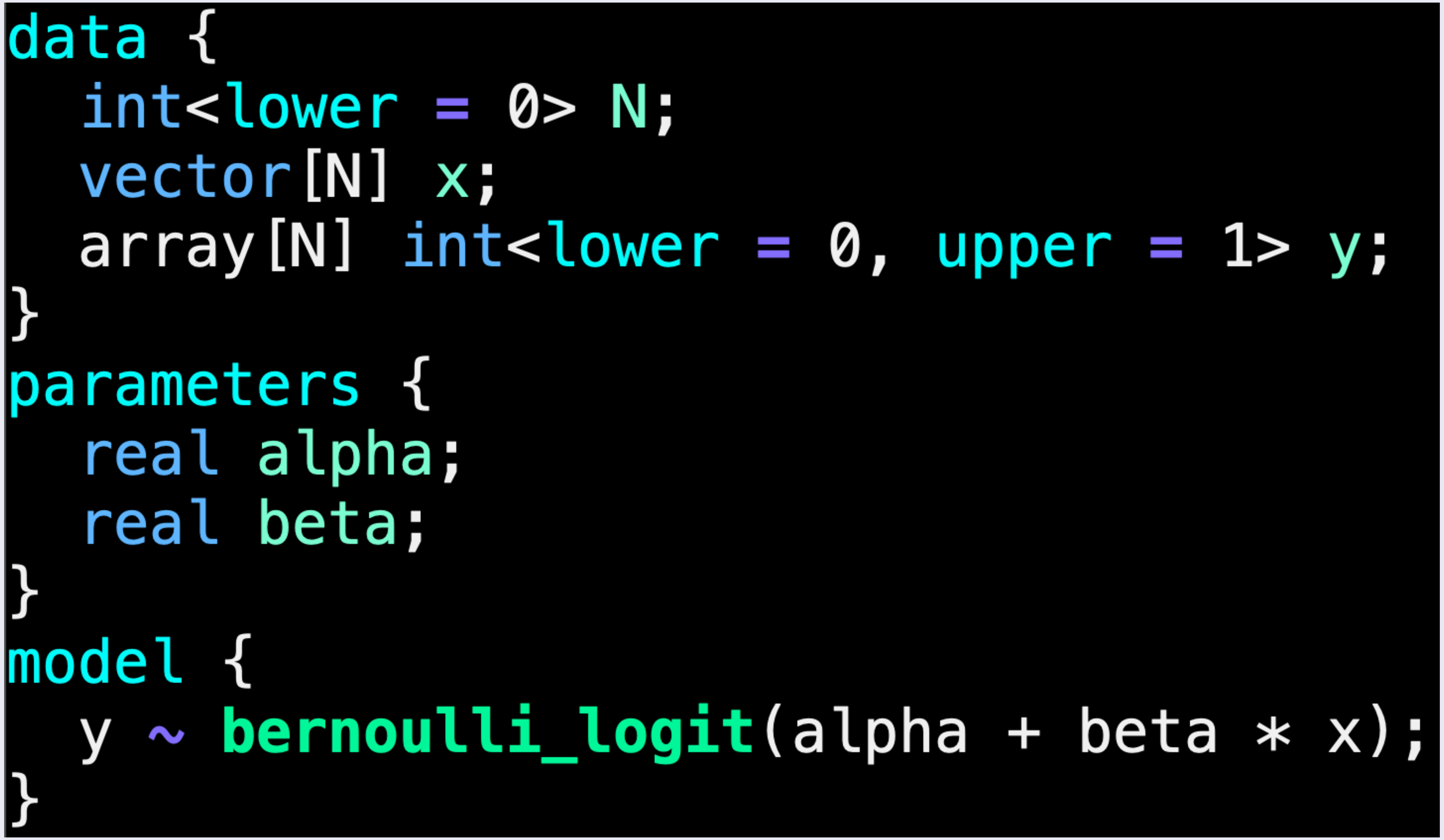
- Stan is a language
    - statically typed, imperative
    - users define programs: data, parameters, **log joint pdf**
- User can specify any *differentiable* joint probability distribution function over data and parameters

Language for Statistical Models

- **Goal:** specify statistical models

$$\underset{\text{model}}{p}(\underset{\text{parameters}}{\theta}, \underset{\text{data}}{x})$$

Example: Logistic Regression



```
data {
  int<lower = 0> N;
  vector[N] x;
  array[N] int<lower = 0, upper = 1> y;
}
parameters {
  real alpha;
  real beta;
}
model {
  y ~ bernoulli_logit(alpha + beta * x);
}
```

Users define the statistical model

$$p(\theta, x)$$

Users define the statistical model

$$p(\theta, x)$$

The statistical model is neither Bayesian or frequentist!!

Inference algorithms use $p(\theta, x)$

- Bayesian inference; Markov Chain Monte Carlo (MCMC)
- Approximate Bayesian inference
- Optimization

Inference algorithms use $p(\theta, x)$

- **Bayesian inference; Markov Chain Monte Carlo (MCMC)**
  - $p(\theta \mid x)$ approximated with $\{\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(N)}\}$
- Approximate Bayesian inference
  - ex: $\hat{p}(\theta \mid x) \approx q(\hat{\phi})$ where $\hat{\phi} = \underset{\phi}{\operatorname{argmin}}\ D_{\mathrm{KL}}\left(q(\theta \mid \phi) \,||\, p(\theta, x)\right)$
- Optimization
  - $\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\ p(\theta, x)$ (only holds when there's a single optima)

Survival Analysis: The Problem

The Problem

- For an individual, how long before an event happens?
    - What's an event?
    - Examples: death, hospitalization, equipment failure
- Difficult to predict for individuals
    - Analysis done on groups of individuals
    - Assumption: exchangeability

- **What's the expected time to an event?**

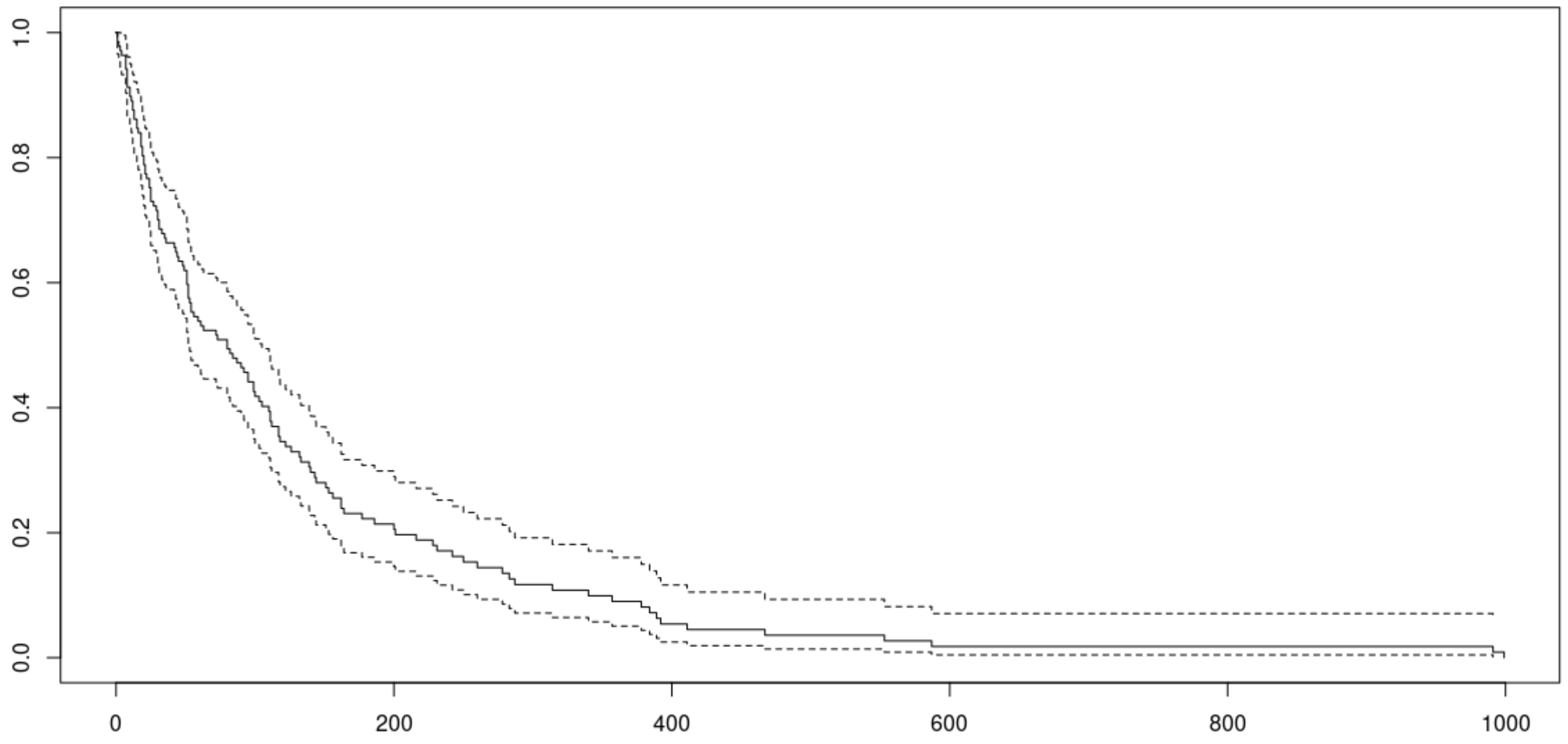
The Data

- Event of interest
    - Note: can be multiple events

- For each individual
    - Covariates. Examples: age, sex, bmi, device manufacturer, batch number
    - Event time
    - Censoring time (with the censoring type)
        - Right censoring: event has not happened at time t
        - Left censoring: event happened before time t

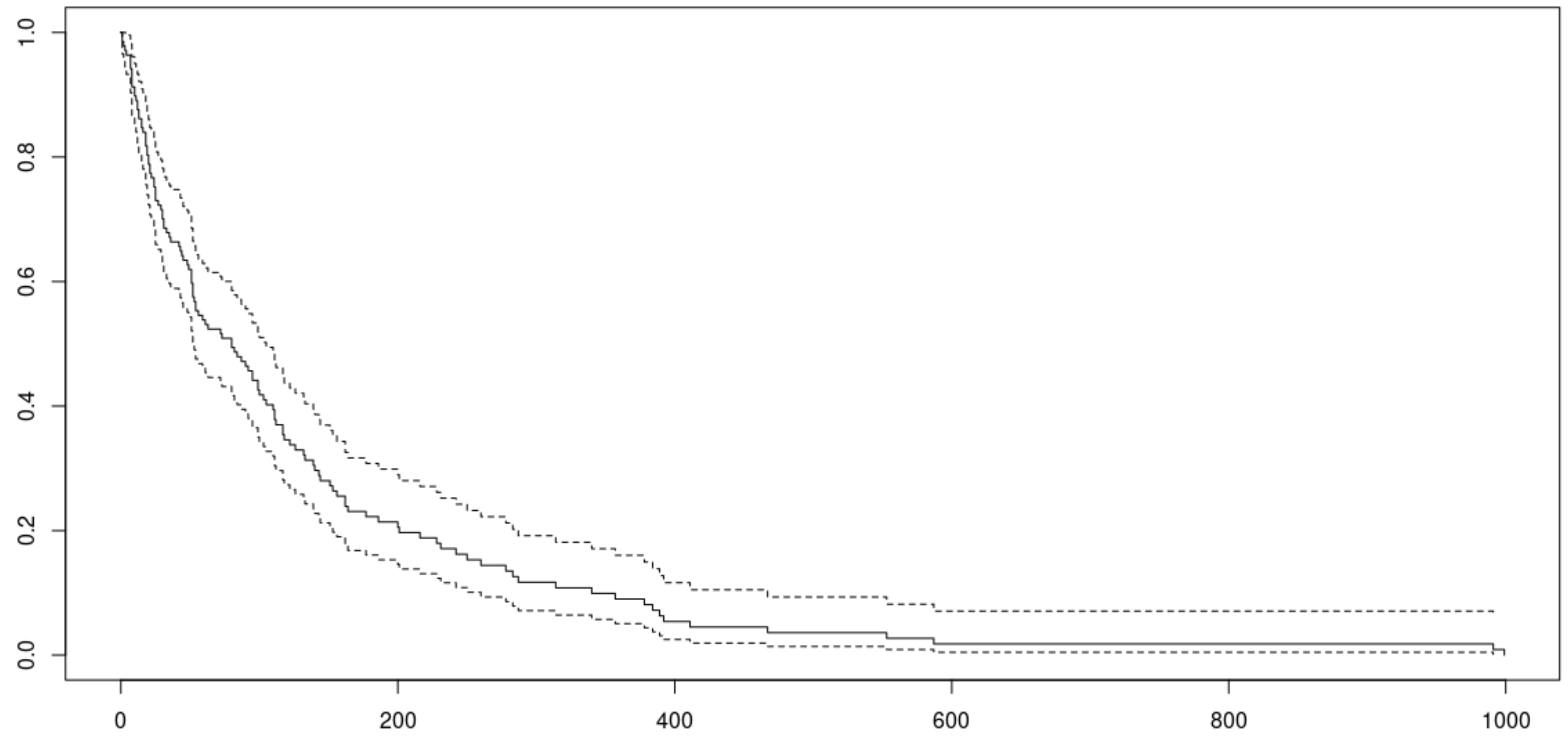
Example data: Veterans' Administration Lung Cancer study

```
library(survival)
veteran
head(veteran)
plot(survfit(Surv(veteran$time, veteran$status) ~ 1))
```



Example data: Veterans' Administration Lung Cancer study

- This plot is a Kaplan-Meier curve
  - Y-axis: percent survival
  - X-axis: time
  - Non-parametric estimate of survival curve



Example data: Veterans' Administration Lung Cancer study

- Take 5 minutes to look at the data.
  Some questions:
  - How many subjects?
  - How many treatments?
  - What's the median survival time?
  - Is there a difference if you only use non-censored observations?
- Next up: survival model math

Survival Model: Math

Terminology

- **time** t
- **survival time** T
  The "true" event time.
- **survival function** $S(t)$
  The probability that the survival time is past time t, i.e. Pr(T > t).
- **hazard function** $h(t)$ or $\lambda(t)$
  The event rate at time t conditioned on survival time T ≥ t
- **cumulative hazard function** $\Lambda(t)$
  Accumulation of hazard over time to t.

Terminology

- **time** t

- **survival time** T
  The "true" event time.

3 • **survival function** $S(t)$
  The probability that the survival time is past time t, i.e. Pr(T > t).

1 • **hazard function** $h(t)$ or $\lambda(t)$
  The event rate at time t conditioned on survival time T ≥ t

2 • **cumulative hazard function** $\Lambda(t)$
  Accumulation of hazard over time to t.

Hazard Function $h(t)$

- **Instantaneous failure rate at time t, conditioned on survival time T ≥ t**

- Hazard function. $h(t)$ or $\lambda(t)$
  The event rate at time t conditioned on survival time T ≥ t.

$$h(t) = \lim_{\Delta t \to 0} \frac{\Pr(t \leq T < t + \Delta t \mid T \geq t)}{\Delta t}$$

  - **Conditions**
    - For all t, $h(t) \geq 0$
    - $\int_0^\infty h(t)dt = \infty$

- This is linked to the cumulative hazard function

Cumulative Hazard Function $\Lambda(t)$

- Accumulation of hazard (risk) over time.

- Cumulative hazard function $\Lambda(t)$
  Area under the curve of the hazard function up until time t.
  The more time passes, the more risk accumulates

$$\begin{aligned}\Lambda(t) &= \int_0^t h(u)du \\ &= \int_0^t \lambda(u)du\end{aligned}$$

- This is linked to the survival function

Survival Function $S(t)$

- The probability that a subject survives past time t.

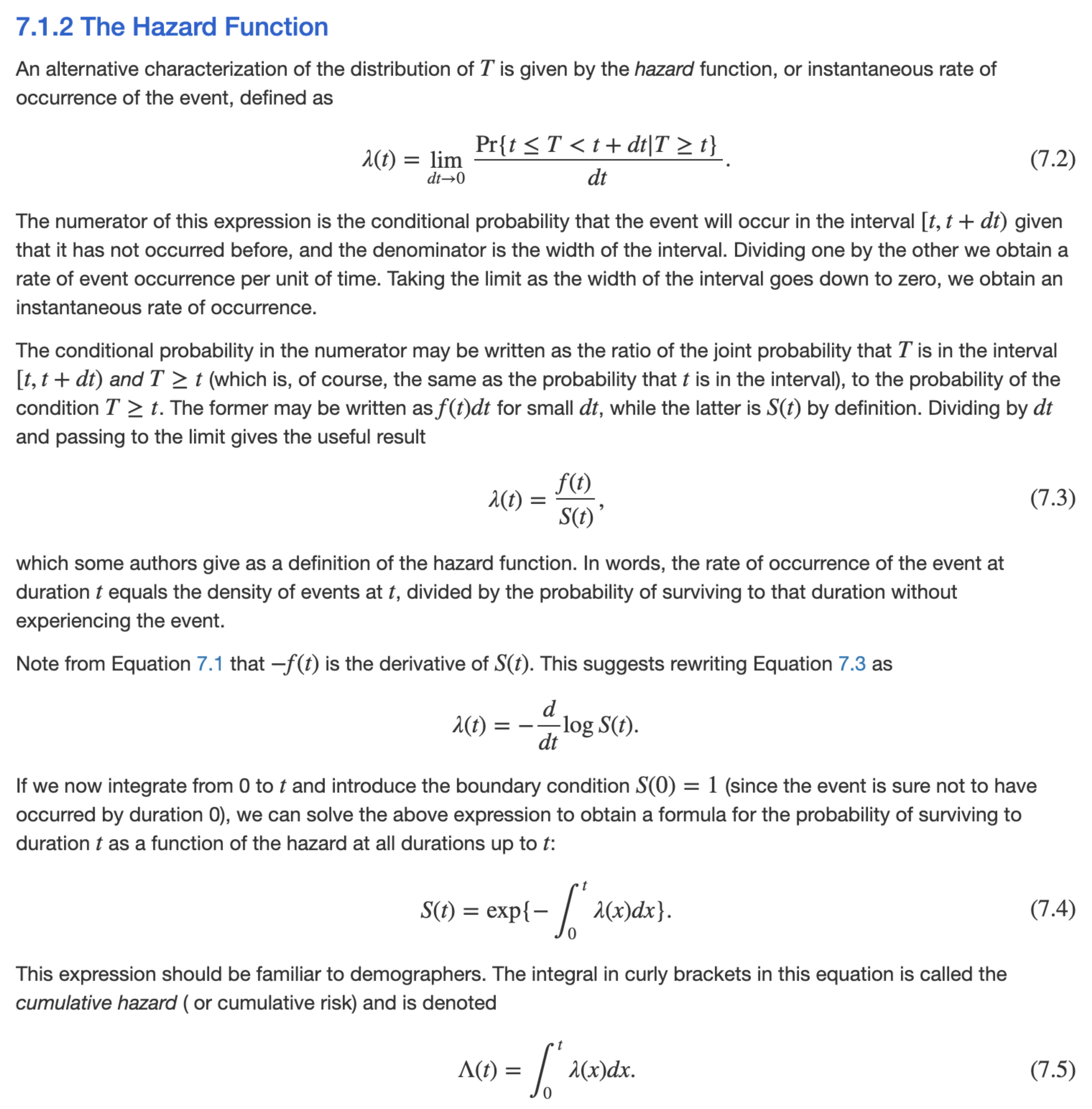
- Survival function.
  - $S(t) = \Pr(T > t)$ ← probability that survival time is greater than t
  - $S(t) = \exp(-\Lambda(t))$ ← exp of the negative cumulative survival function
    - $S(0) = 1$
    - $S(\infty) = 0$

- In Stan, we'll make use of $h(t)$, $\Lambda(t)$, and $S(t)$

Aside: Why $S(t) = \exp(-\Lambda(t))$?

- https://grodri.github.io/glms/notes/c7s1
  Germán Rodríguez

- ↑
  Section 7.1.2 has a clear, mathematical derivation



7.1.2 The Hazard Function

An alternative characterization of the distribution of T is given by the *hazard* function, or instantaneous rate of occurrence of the event, defined as

$$\lambda(t) = \lim_{dt \to 0} \frac{\Pr\{t \le T < t + dt | T \ge t\}}{dt}. \qquad (7.2)$$

The numerator of this expression is the conditional probability that the event will occur in the interval $[t, t + dt)$ given that it has not occurred before, and the denominator is the width of the interval. Dividing one by the other we obtain a rate of event occurrence per unit of time. Taking the limit as the width of the interval goes down to zero, we obtain an instantaneous rate of occurrence.

The conditional probability in the numerator may be written as the ratio of the joint probability that T is in the interval $[t, t + dt)$ *and* $T \ge t$ (which is, of course, the same as the probability that t is in the interval), to the probability of the condition $T \ge t$. The former may be written as $f(t)dt$ for small dt, while the latter is $S(t)$ by definition. Dividing by dt and passing to the limit gives the useful result

$$\lambda(t) = \frac{f(t)}{S(t)}, \qquad (7.3)$$

which some authors give as a definition of the hazard function. In words, the rate of occurrence of the event at duration t equals the density of events at t, divided by the probability of surviving to that duration without experiencing the event.

Note from Equation 7.1 that $-f(t)$ is the derivative of $S(t)$. This suggests rewriting Equation 7.3 as

$$\lambda(t) = -\frac{d}{dt} \log S(t).$$

If we now integrate from 0 to t and introduce the boundary condition $S(0) = 1$ (since the event is sure not to have occurred by duration 0), we can solve the above expression to obtain a formula for the probability of surviving to duration t as a function of the hazard at all durations up to t:

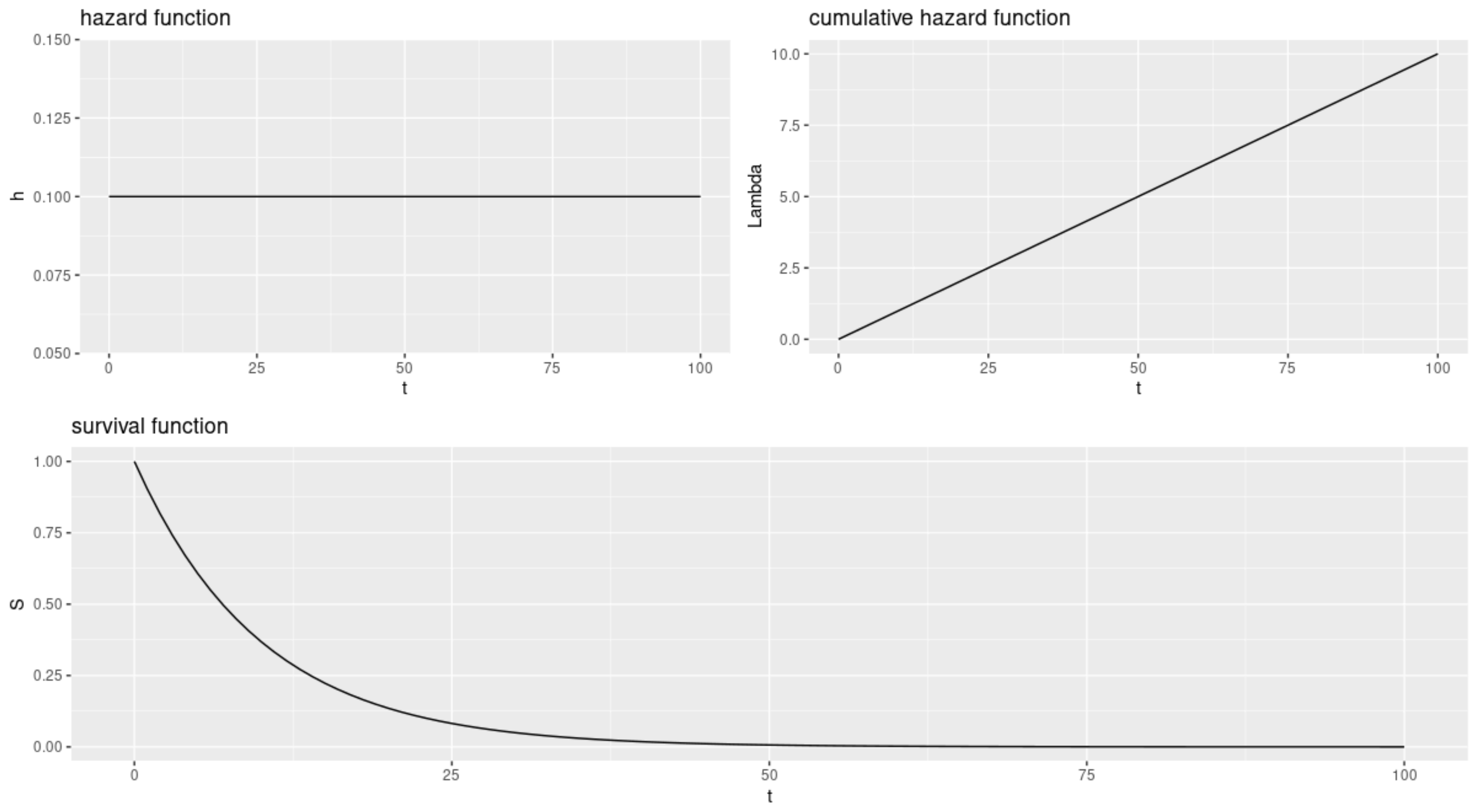
$$S(t) = \exp\{-\int_0^t \lambda(x)dx\}. \qquad (7.4)$$

This expression should be familiar to demographers. The integral in curly brackets in this equation is called the *cumulative hazard* (or cumulative risk) and is denoted

$$\Lambda(t) = \int_0^t \lambda(x)dx. \qquad (7.5)$$

Constant hazard function

- $h(t) = \lambda$, for all t ≥ 0.
- What does this look like? $\lambda = 0.1$



Constant hazard function

- Hazard function: $h(t) = \lambda$
- Cumulative hazard function: $\Lambda(t) = \lambda * t$
- Survival function:

$$\begin{aligned} S(t) &= \Pr(T > t) \\ &= \exp(-\Lambda(t)) \\ &= \exp(-\lambda * t) \end{aligned}$$

- We're going to simulate data from this model first.

Simulating Data
Inversion Sampling

Inversion Sampling

Technique for pseudo-random number sampling

- Goal: generate random output
  - Need: inverse cumulative distribution function (cdf)

- What is a cdf (of t)? $F(t) = \Pr(T \leq t)$

  Function of t that gives the probability of T ≤ t

- What's the inverse cdf? $F^{-1}(p)$
  Quantile function. If you give the function a probability, it gives you back t.

Inversion Sampling

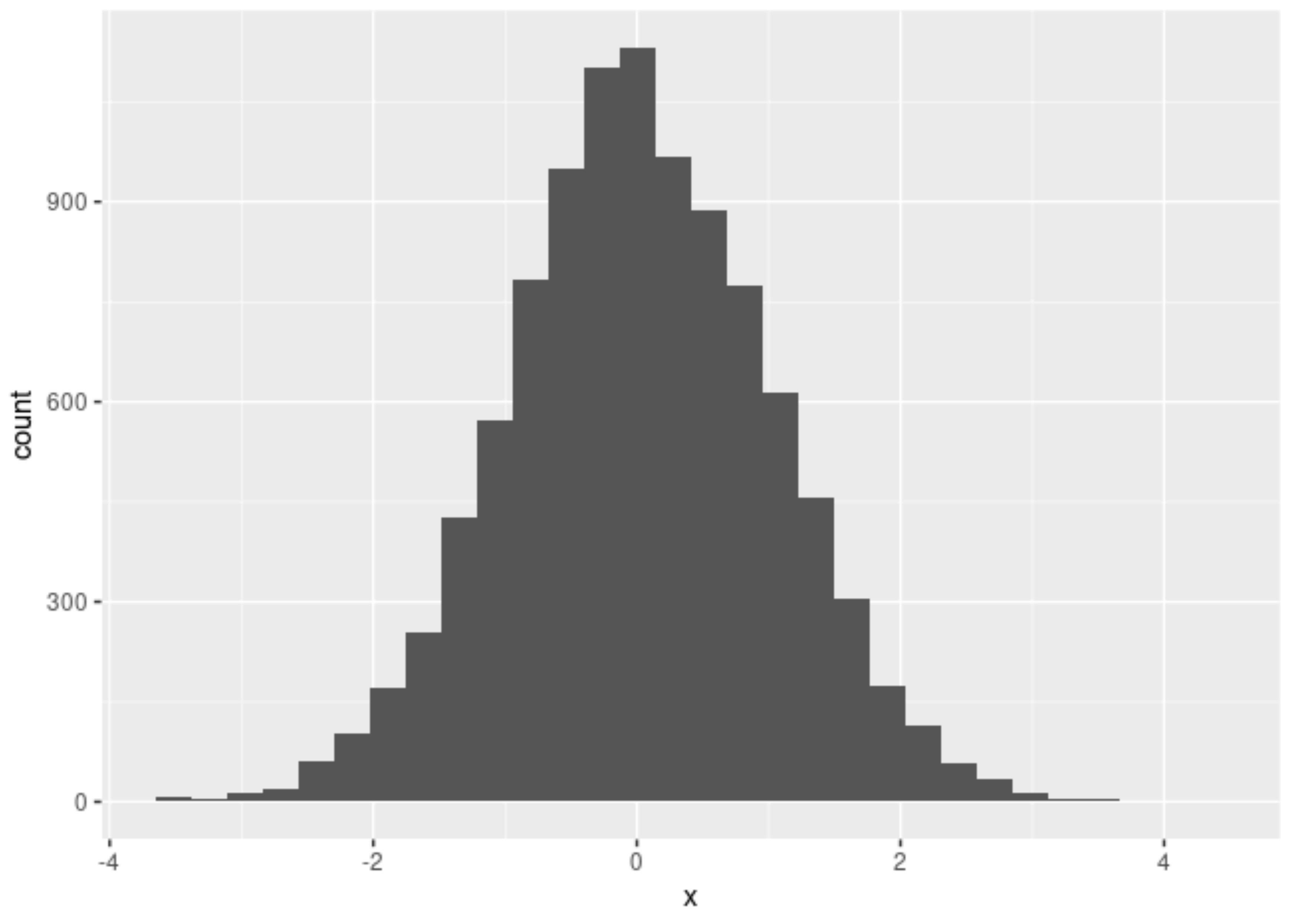
Intuition for why it works

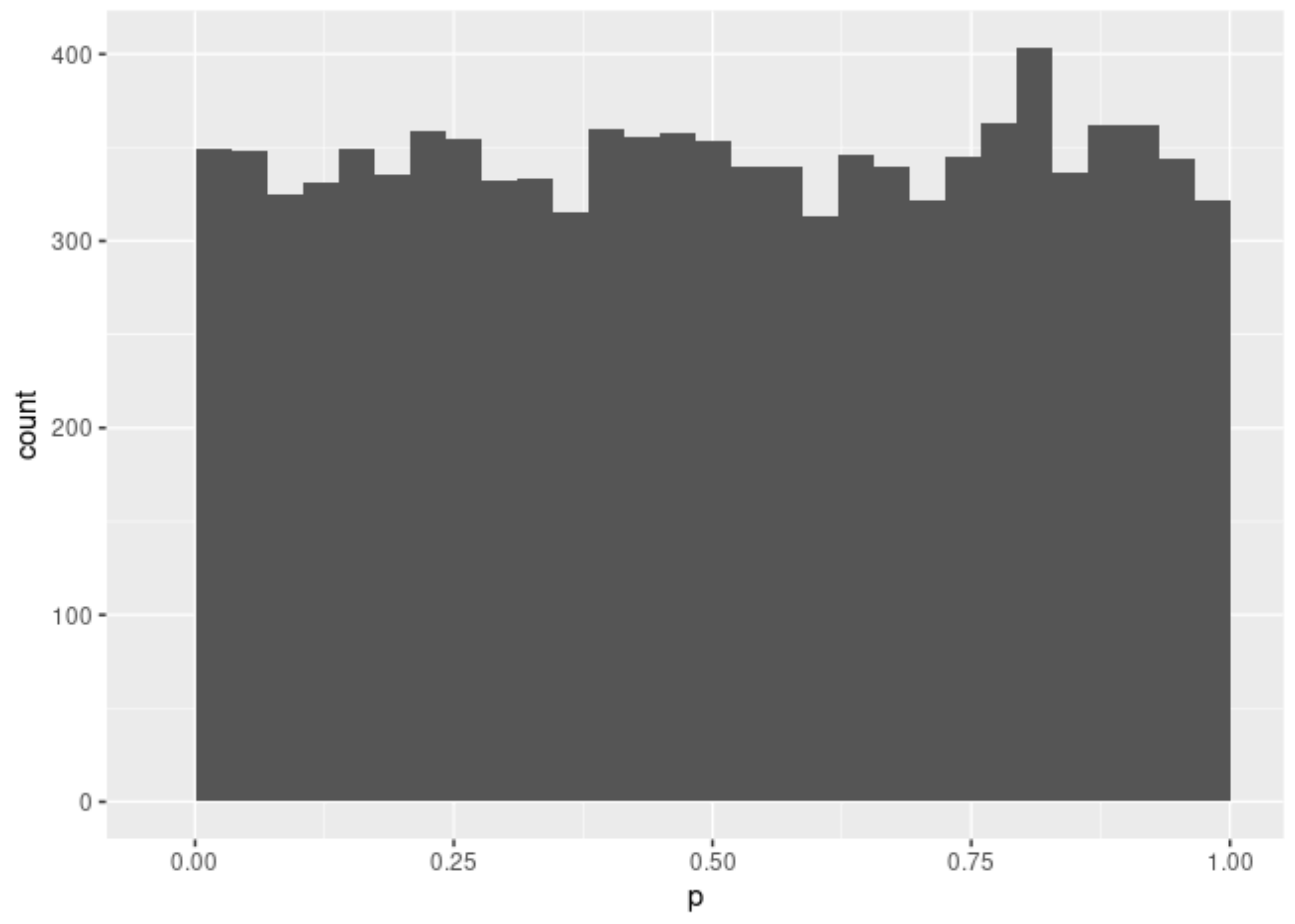
```
x <- rnorm(10000)                   # normal rng
ggplot(data.frame(x = x), aes(x = x)) + geom_histogram()


p <- pnorm(x)                       #  cdf of normal distribution
ggplot(data.frame(p = p), aes(x = p)) +
    geom_histogram(boundary = 1)
```

Inversion Sampling

Intuition for why it works

$$x \longrightarrow p$$





Inversion Sampling

The process

- For each sample
    - Draw p ~ uniform(0, 1)
    - x = inverse_cdf(p)
    - return x

Inversion Sampling

The process; normal example

- For each sample
    - Draw p ~ uniform(0, 1): p <- runif(1, 0, 1)
    - x = inverse_cdf(p): x <- qnorm(p)
    - return x

Vectorized:

```
p <- runif(10000, 0, 1)
x  <- qnorm(p)
ggplot(data.frame(x = x), aes(x = x)) + geom_histogram()
```

What's the CDF of a survival model?

- What is a cdf? $F(t) = \Pr(T \leq t)$
  Function of t that gives the probability of T ≤ t
- This is linked to the survival function: $S(t) = \Pr(T > t)$
  - cdf: $F(t) = 1 - S(t)$
- For constant hazard function, the cdf:
$$\begin{aligned} F(t) &= 1 - S(t) \\ &= 1 - \exp(-\lambda * t) \end{aligned}$$
- The inverse cdf:
$$F^{-1}(p) = \frac{-\log(1-p)}{\lambda}$$

$$\boxed{\begin{aligned} p &= 1 - S(t) \\ p &= 1 - \exp(-\lambda t) \\ p - 1 &= -\exp(-\lambda t) \\ 1 - p &= \exp(-\lambda t) \\ \log(1-p) &= -\lambda t \\ t &= \frac{-\log(1-p)}{\lambda} \end{aligned}}$$

Hands-on. Simulate Data

Pick lambda = 0.1

- Start with simulating a single subject; no covariates.
    - Draw p = runif(0, 1)
    - Since we know lambda, we want to know what time this corresponds to.
      t = -log(1 - p) / lambda
    - What value did you get?
    - Do you think you'll be able to infer lambda just from the single event time?
- Simulate for N = 100
- Advanced: use non-constant hazard function.

Hands-on. Simulate Data

- Simulate 100 individuals

```
N = 100
p = runif(N)
event_time = -log(1 - p) / lambda
```

Hands-on. Simulate Data

Pick lambda = 0.1

- Simulate N subjects (as variable event_time)
- Show KM curve.

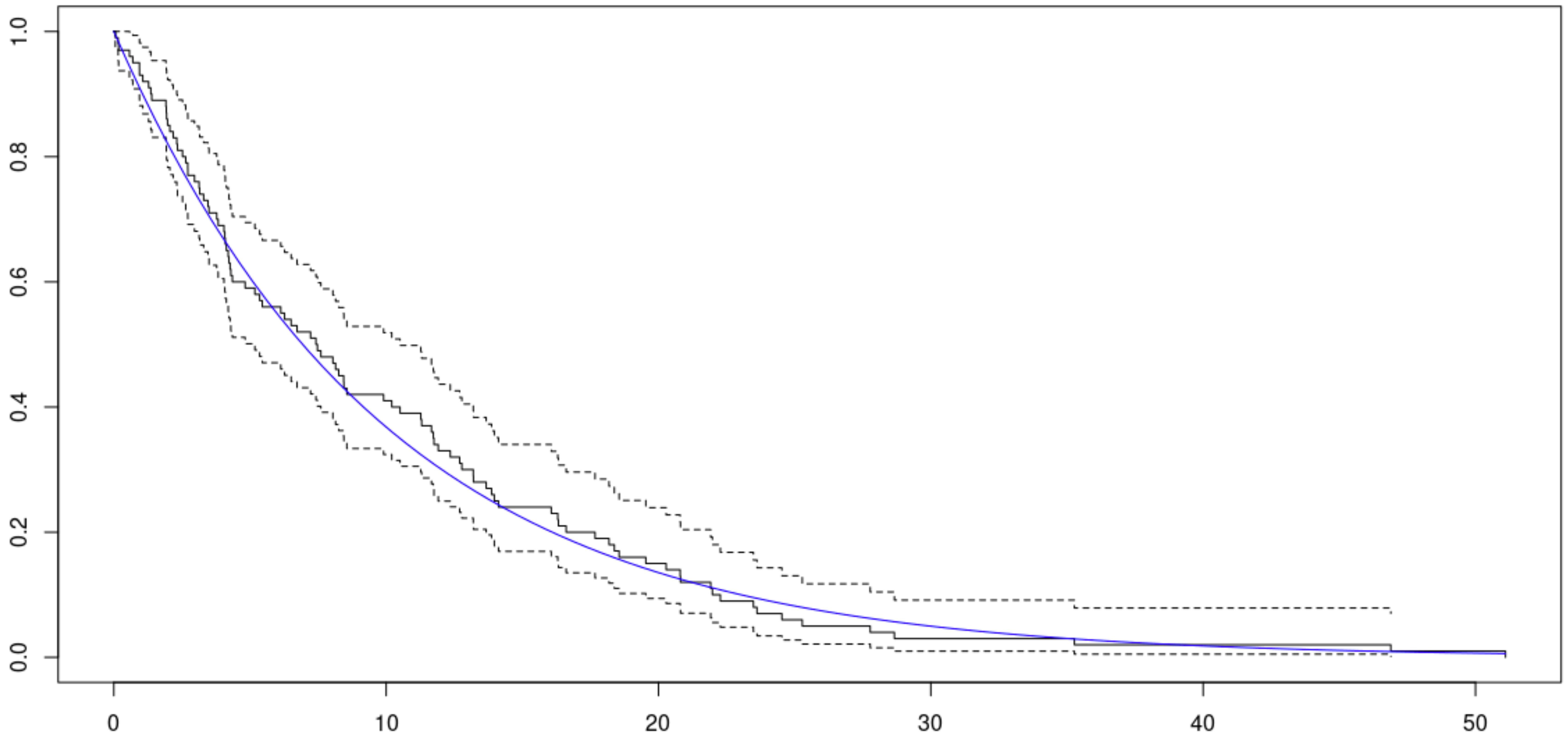
```
library(survival)
Y = Surv(event_time)
plot(survfit(Y ~ 1))
curve(exp(-x * lambda), add = T, col = 'blue')
```

- Bonus: what's the likelihood of an event at time t?

Hands-on. Simulate Data

Pick lambda = 0.1

- KM curve



- Bonus: what's the likelihood of an event at time t?

Hands on: 15 minutes

- Change the value of lambda.
- Plot
    - hazard function
    - cumulative hazard function
    - survival function

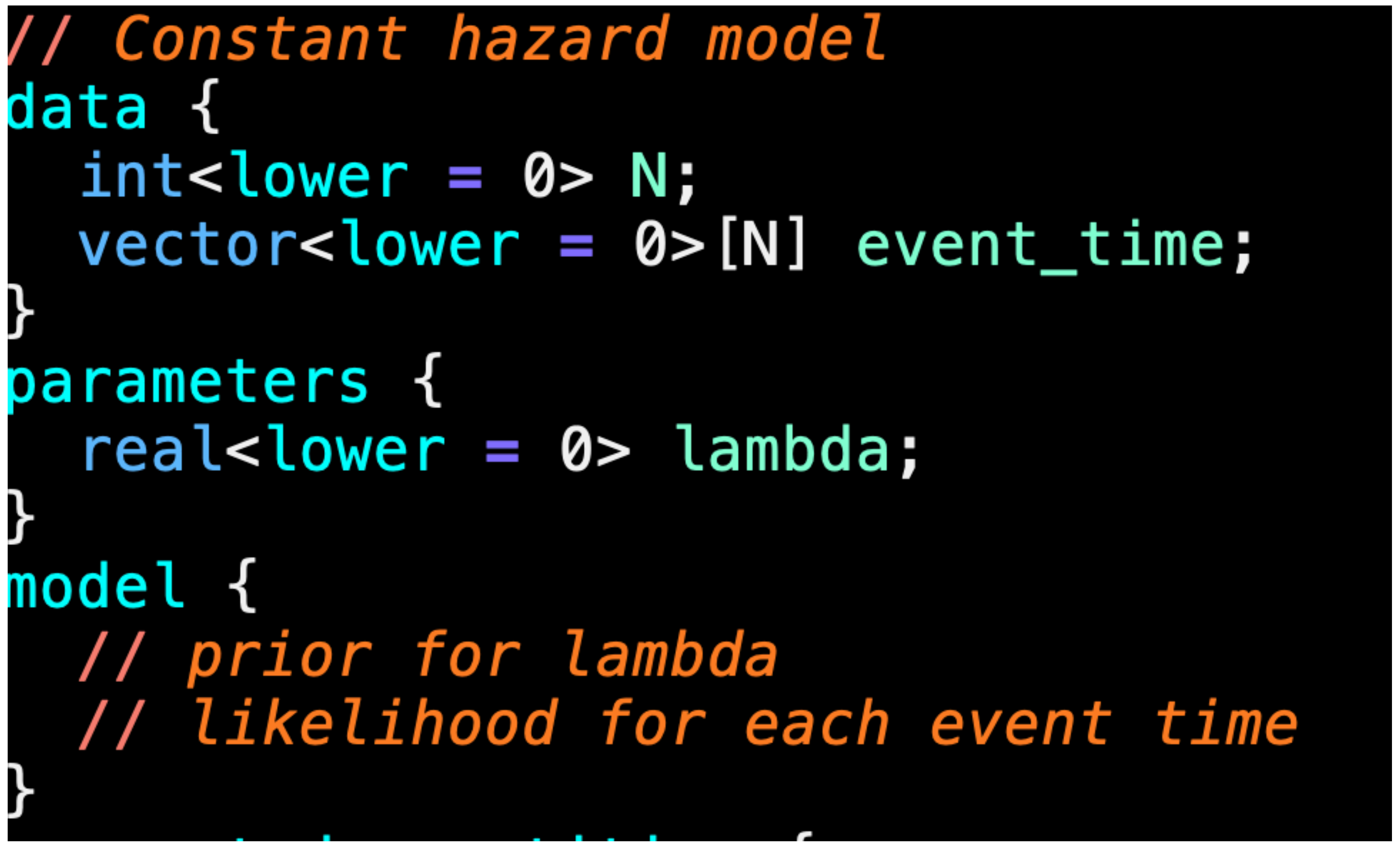
- What's the median survival? (Find t such that S(t) = 0.5.)
  What happens with higher lambda?
- Advanced: use a non-constant hazard function

1. Constant Hazard Model

Hands-on. Stan program

Data and parameters

- Open: survival_1.stan
- Data
  - `int<lower = 0> N;`
  - `vector<lower = 0>[N] event_time;`
- Parameters
  - `real<lower = 0> lambda;`



```
// Constant hazard model
data {
  int<lower = 0> N;
  vector<lower = 0>[N] event_time;
}
parameters {
  real<lower = 0> lambda;
}
model {
  // prior for lambda
  // likelihood for each event time
}
```

Hands-on. Stan program.

Model

1. Leave it alone. Let's make sure we can get this far.
   What do we expect lambda to be?

```
    mod = cmdstan_model("survival_1.stan")
    mod$print()
    data_list = list(N = length(event_time),
                     event_time = event_time)
    fit = mod$sample(data = data_list, seed = 123, chains =
    4, parallel_chains = 4, refresh = 500)
    fit$summary();    mcmc_hist(fit$draws("lambda"))
```

2. Prior on lambda. No likelihood. Let's just put it in the right ballpark.
   `lambda ~ normal(0, 1);`
   (don't like it? Put something wider / different.)

3. Next: let's get to the full model

What's the likelihood of an event at time t?

- We need the likelihood for each observation.

- What's the probability that the event occurred right at the event_time?
    - Pr(T = event_time | lambda)?
        - Hint: we need to combine two things we have
    - The survival time, T, is greater than or equal to event_time.... AND
    - The survival event happened right at time event_time

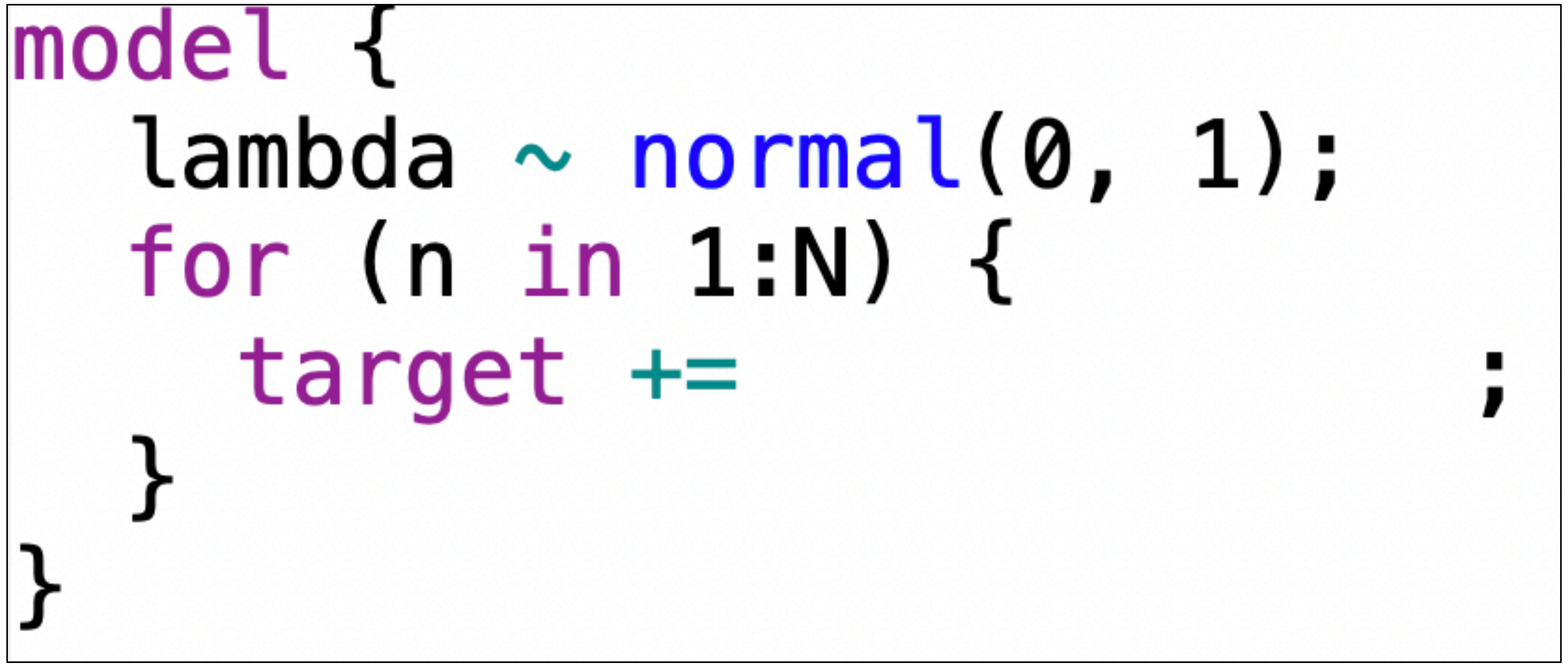
What's the likelihood of an event at time t?

- The survival time, T, is greater than or equal to t (t = event_time) $\longrightarrow$ Pr(T ≥ t) = S(t)
- The survival event happened right at time t $\longrightarrow$ h(t)
- Likelihood: S(t) * h(t)
  - Constant hazard model likelihood: $\exp(-\lambda * t) \times \lambda$
  - Log likelihood: $-\lambda * t + \log(\lambda)$

Hands-on. Stan program. 5 minutes

The model

- For each individual, include the log likelihood: $-\lambda * t + \log(\lambda)$
    - 'target +=' is incrementing the **log likelihood**
    - Likelihood: S(t) * h(t) = exp(-lambda * t) * lambda



```
model {
  lambda ~ normal(0, 1);
  for (n in 1:N) {
    target +=                  ;
  }
}
```

Hands-on. Stan program

The model

- For each individual, include the log likelihood
    - 'target +=' is incrementing the **log likelihood**
    - Likelihood: S(t) * h(t) = exp(-lambda * t) * lambda

```
model {
  lambda ~ normal(0, 1);
  for (n in 1:N) {
    target += -lambda * event_time[n] + log(lambda);
  }
}
```

Fit the model, look at inferences for lambda

- Are you able to recover lambda?
- Simulate different data
    - More data, less data
- Are you getting good inferences?
- What about the prior?
    - Can you remove the prior?
    - Can you add a stronger prior?

```
mod = 
cmdstan_model("survival_1.stan")
mod$print()
data_list = list(N = 
length(event_time), event_time = 
event_time)
fit = mod$sample(data = 
data_list, seed = 123, chains = 
4, parallel_chains = 4, refresh 
= 500)
```

2. Add Covariates

Proportional hazards model

- A survival model that accounts for covariates
- Widely used.

- Modeling assumptions:
    - Covariates matter
    - Covariates effect hazard rate **multiplicatively**

Proportional hazards model

- Hazard function: $h(t) = h_0(t)\ \exp(X \cdot \beta)$
- Two components
  - Baseline hazard function: $h_0(t)$ (or $\lambda_0(t)$)
    hazard function for baseline level of covariates
  - Effect of covariates: $\exp(X \cdot \beta)$
    X are the covariates (for an individual), β are parameters
    If X is 0, the effect is 1
- Note: no time-varying effect of covariates

Hands on. Simulate data. 5 minutes

- Simulate data.
  - Pick lambda = 0.1, beta = log(0.5)
  - Simulate N individuals. Output: N, treatment, event_time
    - Randomly assign a treatment: 0 or 1
    - Compute event time conditional on treatment, lambda, and beta
- Plot KM curves separating on treatment
  - Should see a difference; even taking the median of the times will show it
  - The survival curves should not cross.

Hands on. Simulate data.

- Simulate data with treatment

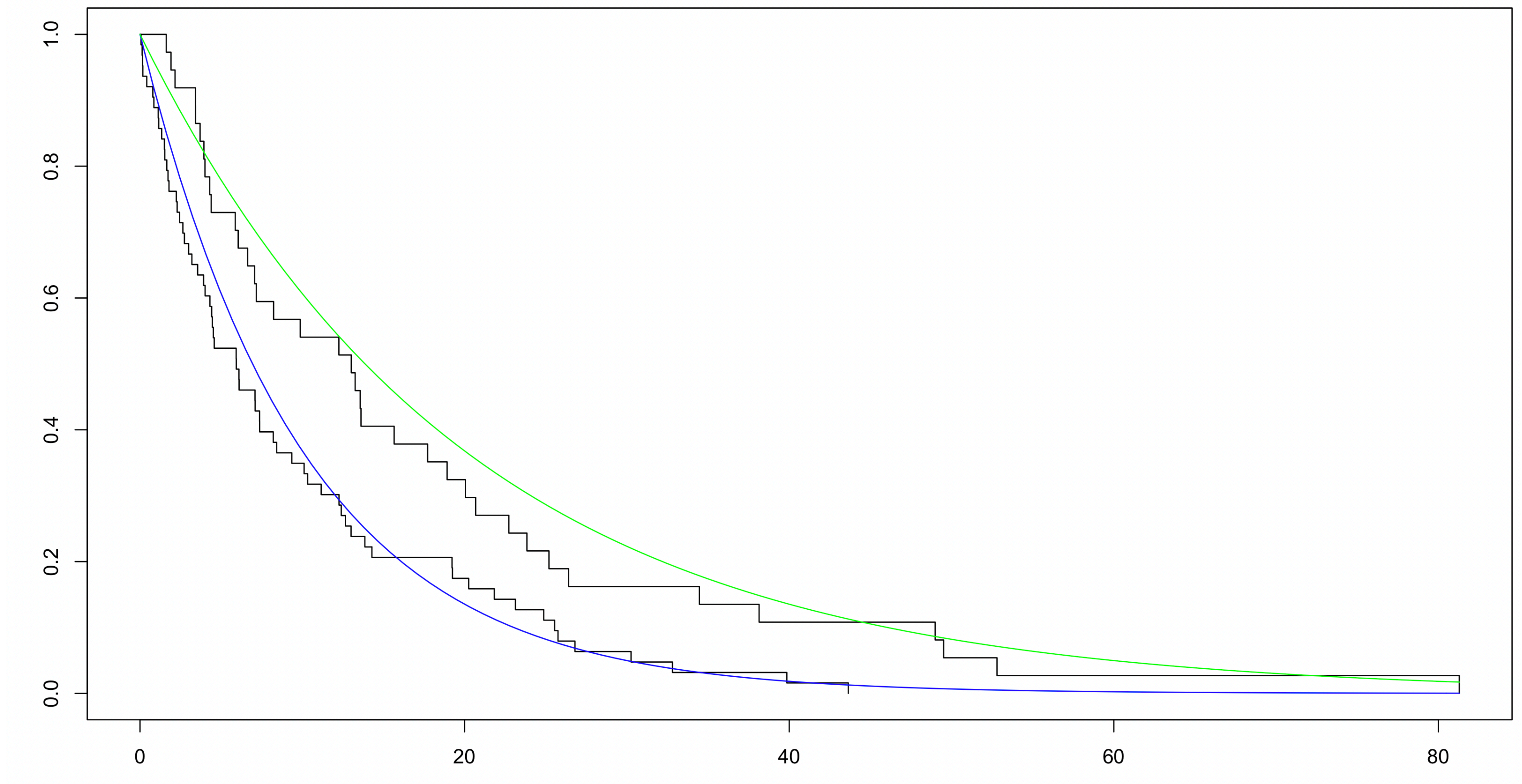
```
lambda = 0.1
beta = log(0.5)

N = 100
treatment = sample(c(0, 1), N, replace = TRUE)
p = runif(N)
event_time = -log(1 - p) / (lambda * exp(treatment * beta))
```

- KM curve

```
Y = Surv(event_time)
plot(survfit(Y ~ treatment))
curve(exp(-x * lambda), add = T, col = 'blue')
curve(exp(-x * lambda * exp(beta)), add = T, col = 'green')
```

Hands on. Simulate data.

- KM curves



Hands-on. Stan program. 15 minutes

Extend the model

- Add data
  - `array[N] int<lower = 0, upper = 1> treatment;`
- Add parameter
  - `real beta;`
- Model

1. Add prior. beta ~ normal(0, 1); Run it.
2. Add proportional hazard

```
Original: target += -lambda * event_time[n] + log(lambda)
Updated:
```

Hands-on. Stan program. 15 minutes

Extend the model

- Add data
  - `array[N] int<lower = 0, upper = 1> treatment;`
- Add parameter
  - `real beta;`
- Model

1. Add prior. beta ~ normal(0, 1); Run it.

2. Add proprotional hazard

```
Original: target += -lambda * event_time[n] + log(lambda);
Updated:  real lambda_n = lambda * exp(treatment[n] * beta);
          target += -lambda_n * event_time[n] +
log(lambda_n);
```

Stan Program

```
data {
  int<lower = 0> N;
  array[N] real event_time;
  array[N] int<lower = 0, upper = 1> treatment;
}
parameters {
  real<lower = 0> lambda;
  real beta;
}
model {
  lambda ~ normal(0, 1);
  beta ~ normal(0, 1);
  for (n in 1:N) {
    real lambda_n = lambda * exp(treatment[n] * beta);
    target += -lambda_n * event_time[n] + log(lambda_n);
  }
}
```

Problems with this model?

- Identifiability problem
  - lambda = 0.1, beta = log(0.5) = -0.7
  - lambda = 0.05, beta = log(2) = 0.7
- Other issues?

3. Add Censoring

Censoring

- Partial information about the event time

- Types
  - Right: event occurs after some known time
  - Left: event occurred before some known time
  - Interval: event occurs between two time points
  - ...

Hands on. Simulate data. 15 minutes

- Simulate right censored data. Event occurs after some known time.
    - Pick lambda = 0.1, beta = 0.5, censor_time = 40
    - Simulate N individuals. Output: N, treatment, event_time, censored
        - Randomly assign a treatment: 0 or 1
        - Compute event time conditional on:
          treatment, lambda, beta, and censor_time
        - Assign censored: 0 or 1

- Bonus: random censoring instead of using censor_time.
  (Draw **independent** censor_time per individual)

Hands on. Simulate data.

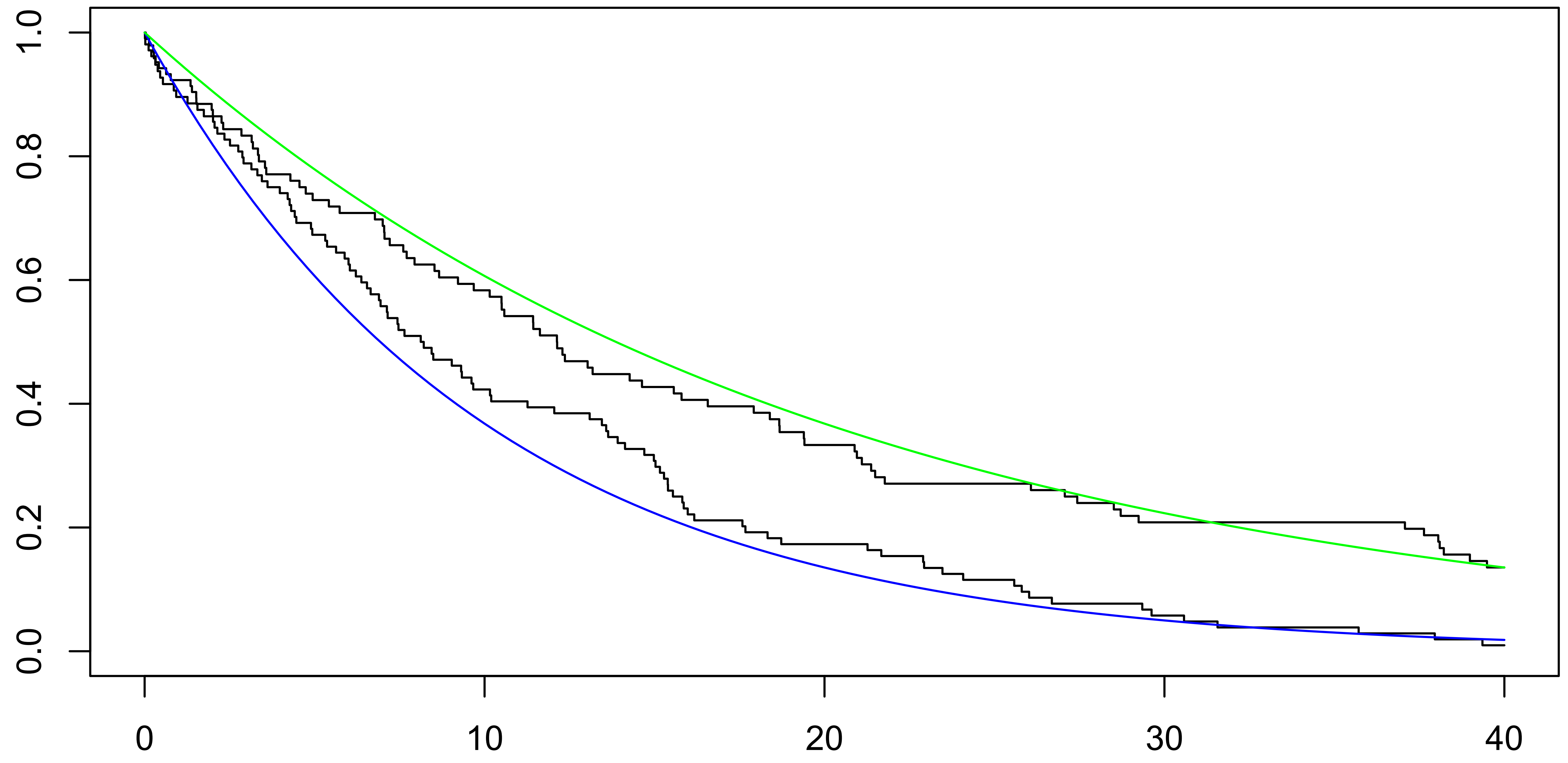
- Simulate right censored data. Event occurs after some known time.

```
lambda = 0.1
beta = log(0.5)
censor_time = 40

N = 200
treatment = sample(c(0, 1), N, replace = TRUE)
p = runif(N)
true_event_time = -log(1 - p) / (lambda * exp(treatment *
beta))
censored = ifelse(true_event_time > censor_time, 1, 0)
event_time = pmin(true_event_time, censor_time)
```

Hands on. Simulate data.

- KM curves

```
Y = Surv(event_time, censored == 0)
plot(survfit(Y ~ treatment))
curve(exp(-x * lambda), add = T, col = 'blue')
curve(exp(-x * lambda * exp(beta)), add = T, col = 'green')
```



Review: What's the likelihood of an event at time t?

- What's the probability that the event occured right at the event_time?
  - Pr(T = event_time | lambda)?
  - The survival time, T, is greater than or equal to event_time.... AND
  - The survival event happened right at time event_time

Review: What's the likelihood of an event at time t?

- The survival time, T, is greater than or equal to t (t = event_time) $\longrightarrow$ Pr(T ≥ t) = S(t)
- The survival event happened right at time t $\longrightarrow$ h(t)
- Likelihood: S(t) * h(t)
  - Constant hazard model likelihood: $\exp(-\lambda * t) \times \lambda$
  - Log likelihood: $-\lambda * t + \log(\lambda)$

What's the censored likelihood at time t?

- What's the probability that the event occured after the censor_time?
    - Pr(T > censor_time | lambda)?
    - The survival time, T, is greater than or equal to censor_time
- (Does this look familiar?)

What's the censored likelihood at time t?

- The survival time, T, is greater than t (t = censor_time) $\longrightarrow$ Pr(T > t) = S(t)
- ~~The survival event happened right at time t $\longrightarrow$ h(t)~~
- Likelihood: S(t)
    - Constant hazard model censoredlikelihood: $\exp(-\lambda * t)$
    - Log censored likelihood: $-\lambda * t$

Hands-on. Stan program. 15 minutes

Extend model: censoring

- Add data
    - `array[N] int<lower = 0, upper = 1> censored;`
- Model
    - Add conditional

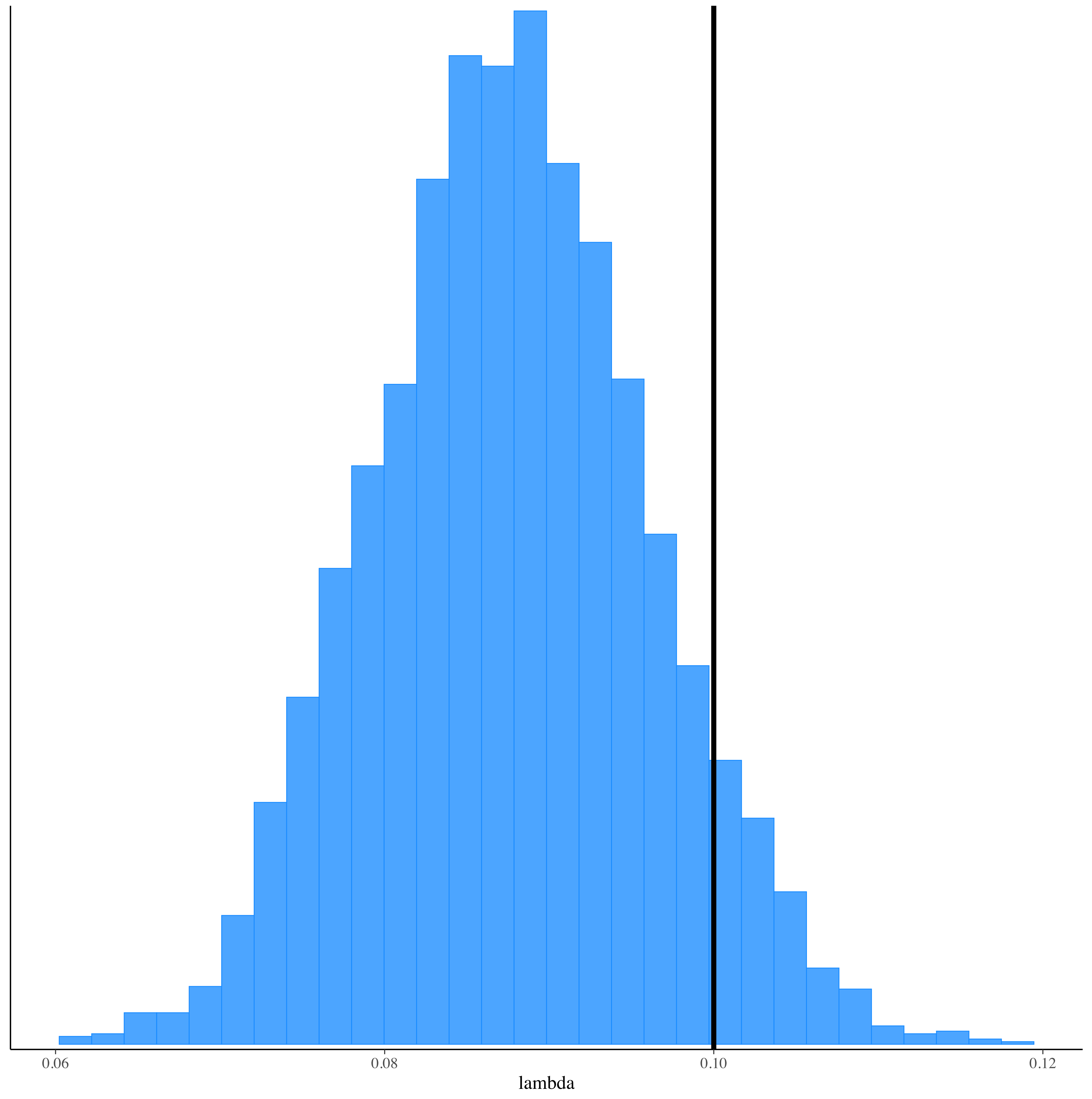
```
if (censored[n] == 0) {
  target +=           ;
} else {
  target +=           ;
}
```

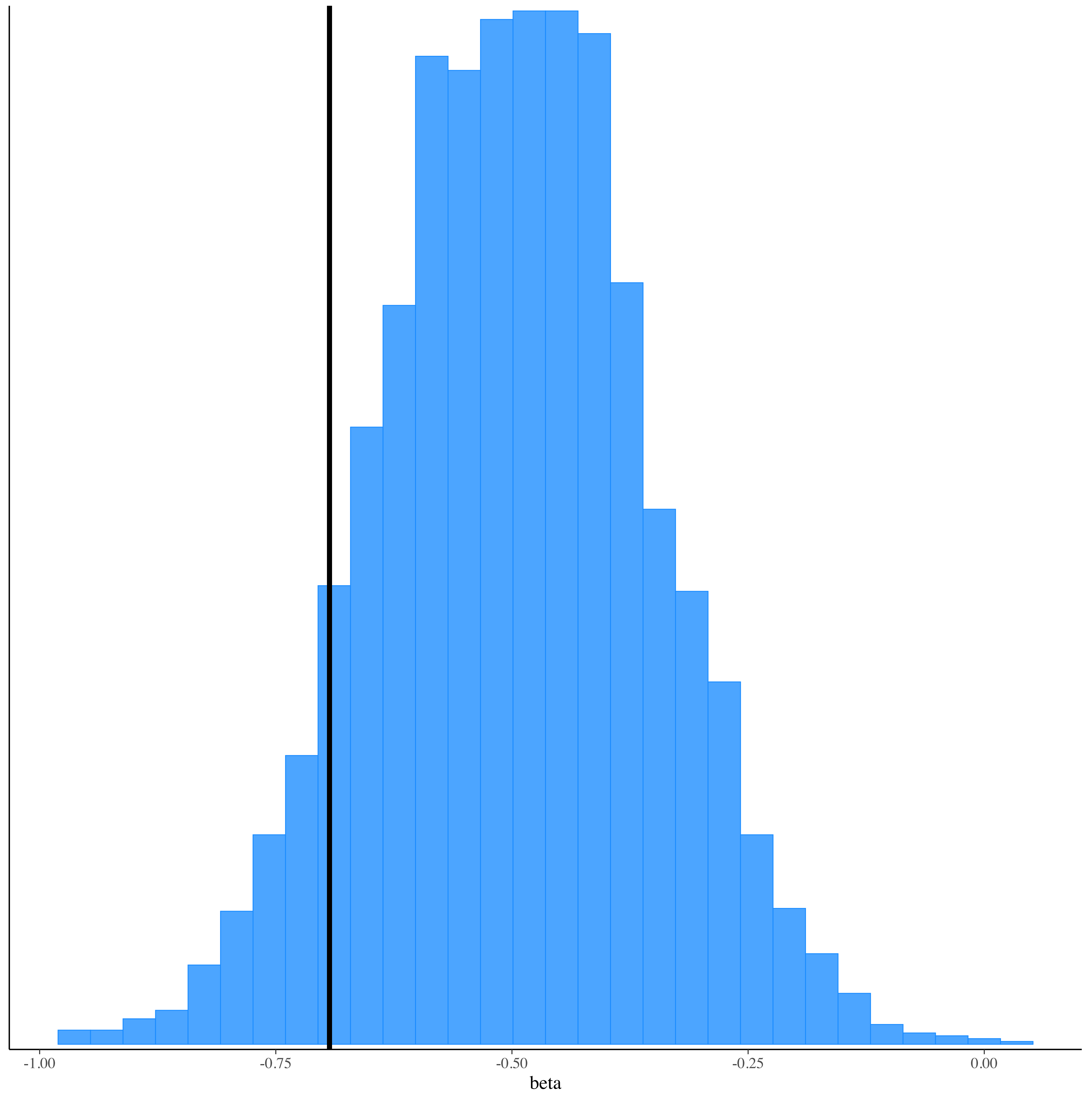
- Bonus: optimize code

Stan Program

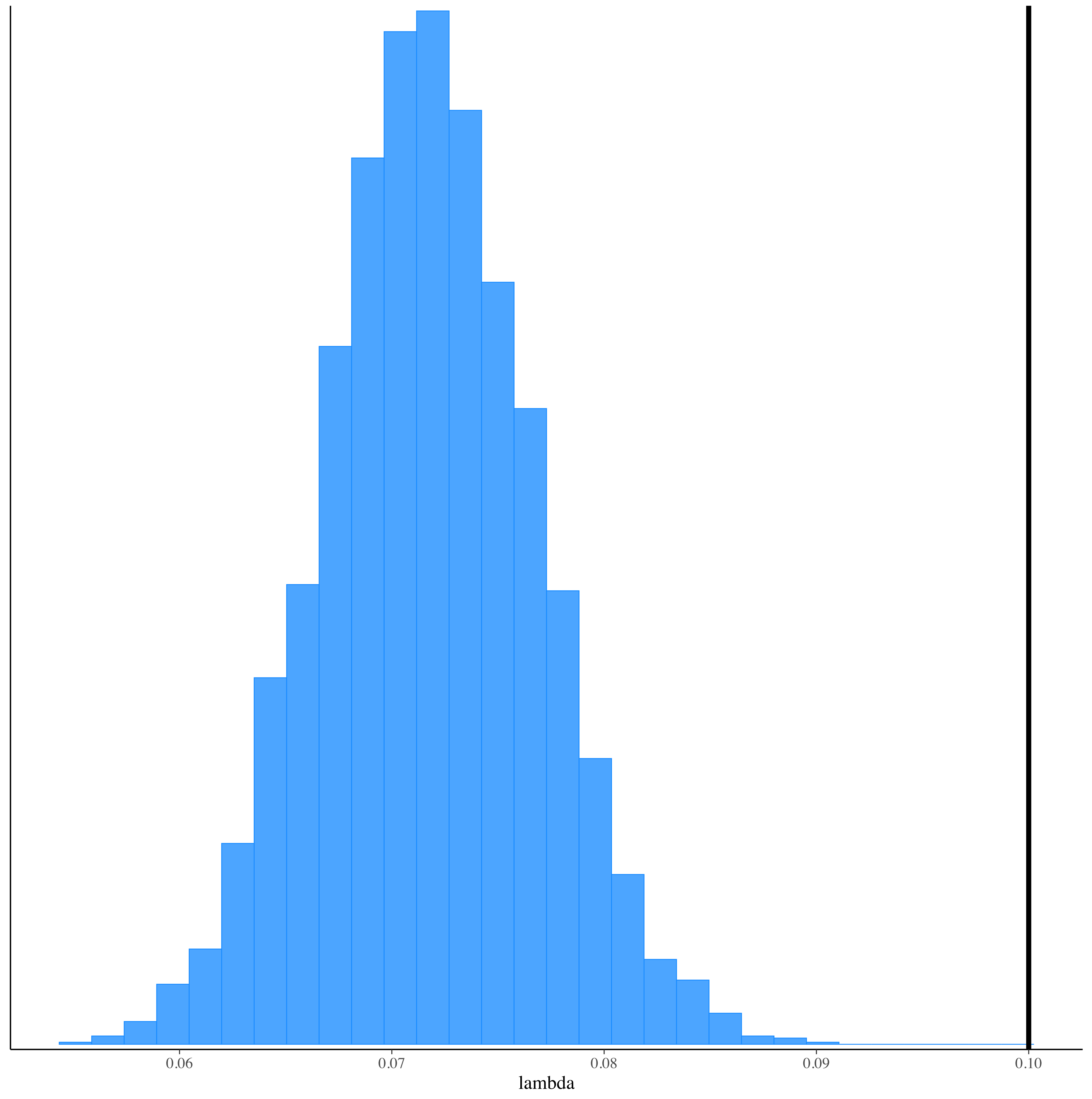
```
data {
  int<lower = 0> N;
  vector<lower = 0>[N] real event_time;
  array[N] int<lower = 0, upper = 1> treatment;
  array[N] int<lower = 0, upper = 1> censored;
}
parameters {
  real<lower = 0> lambda;
  real beta;
}
model {
  lambda ~ normal(0, 1);
  beta ~ normal(0, 1);
  for (n in 1:N) {
    real lambda_n = lambda * exp(treatment[n] * beta);
    if (censored[n] == 0) {
      target += -lambda_n * event_time[n] + log(lambda_n);
    } else {
      target += -lambda_n * event_time[n];
    }
  }
}
```

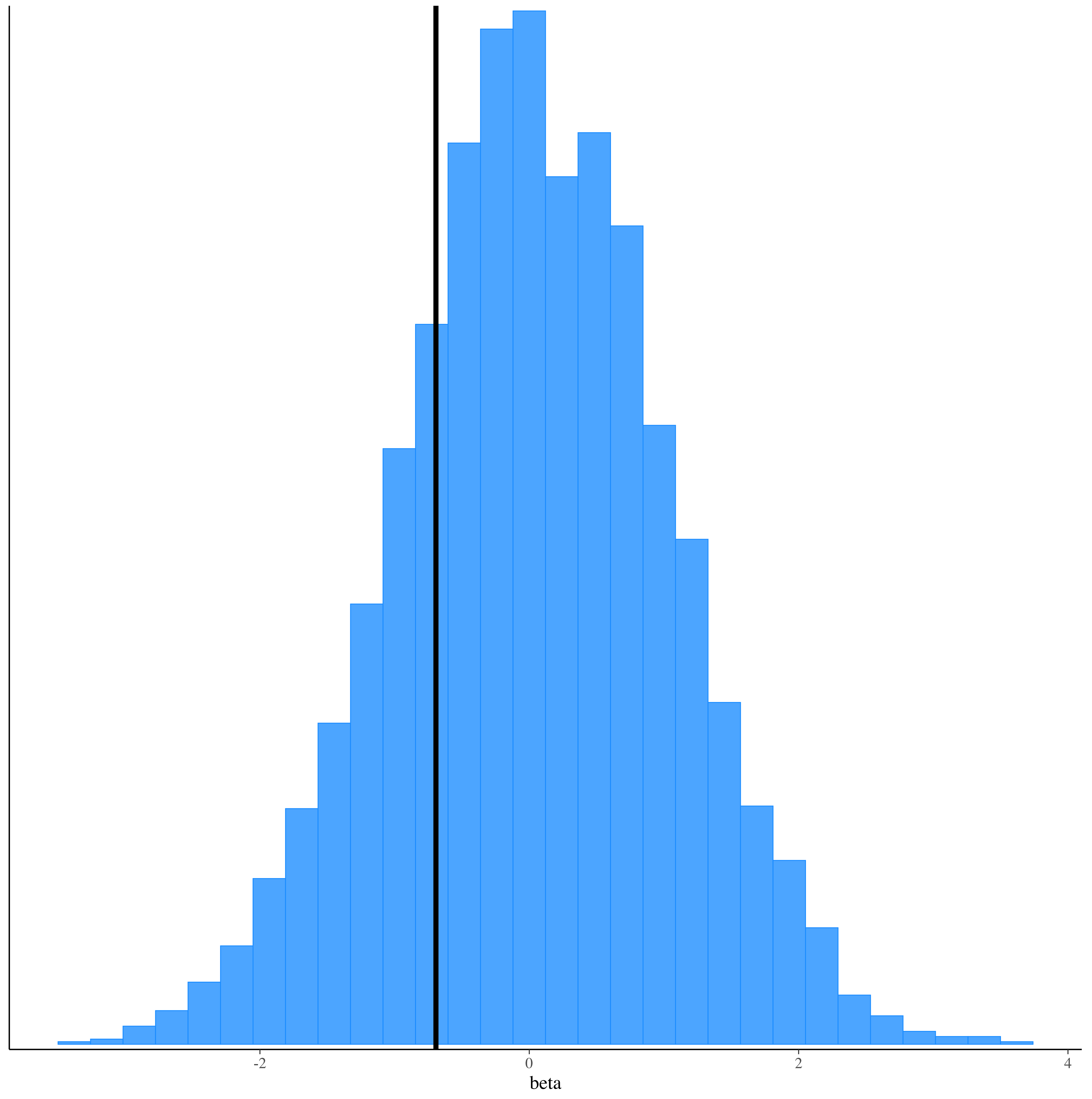
Censored Model Estimates





Non-Censored Model Estimates





4. Hierarchical Model

Hierarchical models

- Multiple parameters related
- Natural to model hierarchy of parameters: individuals within groups
- Model is often parametric
- Parameters of the hierarchical model are called *hyperparameters*

Example hierarchy

- Instead of a global lambda, each individual has a lambda
    - Model is now overparameterized without a hierarchy.
- Simple model for relating the individual lambdas:

$$\lambda_i \sim \mathrm{Normal}(\mu_\lambda, \sigma_\lambda)$$

- Now we want to estimate each individual lambda and hyperparmeters: mu and sigma

Hands on. Simulate data. 5 minutes

- Simulate data.
  - Pick mu_lambda = 0.3, sigma_lambda = 0.05,
    beta = log(0.5), censor_time = 15
  - Simulate N individuals. Output: N, lambda, treatment, event_time, censored
    - Randomly assign a treatment: 0 or 1
    - Compute event time conditional on:
      lambda (individual), beta, treatment, and censor_time
    - Assign censored: 0 or 1 (if true_event_time > censor_time)
- Plot KM curves separating on treatment
  - Should see a difference; even taking the median of the times will show it
  - The survival curves may cross (if beta is close to 0)

Hands on. Simulate data.

- Simulate data with individual lambda

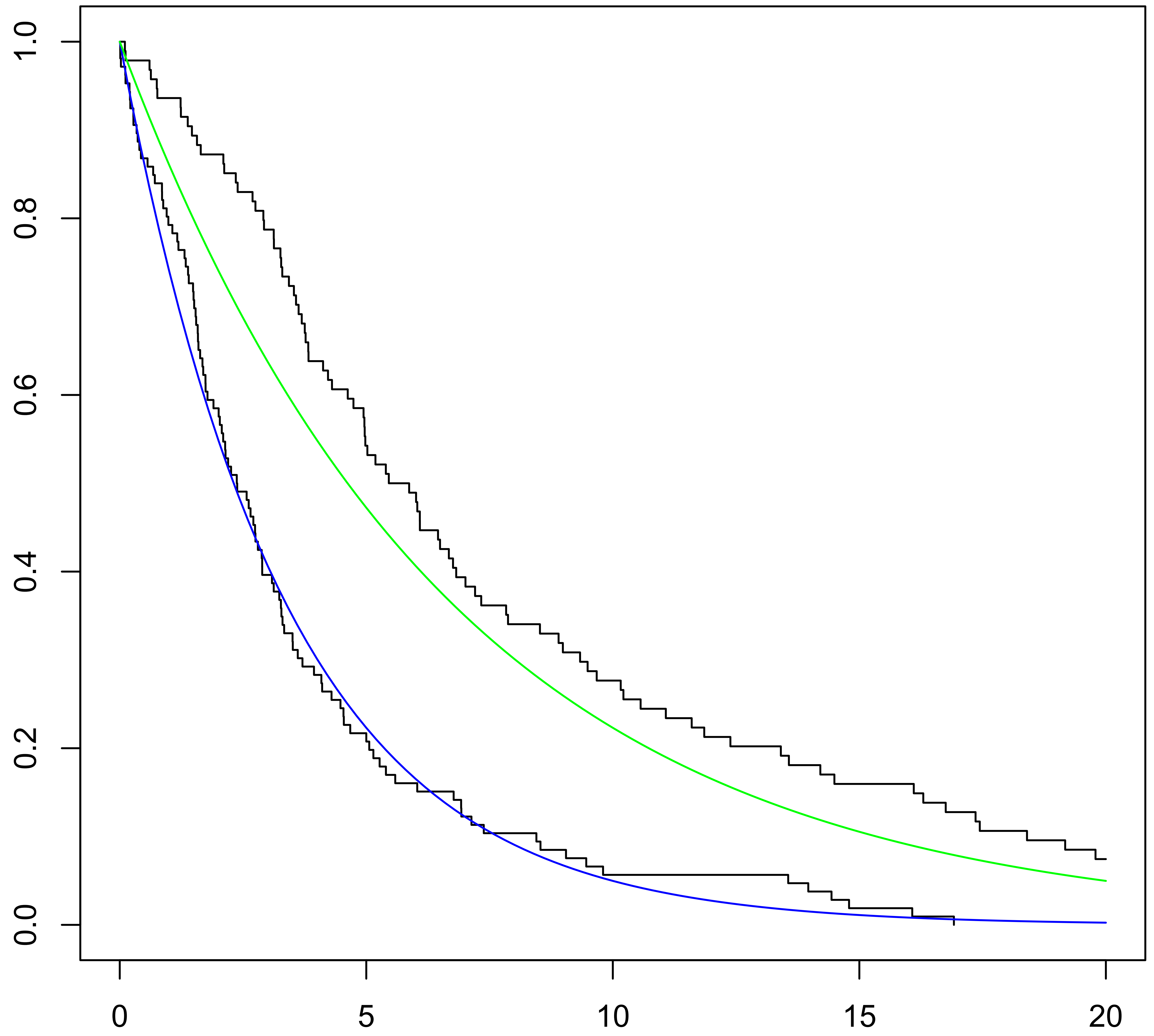
```
mu_lambda = 0.3
sigma_lambda = 0.05
beta = log(0.5)
censor_time = 15

N = 200
lambda = rnorm(N, mu_lambda, sigma_lambda)
treatment = sample(c(0, 1), N, replace = TRUE)
p = runif(N)
true_event_time = -log(1 - p) / (lambda * exp(treatment * beta))
censored = ifelse(true_event_time > censor_time, 1, 0)
event_time = pmin(true_event_time, censor_time)
```

- KM curve

```
Y = Surv(event_time)
plot(survfit(Y ~ treatment))
curve(exp(-x * lambda), add = T, col = 'blue')
curve(exp(-x * lambda * exp(beta)), add = T, col = 'green')
```

Hands on. Simulate data.

- KM curves



Hands-on. Stan program. 15 minutes

Extend the model

- Add data
    - `Nothing`
- Add parameters / change lambda
    - `real<lower = 0> mu_lambda;`
    - `real<lower = 0> sigma_lambda;`
    - `vector<lower = 0>[N] lambda;`
- Model

    1. Add priors for hyper parameters: (Note: these are somewhat informative)
       mu_lambda ~ normal(0, 0.1);
       sigma ~ normal(0, 0.1);

    2. Add hierarchical model:
       lambda ~ ...

    3. Update likelihood term:
       ```
       Original:  real lambda_n = lambda * exp(treatment[n] * beta);
       Updated:
       ```

Hands-on. Stan program. 15 minutes

Extend the model

- Model

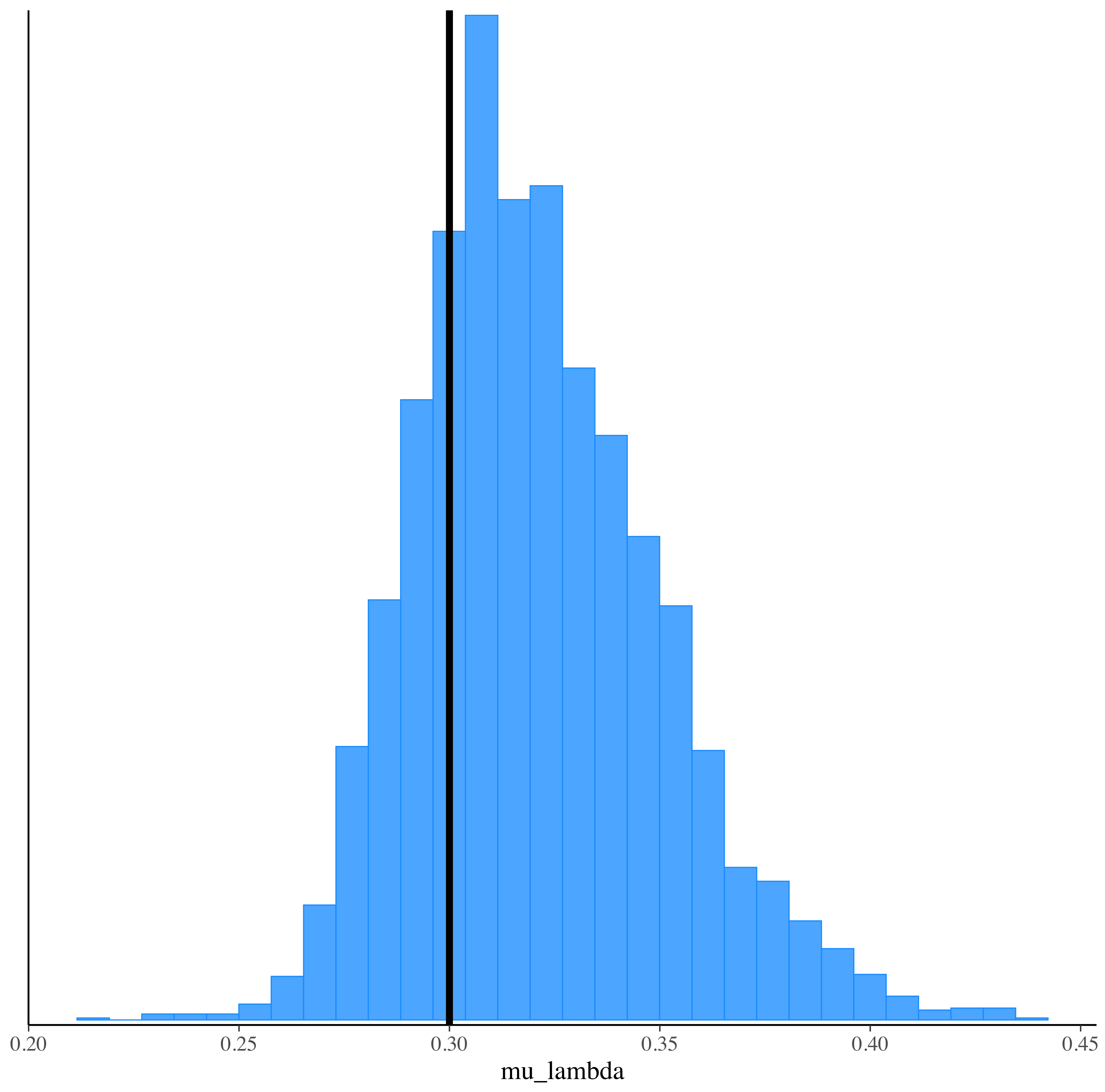
1. Add priors for hyper parameters: (Note: these are somewhat informative)
   mu_lambda ~ normal(0, 0.1);
   sigma ~ normal(0, 0.1);

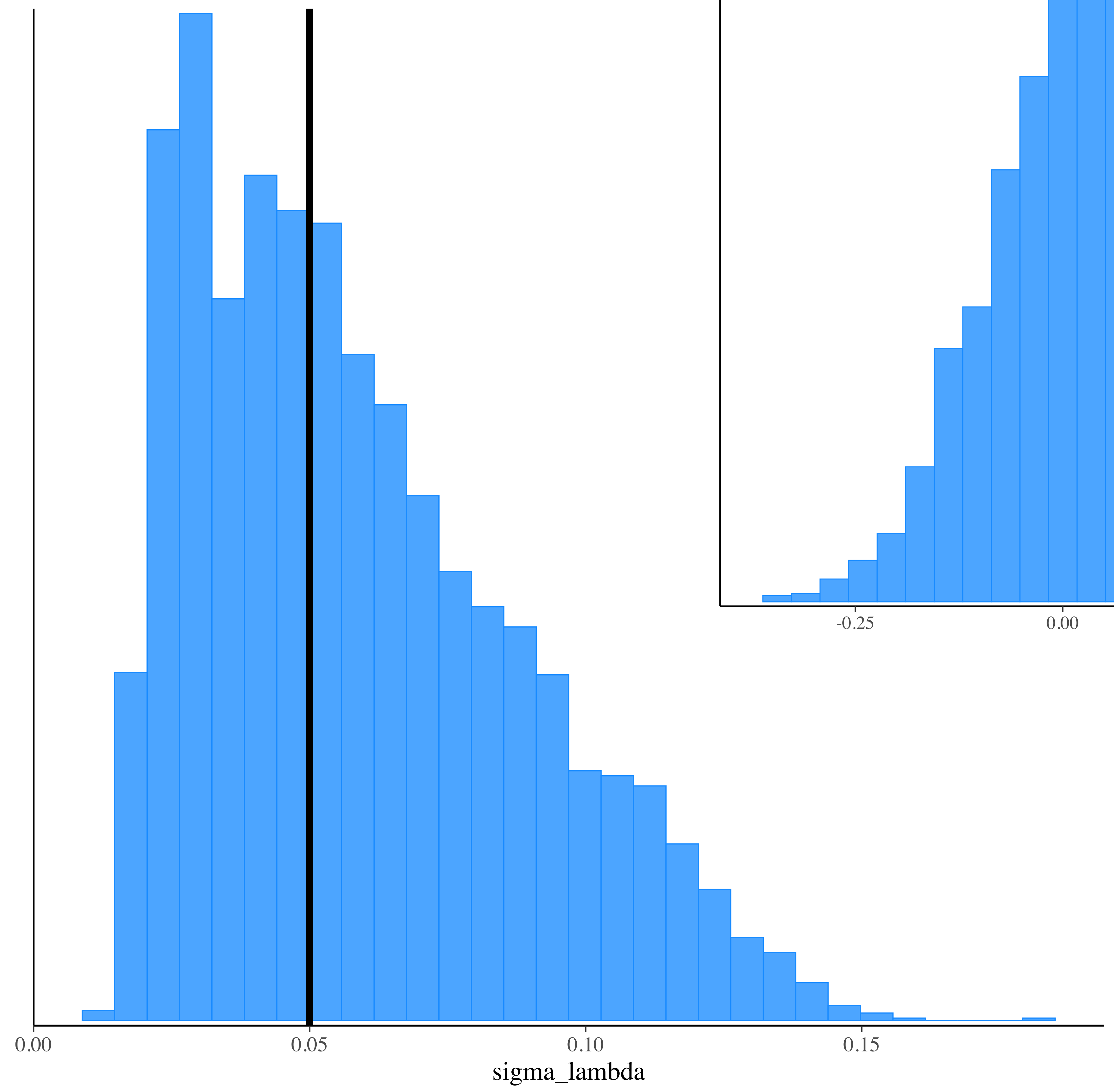
2. Add hierarchical model:
   lambda ~ normal(mu_lambda, sigma_lambda);

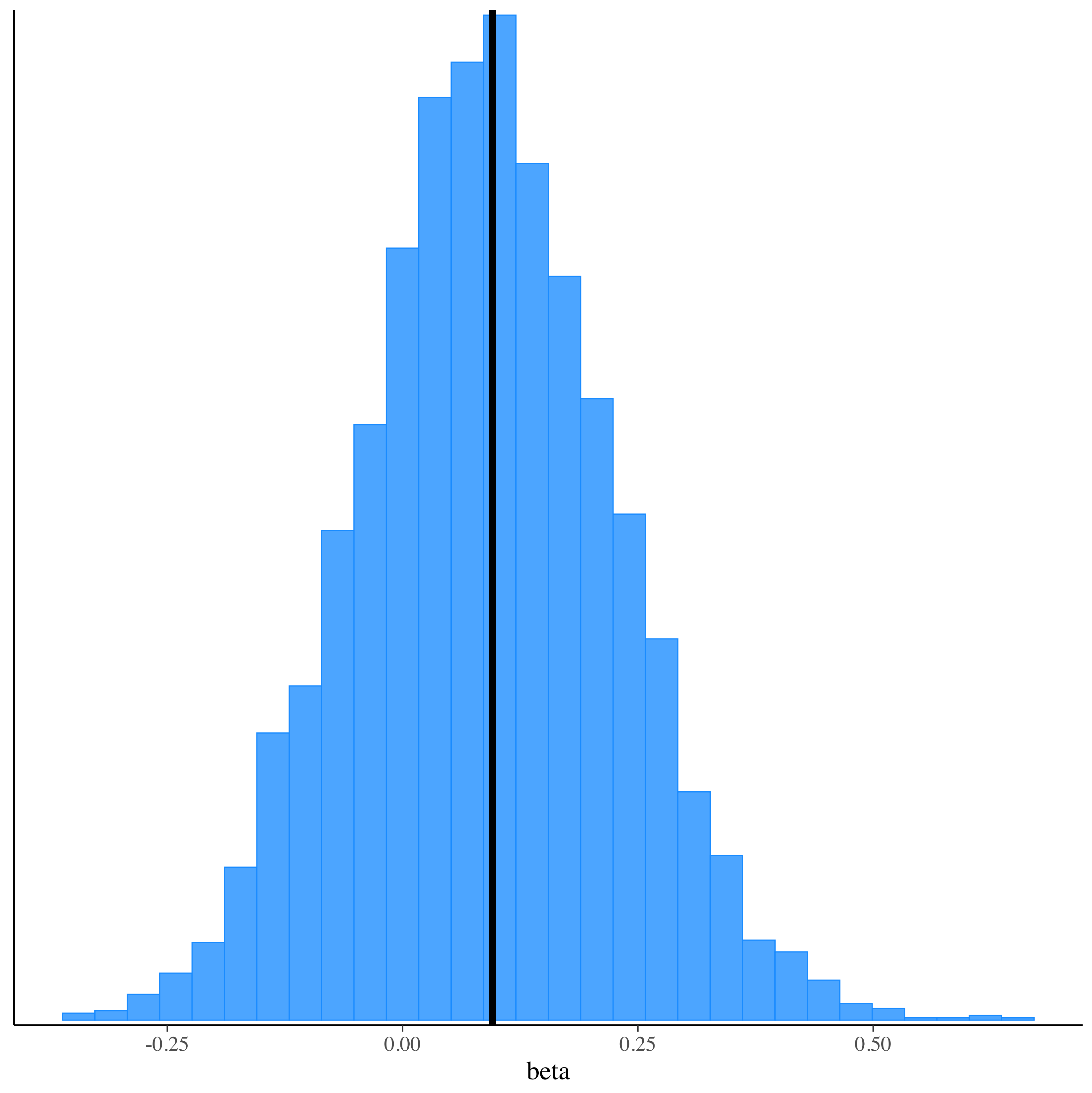
3. Update likelihood term:

```
Original:  real lambda_n = lambda * exp(treatment[n] *
beta);
Updated:   real lambda_n = lambda[n] * exp(treatment[n] *
beta);
```

Estimates for mu, sigma, beta







Problems

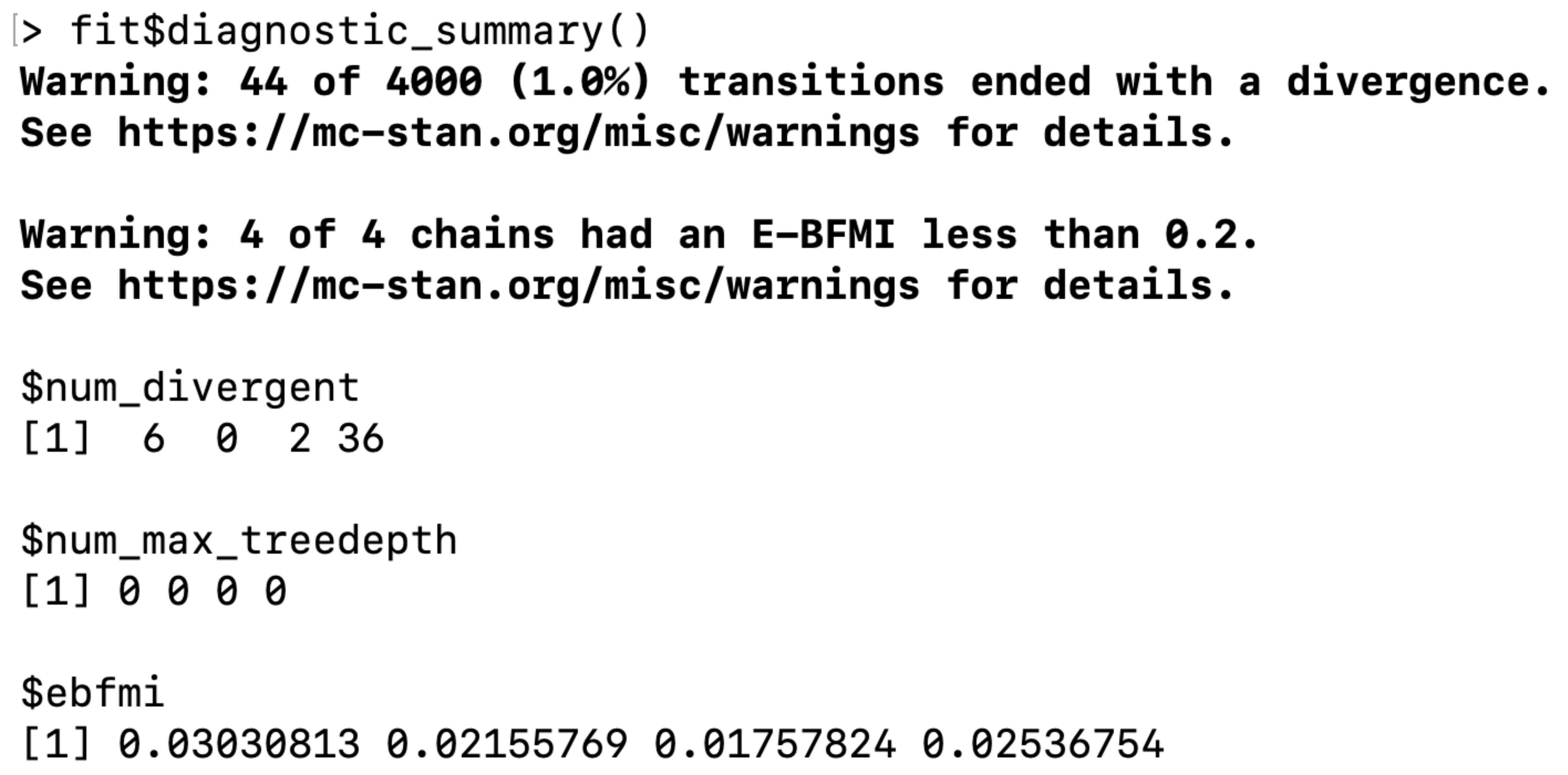
- Divergences: indicates problem estimating posterior

- In 44 iterations, the algorithm couldn't follow the trajectory.

- Why?

  - Posterior geometry:
    stepsize is estimated globally, high curvature vs low curvature

  - Easy to check if Hamiltonian is conserved (potential energy + kinetic energy)



```
> fit$diagnostic_summary()
Warning: 44 of 4000 (1.0%) transitions ended with a divergence.
See https://mc-stan.org/misc/warnings for details.

Warning: 4 of 4 chains had an E-BFMI less than 0.2.
See https://mc-stan.org/misc/warnings for details.

$num_divergent
[1]  6  0  2 36

$num_max_treedepth
[1] 0 0 0 0

$ebfmi
[1] 0.03030813 0.02155769 0.01757824 0.02536754
```

Potential Fixes

- Add argument to $sample()
  adapt_delta = 0.9 (or 0.99)
    - Asks algorithm to have a higher acceptance rate.
      Effectively smaller step size.
    - If this ends with max_treedepth issues, increase max_treedepth:
      max_treedepth = 14 (default = 10)
- Reparameterize model
  (this is for another day)

Recap

Recap

Four Examples:

1. Constant Hazard Model
2. Add Covariates
3. Add Censoring
4. Hierarchical Model

Skills:

- Simulating data
- Running CmdStanR
- Iterating over models
- Visualizing posterior distributions

Why Use Stan?

- Flexibility to write complex models
  - Time-varying hazard
  - Model covariance structures
  - Incorporate measurement models
  - Censoring / missingness models

Thank you!

Open Discussion

Contact

Feel free to reach out

• LinkedIn:	https://www.linkedin.com/in/syclik/
• Sports:	dlee@zelusanalytics.com
• All other:	daniel@bayesianops.com