Group Project 1

Huanyu Chen, Shaolei Ma, Ruiqi Xue

2024-02-19

Proportional-Hazard Assumption

Under proportional-hazards assumption, the hazard function (Cox model) can be written as:

$$h(t|x) = h_0(t)exp(\beta' x)$$

where t is the time, x the vector of covariates, β the vector of regression coefficients, $h_0(t)$ is the baseline hazard function. Then, the survival function is

$$S(t|x) = exp[-H_0(t)exp(\beta' x)]$$

where

$$H_0(t) = \int_0^t h_0(u)du$$

Thus, the distribution function is

$$F(t|x) = 1 - exp[-H_0(t)exp(\beta' x)]$$

Let Y be a random variable with distribution function F, then $U = F(Y) \sim U(0,1)$, $(1-U) \sim U(0,1)$, i.e.

$$U = exp[-H_0(t)exp(\beta' x)] \sim U(0,1)$$

if $h_0(t) > 0$ for all t, then H_0 can be inverted and the survival time T of the model can be written as

$$T = H_0^{-1}[-log(U)exp(-\beta' x)]$$

where $U \sim U(0,1)$.

To simply the problem, here we only consider one covariate x, which indicates whether the sample belongs to the control arm ($x = 0$) or the treatment arm ($x = 1$), and set a negative β under the assumption that the treatment has a consistent positive effect.

Now, we only need to know H_0^{-1} to simulate the survival time. To do so, we consider two commonly used survival time distributions: exponential and Weibull distribution.

For exponential distribution with scale parameter $\lambda > 0$, the possibility density function is

$$f_0 = \lambda exp(-\lambda t)$$

Then,

$$F_0(t) = 1 - exp(-\lambda t)$$

$$S_0(t) = 1 - F_0(t) = exp(-\lambda t)$$

$$H_0(t) = -log(S_0(t)) = \lambda t$$

$$h(t) = H_0'(t) = \lambda > 0$$

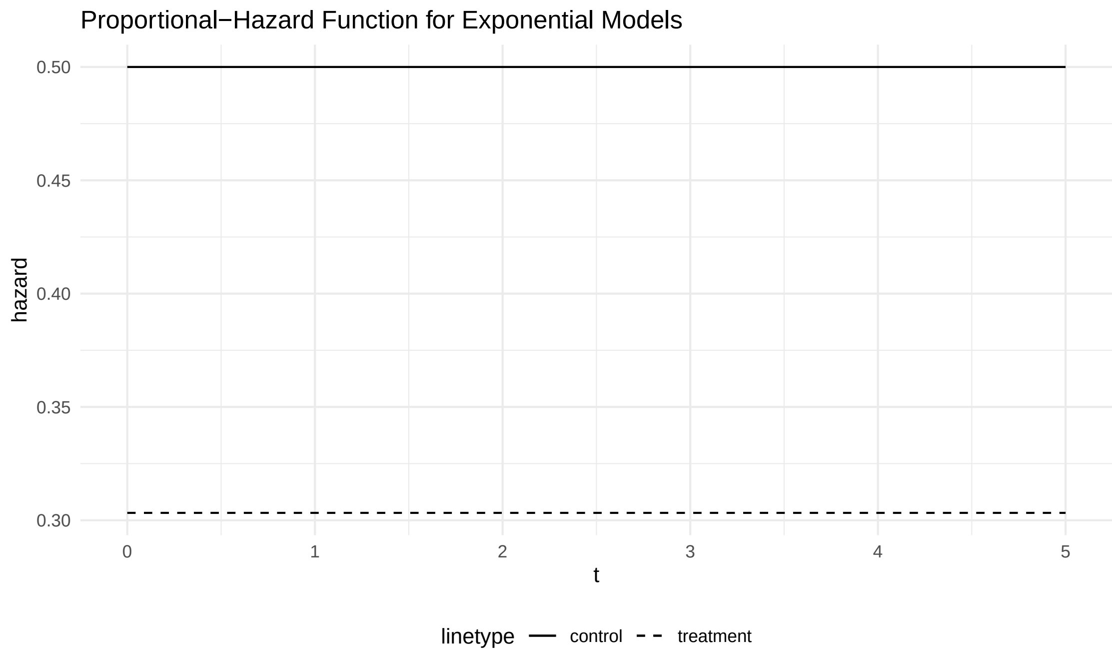
$$H_0^{-1}(t) = \lambda^{-1}t$$

Thus,

$$T = -\lambda^{-1}log(U)exp(-\beta'x)$$

where $U \sim U(0, 1)$.

The hazard function under $\lambda = 0.5$, $\beta = -0.5$ could be shown as follows:



For Weibull distribution with the scale parameter λ, and is the shape parameter γ, the possibility density function is

$$f_0 = \lambda\gamma t^{\gamma-1}exp(-\lambda t^{\gamma})$$

Then,

$$F_0(t) = 1 - exp(-\lambda t^{\gamma})$$

$$S_0(t) = 1 - F_0(t) = exp(-\lambda t^{\gamma})$$

$$H_0(t) = -log(S_0(t)) = \lambda t^{\gamma}$$

$$h(t) = H_0'(t) = \lambda\gamma t^{(\gamma-1)} > 0$$

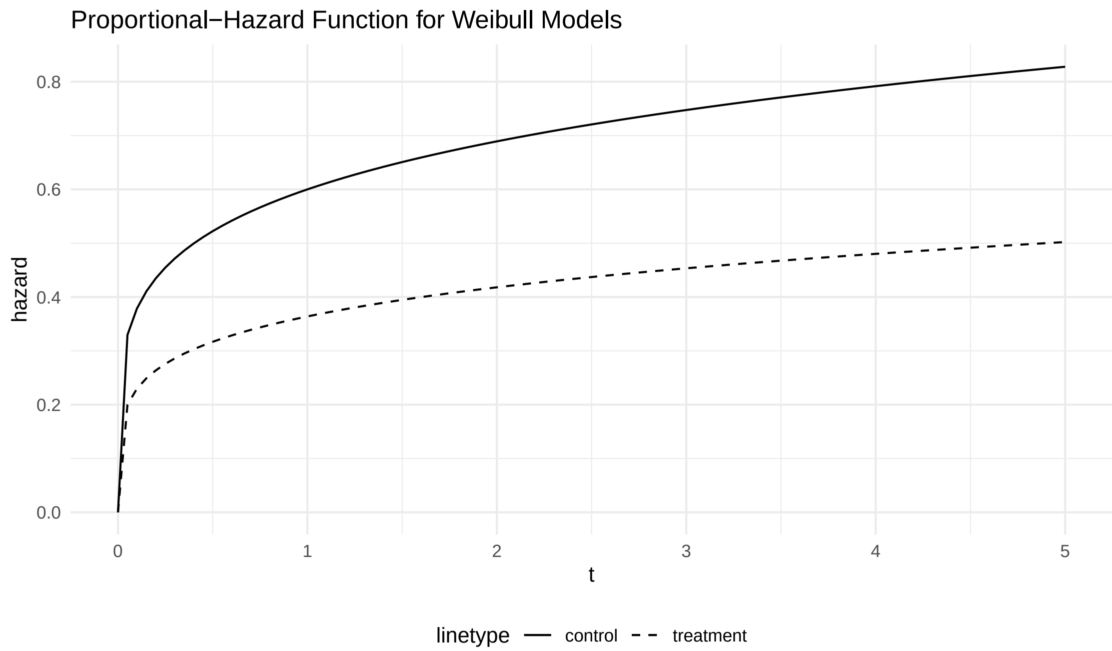
$$H_0^{-1}(t) = (\lambda^{-1}t)^{1/\gamma}$$

Thus,

$$T = (-\lambda^{-1}log(U)exp(-\beta'x))^{1/\gamma}$$

where $U \sim U(0, 1)$.

The hazard function under $\lambda = 0.5$, $\gamma = 1.2$, $\beta = -0.5$ could be shown as follows:



We can write the simulation process as follows:

```
ph_simulate_func = function(n, baseline, lambda, gamma = NULL, coveff)
{
  # Simulate treatment indicator variable
  x = rbinom(n = n, size = 1, prob = 0.5)
  # Draw from a U(0,1) random variable
  u = runif(n)
  # Simulate survival times depending on the baseline hazard
  if (baseline == "Exponential") {
    t = -log(u) / (lambda * exp(x * coveff))
    # Set the administrative censoring time to guarantee a censor rate of 0.2
    censor_time = qexp(0.8, rate = lambda)
  } else if(baseline == "Weibull") {
    t = (-log(u) / (lambda * exp(x * coveff)))^(1 / gamma)
    censor_time = qweibull(0.8, shape = gamma, scale = lambda)
  }
  # Make event indicator variable applying administrative censoring
  d = as.numeric(t < censor_time)
  t = pmin(t, censor_time)
  # Return a tibble object
  if (baseline == "Exponential") {
    return(tibble(x, t, d, n, baseline, lambda, coveff))
  } else if(baseline == "Weibull") {
    return(tibble(x, t, d, n, baseline, lambda, gamma, coveff))
  }
}
```

To observe the potential relevance between test performance and number of samples (n), parameter value (λ, γ), and coefficient β, we set $n = 50$, 100, 200, $\lambda = 0$, 0.5, 1, $\gamma = 1.2$, 1.5, and $\beta = 0$, 1, 2. We repeat 50 times for each value setting. The generation process is written as follows:

```r
exp_param_df = expand.grid(iteration = c(1:50), n = c(50, 100, 200),
            lambda = c(0.5, 0.8, 1), beta = c(-0.5, -1, -5))
wei_param_df = expand.grid(iteration = c(1:50), n = c(50, 100, 200),
            lambda = c(0.5, 0.8, 1), gamma = c(1.2, 1.5),
            beta = c(-0.5, -1, -5))

exp_results =
  mapply(ph_simulate_func, n = exp_param_df$n, baseline = "Exponential",
         lambda = exp_param_df$lambda, coveff = exp_param_df$beta)
wei_results =
  mapply(ph_simulate_func, n = wei_param_df$n, baseline = "Weibull",
         lambda = wei_param_df$lambda, gamma = wei_param_df$gamma,
         coveff = wei_param_df$beta)

ph_exp_df = tibble()
ph_wei_df = tibble()

for(i in 1:ncol(exp_results))
{
  a = exp_results[, i]
  ph_exp_df = cbind.data.frame(x = a$x, t = a$t, d = a$d, n = a$n,
                               baseline = "Exponential", lambda = a$lambda,
                               beta = a$coveff) |> as_tibble() |>
    nest(data = c(x : d)) |> rbind(ph_exp_df)
}

for(i in 1:ncol(wei_results))
{
  a = wei_results[, i]
  ph_wei_df =
    cbind.data.frame(x = a$x, t = a$t, d = a$d, n = a$n, baseline = "Weibull",
          lambda = a$lambda, gamma = a$gamma, beta = a$coveff) |>
    as_tibble() |> nest(data = c(x : d)) |> rbind(ph_wei_df)
}

ph_exp_df = ph_exp_df |> nest(simulations = c(data))
ph_wei_df = ph_wei_df |> nest(simulations = c(data))
```

Under different settings, we want to test the H_0 : there is no difference in survival between the treatment and control arm. Therefore, we use three different log-rank tests and compare the test power at the 0.05 significance level.

```r
pwr_func = function(list_df, n = 50)
{
  test1_reject = 0
  test2_reject = 0
  test3_reject = 0
  for(j in 1:nrow(list_df)) {
    dat = list_df |> slice(j) |> unnest(cols = c(data))
    test_results = logrank.maxtest(dat$t, dat$d, dat$x)
    test1_reject = test1_reject +
      ((test_results$tests |> filter(Test == 1) |> pull(p)) < 0.05)
    test2_reject = test2_reject +
```

```r
      ((test_results$tests |> filter(Test == 2) |> pull(p)) < 0.05)
    test3_reject = test3_reject +
      ((test_results$tests |> filter(Test == 3) |> pull(p)) < 0.05)
  }
  return(
    tibble(
      test1_power = test1_reject / n,
      test2_power = test2_reject / n,
      test3_power = test3_reject / n
    )
  )
}
(ph_exp_df =
  ph_exp_df |> mutate(power = map(simulations, pwr_func)) |> unnest(power))
```

```
## # A tibble: 27 x 8
##        n baseline   lambda  beta simulations test1_power test2_power test3_power
##    <dbl> <chr>       <dbl> <dbl> <list>            <dbl>       <dbl>       <dbl>
##  1   200 Exponenti~    1      -5 <tibble>              1        1              1
##  2   100 Exponenti~    1      -5 <tibble>              1        1              1
##  3    50 Exponenti~    1      -5 <tibble>              1        1              1
##  4   200 Exponenti~    0.8    -5 <tibble>              1        1              1
##  5   100 Exponenti~    0.8    -5 <tibble>              1        1              1
##  6    50 Exponenti~    0.8    -5 <tibble>              1        1              1
##  7   200 Exponenti~    0.5    -5 <tibble>              1        1              1
##  8   100 Exponenti~    0.5    -5 <tibble>              1        1              1
##  9    50 Exponenti~    0.5    -5 <tibble>              1        1              1
## 10   200 Exponenti~    1      -1 <tibble>              1        0.98           1
## # i 17 more rows
```

```r
(ph_wei_df =
  ph_wei_df |> mutate(power = map(simulations, pwr_func)) |> unnest(power))
```

```
## # A tibble: 54 x 9
##        n baseline lambda gamma  beta simulations       test1_power test2_power
##    <dbl> <chr>     <dbl> <dbl> <dbl> <list>                  <dbl>       <dbl>
##  1   200 Weibull     1     1.5    -5 <tibble [50 x 1]>        1           1
##  2   100 Weibull     1     1.5    -5 <tibble [50 x 1]>        1           1
##  3    50 Weibull     1     1.5    -5 <tibble [50 x 1]>        1           1
##  4   200 Weibull     0.8   1.5    -5 <tibble [50 x 1]>        1           1
##  5   100 Weibull     0.8   1.5    -5 <tibble [50 x 1]>        1           1
##  6    50 Weibull     0.8   1.5    -5 <tibble [50 x 1]>        1           1
##  7   200 Weibull     0.5   1.5    -5 <tibble [50 x 1]>        1           1
##  8   100 Weibull     0.5   1.5    -5 <tibble [50 x 1]>        1           1
##  9    50 Weibull     0.5   1.5    -5 <tibble [50 x 1]>        0.82        0.6
## 10   200 Weibull     1     1.2    -5 <tibble [50 x 1]>        1           1
## # i 44 more rows
## # i 1 more variable: test3_power <dbl>
```

Non-Proportional-Hazard Assumption

Under NPH assumption, we still consider the exponential model and weibull model.

Piecewise Exponential Model

To simplify the problem, we set the baseline hazard function to be a constant $\lambda_0 = 0.5$, which indicates that the survival time for the control arm follows exponential distribution.

Late Effect

For the treatment arm, we suppose the hazard function for the treatment arm is:

$$h(t|x=1) = \begin{cases} \lambda_0 & t < 1 \\ \lambda_1 & t \geq 1 \end{cases}$$

Then,

$$H(t|x=1) = \begin{cases} \lambda_0 t & t < 1 \\ (\lambda_0 + \lambda_1)t - \lambda_1 & t \geq 1 \end{cases}$$

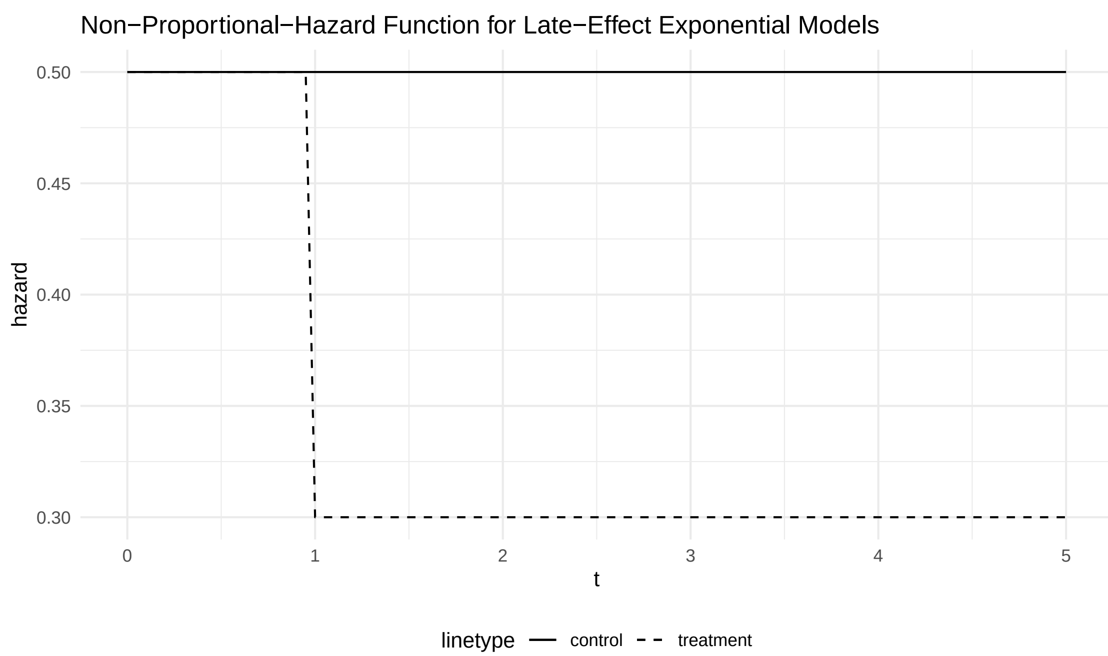
$$S(t|x=1) = exp(-H(t|x=1)) = \begin{cases} exp(-\lambda_0 t) & t < 1 \\ exp(-(\lambda_0 + \lambda_1)t + \lambda_1) & t \geq 1 \end{cases}$$

$$F(t|x=1) = 1 - S(t|x=1) = \begin{cases} 1 - exp(-\lambda_0 t) & t < 1 \\ 1 - exp(-(\lambda_0 + \lambda_1)t + \lambda_1) & t \geq 1 \end{cases}$$

Let $1 - U = F(t|x=1)$, then $(1-U) \sim U(0,1)$, $U = S(t|x=1) \sim U(0,1)$. Thus,

$$T = \begin{cases} -\lambda_0^{-1} log(U) & U > exp(-\lambda_0) \\ \dfrac{\lambda_1 - log(U)}{\lambda_0 + \lambda_1} & U \leq exp(-\lambda_0) \end{cases}$$

The hazard function under $\lambda_0 = 0.5$, $\lambda_1 = 0.3$ could be shown as follows:



With the distribution function of survival times, we can write the simulation process as follows (note: for early effect piecewise models, the expression for all functions are similar except for the definition domains, so the simulation process is similar and we write it down as well.)

```r
piecewise_sim_func = function(n, lambda0 = 0.5, lambda1, type)
{
  # Set the administrative censoring time to guarantee a censor rate of 0.2 for control arm
  censor_time = qexp(0.8, rate = lambda0)

  u0 = runif(n)
  t0 = - log(u0) / lambda0
  u1 = runif(n)
  if(type == "late")
    t1 = (u1 > exp(-lambda0)) * (-log(u1) / lambda0) +
    (u1 <= exp(-lambda0)) * ((lambda1 - log(u1)) / (lambda0 + lambda1))
  else if(type == "early")
    t1 = (u1 <= exp(-lambda0)) * (-log(u1) / lambda0) +
    (u1 > exp(-lambda0)) * ((lambda1 - log(u1)) / (lambda0 + lambda1))

  # Make event indicator variable applying administrative censoring
  d0 = as.numeric(t0 < censor_time)
  d1 = as.numeric(t1 < censor_time)
  t0 = pmin(t0, censor_time)
  t1 = pmin(t1, censor_time)

  control_df = tibble(x = rep(0, n), t = t0, d = d0, n, lambda0, lambda1)
  treat_df = tibble(x = rep(1, n), t = t1, d = d1, n, lambda0, lambda1)
  return(rbind(control_df, treat_df))
}
```

```r
late_pw_param_df = expand.grid(iteration = c(1:50), n = c(50, 100, 200),
            lambda0 = c(0.5, 0.8), lambda1 = c(0.3, 0.4))

late_pw_results =
  mapply(piecewise_sim_func, n = late_pw_param_df$n,
         lambda0 = late_pw_param_df$lambda0, lambda1 = late_pw_param_df$lambda1,
         type = "late")

late_pw_df = tibble()

for(i in 1:ncol(late_pw_results))
{
  a = late_pw_results[, i]
  late_pw_df = cbind.data.frame(x = a$x, t = a$t, d = a$d, n = a$n,
                                lambda0 = a$lambda0, lambda1 = a$lambda1) |>
    as_tibble() |> nest(data = c(x : d)) |> rbind(late_pw_df)
}

late_pw_df = late_pw_df |> nest(simulations = c(data))
```

Under different settings, we want to test the H_0 : there is no difference in survival between the treatment and control arm. Therefore, we use three different log-rank tests and compare the test power at the 0.05 significance level.

```r
(late_pw_df =
  late_pw_df |> mutate(power = map(simulations, pwr_func)) |> unnest(power))
```

```
## # A tibble: 12 x 7
##        n lambda0 lambda1 simulations       test1_power test2_power test3_power
##    <dbl>   <dbl>   <dbl> <list>                  <dbl>       <dbl>       <dbl>
##  1   200     0.8     0.4 <tibble [50 x 1]>        0.34        0.62        0.1
##  2   100     0.8     0.4 <tibble [50 x 1]>        0.16        0.36        0.08
##  3    50     0.8     0.4 <tibble [50 x 1]>        0.02        0.1         0.02
##  4   200     0.5     0.4 <tibble [50 x 1]>        0.82        0.96        0.34
##  5   100     0.5     0.4 <tibble [50 x 1]>        0.44        0.72        0.14
##  6    50     0.5     0.4 <tibble [50 x 1]>        0.3         0.48        0.14
##  7   200     0.8     0.3 <tibble [50 x 1]>        0.14        0.32        0.02
##  8   100     0.8     0.3 <tibble [50 x 1]>        0.12        0.14        0
##  9    50     0.8     0.3 <tibble [50 x 1]>        0.12        0.1         0.1
## 10   200     0.5     0.3 <tibble [50 x 1]>        0.68        0.88        0.3
## 11   100     0.5     0.3 <tibble [50 x 1]>        0.32        0.46        0.2
## 12    50     0.5     0.3 <tibble [50 x 1]>        0.28        0.28        0.2
```

Early Effect

We can use the similar simulation method to generate piecewise exponential models in which the treatment arm shows early effect. The hazard function becomes:

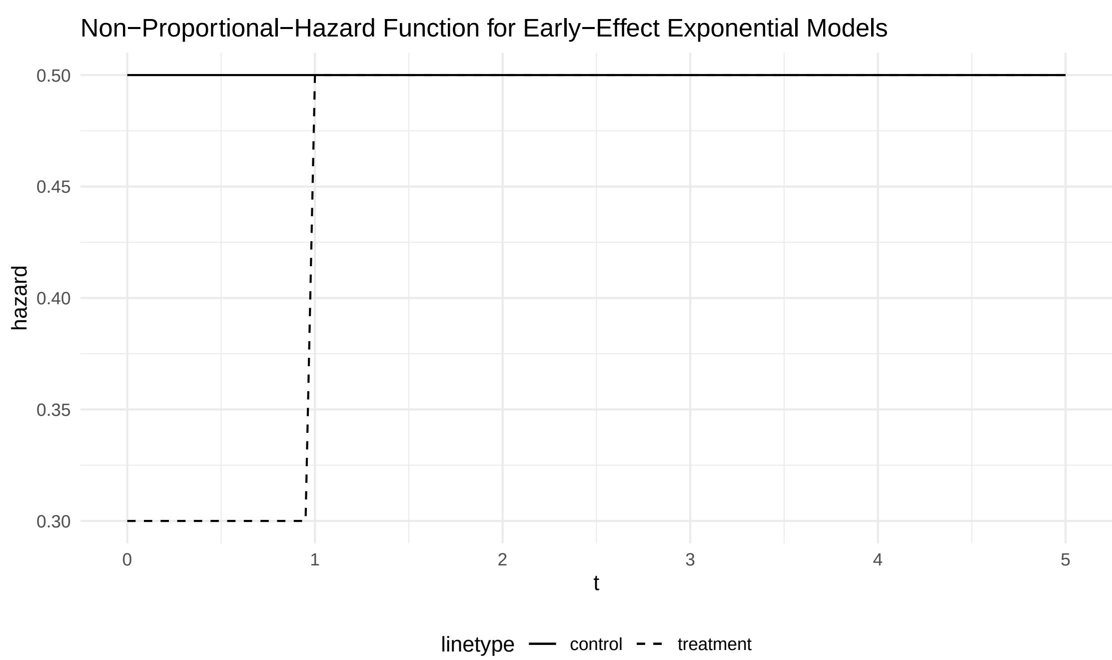
$$h(t|x=1) = \begin{cases} \lambda_0 & t \geq 1 \\ \lambda_1 & t < 1 \end{cases}$$

Similarly, it can be derived that

$$T = \begin{cases} -\lambda_0^{-1} log(U) & U \leq exp(-\lambda_0) \\ \dfrac{\lambda_1 - log(U)}{\lambda_0 + \lambda_1} & U > exp(-\lambda_0) \end{cases}$$

where $U \sim U(0,1)$.

The hazard function under $\lambda_0 = 0.5$, $\lambda_1 = 0.3$ could be shown as follows:



```r
early_pw_param_df = expand.grid(iteration = c(1:50), n = c(50, 100, 200),
            lambda0 = c(0.5, 0.8), lambda1 = c(0.3, 0.4))

early_pw_results =
  mapply(piecewise_sim_func, n = early_pw_param_df$n,
         lambda0 = early_pw_param_df$lambda0, lambda1 = early_pw_param_df$lambda1,
         type = "early")

early_pw_df = tibble()

for(i in 1:ncol(early_pw_results))
{
  a = early_pw_results[, i]
  early_pw_df = cbind.data.frame(x = a$x, t = a$t, d = a$d, n = a$n,
                                 lambda0 = a$lambda0, lambda1 = a$lambda1) |>
    as_tibble() |> nest(data = c(x : d)) |> rbind(early_pw_df)
}

early_pw_df = early_pw_df |> nest(simulations = c(data))
```

Under different settings, we want to test the H_0 : there is no difference in survival between the treatment and control arm. Therefore, we use three different log-rank tests and compare the test power at the 0.05 significance level.

```r
(early_pw_df =
  early_pw_df |> mutate(power = map(simulations, pwr_func)) |> unnest(power))
```

```
## # A tibble: 12 x 7
##        n lambda0 lambda1 simulations       test1_power test2_power test3_power
##    <dbl>   <dbl>   <dbl> <list>                  <dbl>       <dbl>       <dbl>
##  1   200     0.8     0.4 <tibble [50 x 1]>        0.18        0.14        0.64
##  2   100     0.8     0.4 <tibble [50 x 1]>        0.08        0.12        0.3
##  3    50     0.8     0.4 <tibble [50 x 1]>        0.12        0.1         0.26
##  4   200     0.5     0.4 <tibble [50 x 1]>        0.02        0.02        0.36
##  5   100     0.5     0.4 <tibble [50 x 1]>        0.08        0.16        0.2
##  6    50     0.5     0.4 <tibble [50 x 1]>        0.06        0.02        0.08
##  7   200     0.8     0.3 <tibble [50 x 1]>        0.08        0.1         0.42
##  8   100     0.8     0.3 <tibble [50 x 1]>        0.1         0.1         0.26
##  9    50     0.8     0.3 <tibble [50 x 1]>        0.12        0.1         0.2
## 10   200     0.5     0.3 <tibble [50 x 1]>        0.04        0.18        0.12
## 11   100     0.5     0.3 <tibble [50 x 1]>        0.08        0.08        0.14
## 12    50     0.5     0.3 <tibble [50 x 1]>        0.08        0.06        0.08
```

Weibull Model

To simplify the problem, we assume the control and treatment arm share the same scale parameter λ. For the control arm, suppose the hazard function is:

$$h(t|x=0) = \lambda\gamma_0 t^{(\gamma_0 - 1)}.$$

Then,

$$H(t|x=0) = \lambda t_0^{\gamma}$$

$$S(t|x=0) = exp(-H(t|x=0)) = exp(-\lambda t_0^{\gamma})$$
$$F(t|x=0) = 1 - S(t|x=0) = 1 - exp(-\lambda t_0^{\gamma})$$

Let $1-U = F(t|x=0)$, then $(1-U) \sim U(0,1)$, $U = S(t|x=0) \sim U(0,1)$. Thus,

$$T = (-\lambda^{-1} log(U))^{1/\gamma_0}$$

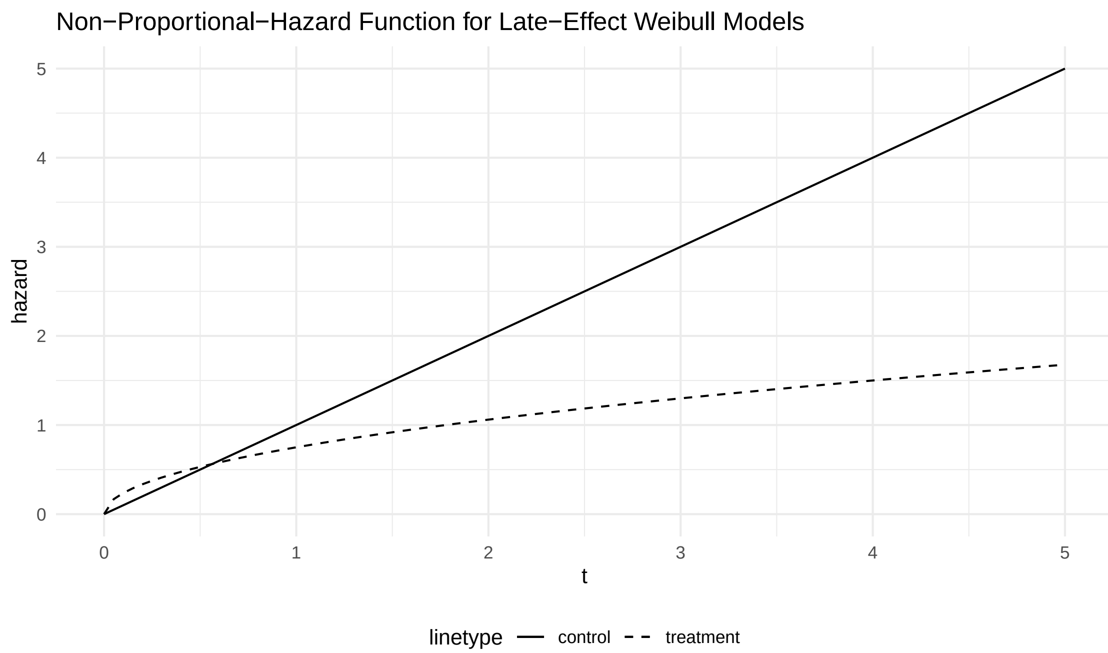
Similarly, we can write the hazard function for the treatment arm as:

$$h(t|x=1) = \lambda \gamma_1 t^{(\gamma_1 - 1)}.$$

We can derive that

$$T = (-\lambda^{-1} log(U))^{1/\gamma_1}$$

The hazard function under $\lambda = 0.5$, $\gamma_0 = 2$, $\gamma_1 = 1.5$ could be shown as follows:



With the distribution function of survival times, we can write the simulation process as follows.

```
weibull_sim_func = function(n, lambda = 0.5, gamma0, gamma1)
{
  # Set the administrative censoring time to guarantee a censor rate of 0.2 for control arm
  censor_time = qweibull(0.8, shape = gamma0, scale = lambda)

  u0 = runif(n)
  t0 = (- log(u0) / lambda) ^ (1 / gamma0)
  u1 = runif(n)
  t1 = (- log(u1) / lambda) ^ (1 / gamma1)

  # Make event indicator variable applying administrative censoring
  d0 = as.numeric(t0 < censor_time)
  d1 = as.numeric(t1 < censor_time)
  t0 = pmin(t0, censor_time)
  t1 = pmin(t1, censor_time)
```

```r
  control_df = tibble(x = rep(0, n), t = t0, d = d0,
                      n, lambda, gamma0, gamma1)
  treat_df = tibble(x = rep(1, n), t = t1, d = d1,
                    n, lambda, gamma0, gamma1)
  return(rbind(control_df, treat_df))
}
```

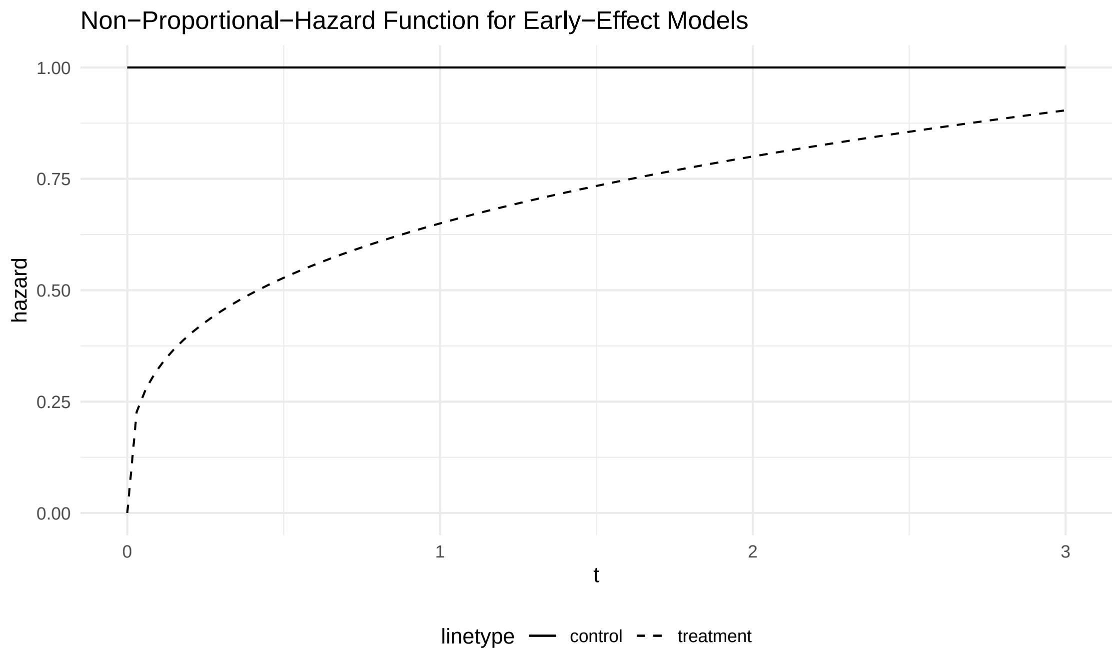
```r
late_wei_param_df =
  expand.grid(iteration = c(1:50), n = c(50, 100, 200),
              lambda = c(0.5, 0.8), gamma0 = c(2, 3),
              gamma1 = c(1.5, 1.8))

late_wei_results =
  mapply(weibull_sim_func, n = late_wei_param_df$n,
         lambda = late_wei_param_df$lambda,
         gamma0 = late_wei_param_df$gamma0, gamma1 = late_wei_param_df$gamma1)

late_wei_df = tibble()

for(i in 1:ncol(late_wei_results))
{
  a = late_wei_results[, i]
  late_wei_df = cbind.data.frame(x = a$x, t = a$t, d = a$d, n = a$n,
                                 lambda = a$lambda, gamma0 = a$gamma0,
                                 gamma1 = a$gamma1) |>
    as_tibble() |> nest(data = c(x : d)) |> rbind(late_wei_df)
}

late_wei_df = late_wei_df |> nest(simulations = c(data))
```

Under different settings, we want to test the H_0 : there is no difference in survival between the treatment and control arm. Therefore, we use three different log-rank tests and compare the test power at the 0.05 significance level.

```r
(late_wei_df =
  late_wei_df |> mutate(power = map(simulations, pwr_func)) |> unnest(power))
```

```
## # A tibble: 24 x 8
##        n lambda gamma0 gamma1 simulations test1_power test2_power test3_power
##    <dbl>  <dbl>  <dbl>  <dbl> <list>            <dbl>       <dbl>       <dbl>
##  1   200    0.8      3    1.8 <tibble>           0.28        0.04        0.42
##  2   100    0.8      3    1.8 <tibble>           0.12        0.04        0.22
##  3    50    0.8      3    1.8 <tibble>           0.12        0.02        0.18
##  4   200    0.5      3    1.8 <tibble>           0.68        0.24        0.7
##  5   100    0.5      3    1.8 <tibble>           0.42        0.18        0.44
##  6    50    0.5      3    1.8 <tibble>           0.28        0.14        0.3
##  7   200    0.8      2    1.8 <tibble>           0.06        0.04        0.06
##  8   100    0.8      2    1.8 <tibble>           0.04        0.08        0.04
##  9    50    0.8      2    1.8 <tibble>           0.08        0.1         0.06
## 10   200    0.5      2    1.8 <tibble>           0.06        0.06        0.06
## # i 14 more rows
```

If set the survival time for the control arm to follow exponential distribution, then the hazard function under $\lambda = 1$, $\gamma = 1.3$ could be shown as follows:



References

Bender, R., Augustin, T., & Blettner, M. (2005). Generating survival times to simulate Cox proportional hazards models. *Statistics in medicine, 24*(11), 1713–1723. https://doi.org/10.1002/sim.2059 Austin P. C. (2012). Generating survival times to simulate Cox proportional hazards models with time-varying covariates. *Statistics in medicine, 31*(29), 3946–3958. https://doi.org/10.1002/sim.5452