Group Project 1

Huanyu Chen, Shaolei Ma, Ruiqi Xue

2024-02-25

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,

$$F_0(t) = 1 - exp(-\lambda t)$$
$$S_0(t) = 1 - F_0(t) = exp(-\lambda t)$$
$$H_0(t) = -log(S_0(t)) = \lambda t$$
$$h(t) = H_0'(t) = \lambda > 0$$

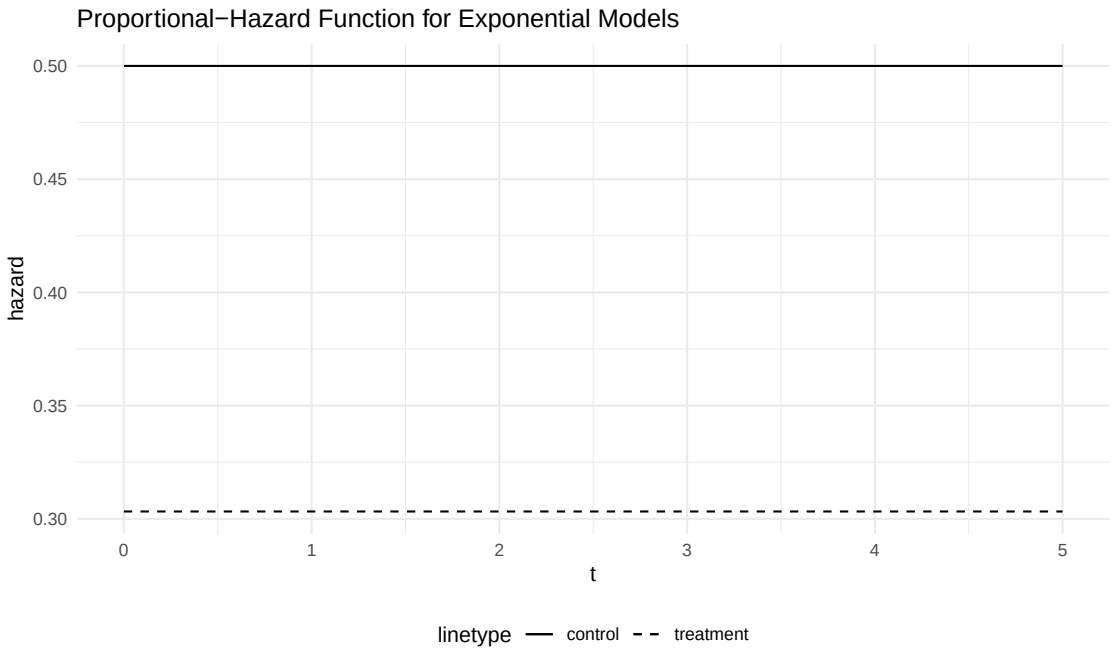
$$H_0^{-1}(t) = \lambda^{-1}t$$

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,

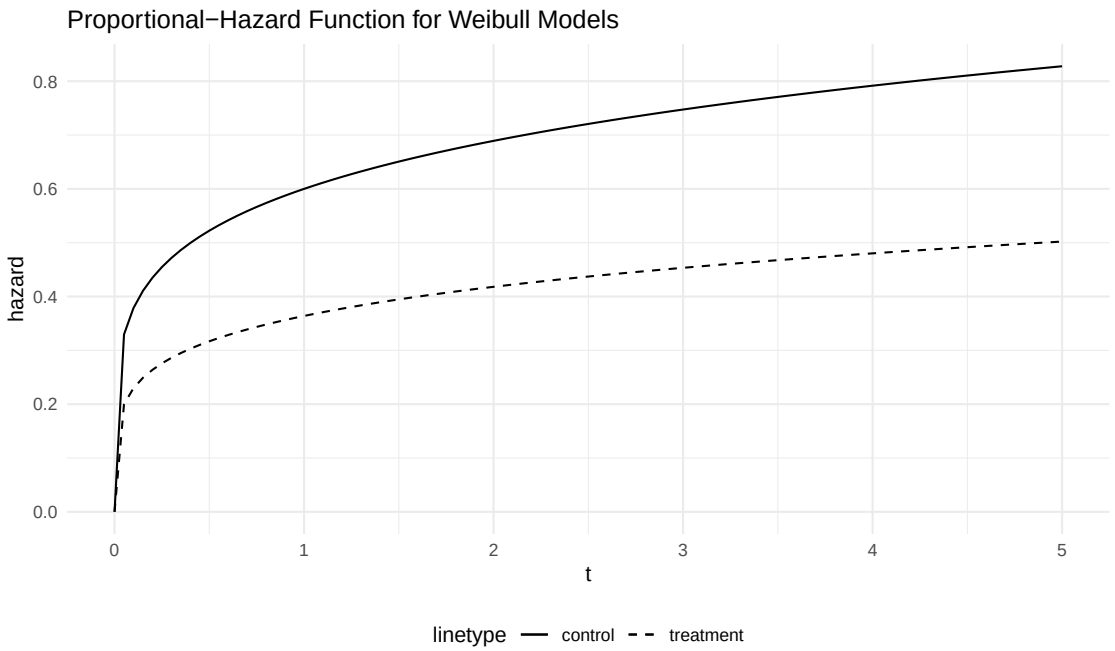
$$F_0(t) = 1 - exp(-\lambda t^{\gamma})$$
$$S_0(t) = 1 - F_0(t) = exp(-\lambda t^{\gamma})$$
$$H_0(t) = -log(S_0(t)) = \lambda t^{\gamma}$$
$$h(t) = H_0'(t) = \lambda\gamma t^{(\gamma-1)} > 0$$
$$H_0^{-1}(t) = (\lambda^{-1}t)^{1/\gamma}$$

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:

```r
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 = 1 / 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:

```
exp_param_df = expand.grid(iteration = c(1:50), n = c(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(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.

```
specif_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 +
```

```
      ((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_specificity = test1_reject / n,
      test2_specificity = test2_reject / n,
      test3_specificity = test3_reject / n
    )
  )
}


ph_exp_df = ph_exp_df |>
  mutate(specificity = map(simulations, specif_func)) |>
  unnest(specificity) |>
  select(-simulations)


ph_wei_df = ph_wei_df |>
  mutate(specificity = map(simulations, specif_func)) |>
  unnest(specificity) |>
  select(-simulations)
```

```r
kable(ph_exp_df, caption = "Specificity of 3 Log-Rank Tests based on PH Assumption")
```

Table 1: Specificity of 3 Log-Rank Tests based on PH Assumption

n	baseline	lambda	beta	test1_specificity	test2_specificity	test3_specificity
200	Exponential	1.0	-5.0	1.00	1.00	1.00
100	Exponential	1.0	-5.0	1.00	1.00	1.00
200	Exponential	0.8	-5.0	1.00	1.00	1.00
100	Exponential	0.8	-5.0	1.00	1.00	1.00
200	Exponential	0.5	-5.0	1.00	1.00	1.00
100	Exponential	0.5	-5.0	1.00	1.00	1.00
200	Exponential	1.0	-1.0	1.00	1.00	1.00
100	Exponential	1.0	-1.0	0.94	0.96	0.94
200	Exponential	0.8	-1.0	1.00	1.00	1.00
100	Exponential	0.8	-1.0	0.98	0.92	0.98
200	Exponential	0.5	-1.0	1.00	1.00	1.00
100	Exponential	0.5	-1.0	0.98	0.94	0.96
200	Exponential	1.0	-0.5	0.90	0.68	0.84
100	Exponential	1.0	-0.5	0.60	0.52	0.54
200	Exponential	0.8	-0.5	0.80	0.74	0.74
100	Exponential	0.8	-0.5	0.64	0.48	0.58
200	Exponential	0.5	-0.5	0.84	0.74	0.74
100	Exponential	0.5	-0.5	0.62	0.46	0.60

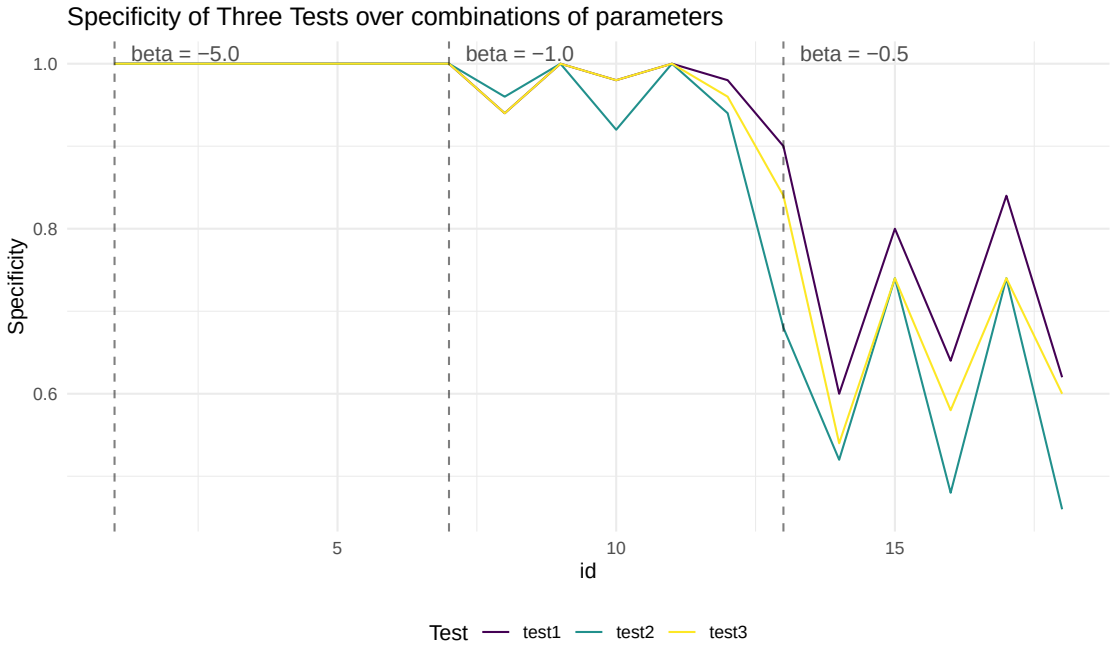
```
kable(ph_wei_df, caption = "Specificity of 3 Log-Rank Tests based on PH Assumption")
```

Table 2: Specificity of 3 Log-Rank Tests based on PH Assumption

n	baseline	lambda	gamma	beta	test1_specificity	test2_specificity	test3_specificity
200	Weibull	1.0	1.5	-5.0	1.00	1.00	1.00
100	Weibull	1.0	1.5	-5.0	1.00	1.00	1.00
200	Weibull	0.8	1.5	-5.0	1.00	1.00	1.00
100	Weibull	0.8	1.5	-5.0	1.00	1.00	1.00
200	Weibull	0.5	1.5	-5.0	1.00	1.00	1.00
100	Weibull	0.5	1.5	-5.0	1.00	1.00	1.00
200	Weibull	1.0	1.2	-5.0	1.00	1.00	1.00
100	Weibull	1.0	1.2	-5.0	1.00	1.00	1.00
200	Weibull	0.8	1.2	-5.0	1.00	1.00	1.00
100	Weibull	0.8	1.2	-5.0	1.00	1.00	1.00
200	Weibull	0.5	1.2	-5.0	1.00	1.00	1.00
100	Weibull	0.5	1.2	-5.0	1.00	1.00	1.00
200	Weibull	1.0	1.5	-1.0	1.00	1.00	1.00
100	Weibull	1.0	1.5	-1.0	0.96	0.94	0.94
200	Weibull	0.8	1.5	-1.0	1.00	0.98	1.00
100	Weibull	0.8	1.5	-1.0	0.98	0.94	0.98
200	Weibull	0.5	1.5	-1.0	1.00	1.00	1.00
100	Weibull	0.5	1.5	-1.0	0.96	0.98	0.96
200	Weibull	1.0	1.2	-1.0	1.00	1.00	1.00
100	Weibull	1.0	1.2	-1.0	0.98	0.96	0.98
200	Weibull	0.8	1.2	-1.0	1.00	1.00	1.00
100	Weibull	0.8	1.2	-1.0	0.98	0.96	0.98
200	Weibull	0.5	1.2	-1.0	1.00	1.00	1.00
100	Weibull	0.5	1.2	-1.0	1.00	1.00	0.98
200	Weibull	1.0	1.5	-0.5	0.86	0.76	0.78
100	Weibull	1.0	1.5	-0.5	0.54	0.40	0.52
200	Weibull	0.8	1.5	-0.5	0.80	0.76	0.76
100	Weibull	0.8	1.5	-0.5	0.52	0.48	0.52
200	Weibull	0.5	1.5	-0.5	1.00	0.88	0.92
100	Weibull	0.5	1.5	-0.5	0.52	0.36	0.54
200	Weibull	1.0	1.2	-0.5	0.94	0.82	0.86
100	Weibull	1.0	1.2	-0.5	0.64	0.50	0.56
200	Weibull	0.8	1.2	-0.5	0.84	0.64	0.82
100	Weibull	0.8	1.2	-0.5	0.54	0.40	0.44
200	Weibull	0.5	1.2	-0.5	0.88	0.78	0.86
100	Weibull	0.5	1.2	-0.5	0.72	0.58	0.64

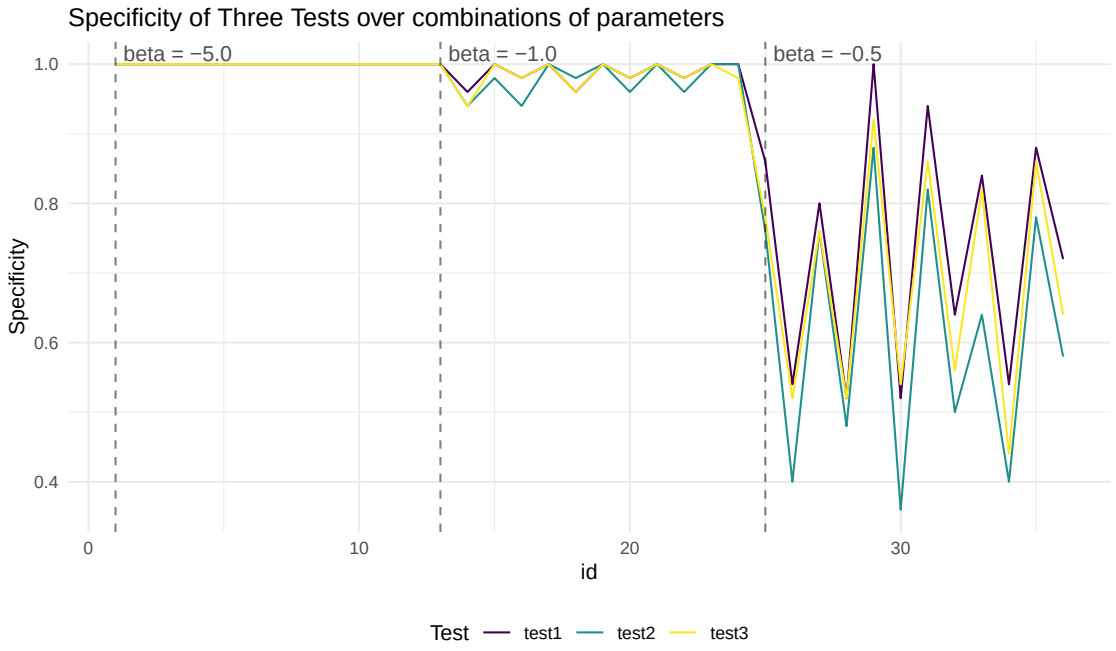
```
ph_exp_iddf = ph_exp_df |>
  mutate(id = seq_len(nrow(ph_exp_df ))) |>
  select(id, everything()) |>
  pivot_longer(cols = starts_with("test"), names_to = "Test", values_to = "Specificity") |>
  mutate(Test = str_remove(Test, "_specificity"))

ph_exp_power_graph = ggplot(ph_exp_iddf, aes(x = id, y = Specificity, color = Test)) +
  geom_line() +
  geom_vline(xintercept = 1, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 1.3, y = 1, label = "beta = -5.0", vjust = -0.2, hjust = 0, alpha = 0.7) +
```

```
  geom_vline(xintercept = 7, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 7.3, y = 1, label = "beta = -1.0", vjust = -0.2, hjust = 0, alpha = 0.7) +
  geom_vline(xintercept = 13, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 13.3, y = 1, label = "beta = -0.5", vjust = -0.2, hjust = 0, alpha = 0.7) +
  labs(
    x = "id",
    y = "Specificity",
    title = "Specificity of Three Tests over combinations of parameters",
    color = "Test"
  )

ph_exp_power_graph
```




```
ph_wei_iddf = ph_wei_df |>
  mutate(id = seq_len(nrow(ph_wei_df ))) |>
  select(id, everything()) |>
  pivot_longer(cols = starts_with("test"), names_to = "Test", values_to = "Specificity") |>
  mutate(Test = str_remove(Test, "_specificity"))

ph_wei_spe_graph = ggplot(ph_wei_iddf, aes(x = id, y = Specificity, color = Test)) +
  geom_line() +
  geom_vline(xintercept = 1, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 1.3, y = 1, label = "beta = -5.0", vjust = -0.2, hjust = 0, alpha = 0.7) +
  geom_vline(xintercept = 13, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 13.3, y = 1, label = "beta = -1.0", vjust = -0.2, hjust = 0, alpha = 0.7) +
  geom_vline(xintercept = 25, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 25.3, y = 1, label = "beta = -0.5", vjust = -0.2, hjust = 0, alpha = 0.7) +
  labs(
    x = "id",
    y = "Specificity",
    title = "Specificity of Three Tests over combinations of parameters",
```

```
    color = "Test"
  )

ph_wei_spe_graph
```



Non-Proportional-Hazard Assumption

Under Non-Proportional-Hazard 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,

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

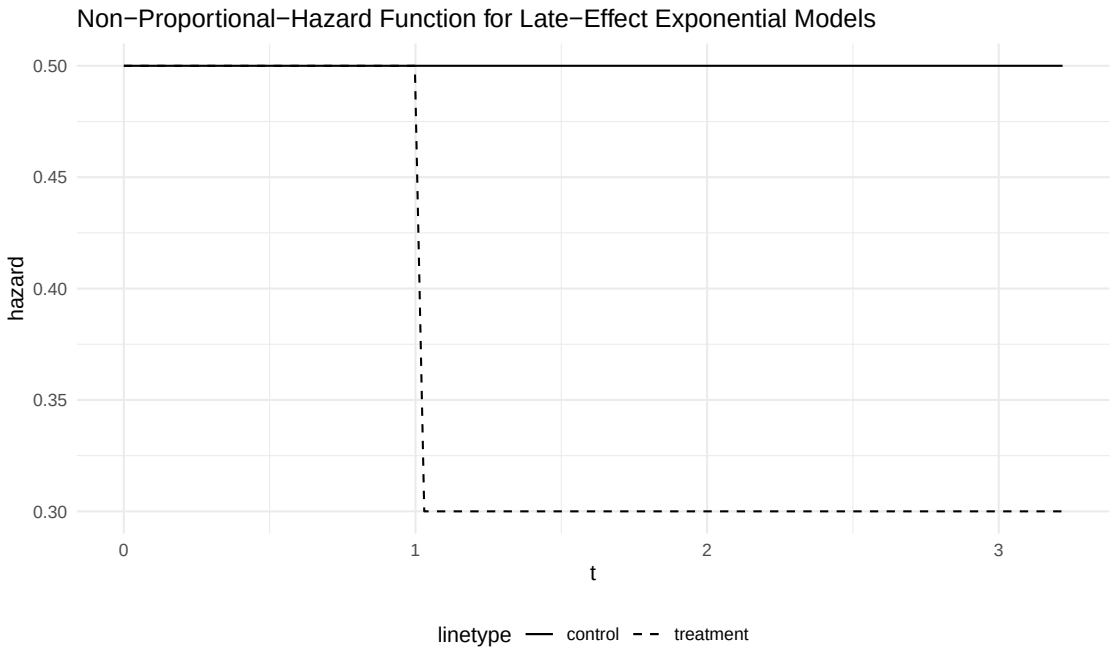
$$S(t|x=1) = exp(-H(t|x=1)) = \begin{cases} exp(-\lambda_0 t) & t < 1 \\ exp(-(\lambda_0 + \lambda_1)t + \lambda_1) & t \geq 1 \end{cases}$$

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

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.)

```
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))
}
```

```
late_pw_param_df = expand.grid(iteration = c(1:50), n = c(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.

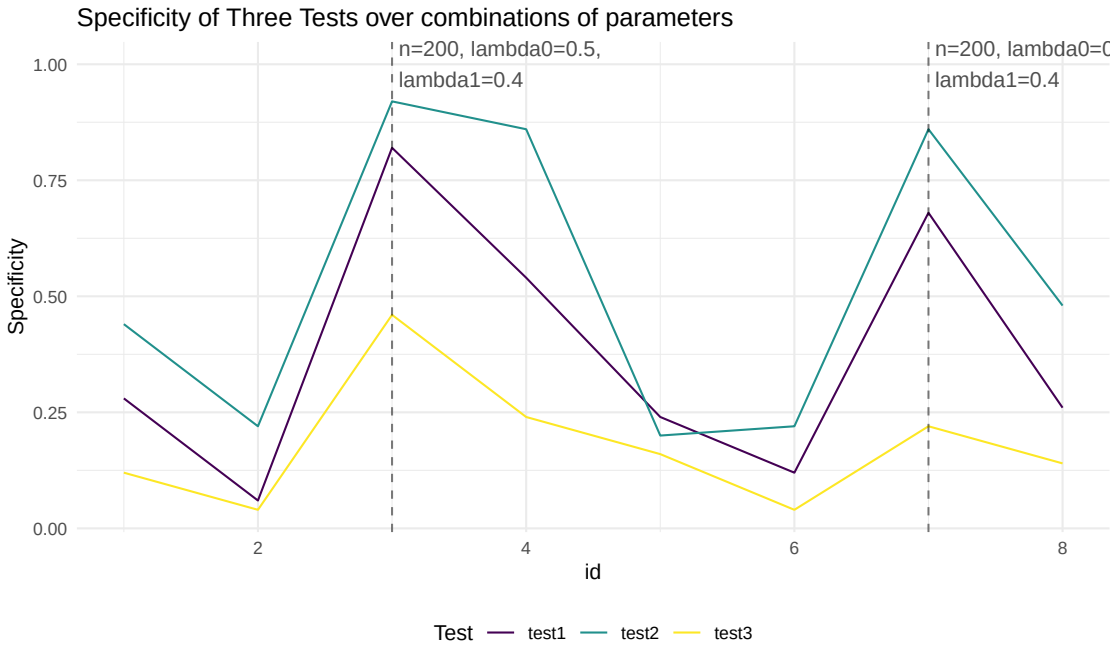
```
late_pw_df = late_pw_df |>
  mutate(specificity = map(simulations, specif_func)) |>
  unnest(specificity) |>
  select(-simulations)
```

```
kable(late_pw_df, caption = "Specificity of 3 Log-Rank Tests based on NPH Assumption (Late)")
```

Table 3: Specificity of 3 Log-Rank Tests based on NPH Assumption (Late)

n	lambda0	lambda1	test1_specificity	test2_specificity	test3_specificity
200	0.8	0.4	0.28	0.44	0.12
100	0.8	0.4	0.06	0.22	0.04
200	0.5	0.4	0.82	0.92	0.46
100	0.5	0.4	0.54	0.86	0.24
200	0.8	0.3	0.24	0.20	0.16
100	0.8	0.3	0.12	0.22	0.04
200	0.5	0.3	0.68	0.86	0.22
100	0.5	0.3	0.26	0.48	0.14

```
late_pw_iddf = late_pw_df |>
  mutate(id = seq_len(nrow(late_pw_df))) |>
  select(id, everything()) |>
  pivot_longer(cols = starts_with("test"), names_to = "Test", values_to = "Specificity") |>
  mutate(Test = str_remove(Test, "_specificity"))

late_pw_spe_graph = ggplot(late_pw_iddf, aes(x = id, y = Specificity, color = Test)) +
  geom_line() +
  geom_vline(xintercept = 3, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 3.05, y = 1, label = "n=200, lambda0=0.5,\nlambda1=0.4", vjust = 0.5, hjust = 0,
  geom_vline(xintercept = 7, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 7.05, y = 1, label = "n=200, lambda0=0.5,\nlambda1=0.4", vjust = 0.5, hjust = 0,
  labs(
    x = "id",
    y = "Specificity",
    title = "Specificity of Three Tests over combinations of parameters",
    color = "Test"
  )

late_pw_spe_graph
```



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:

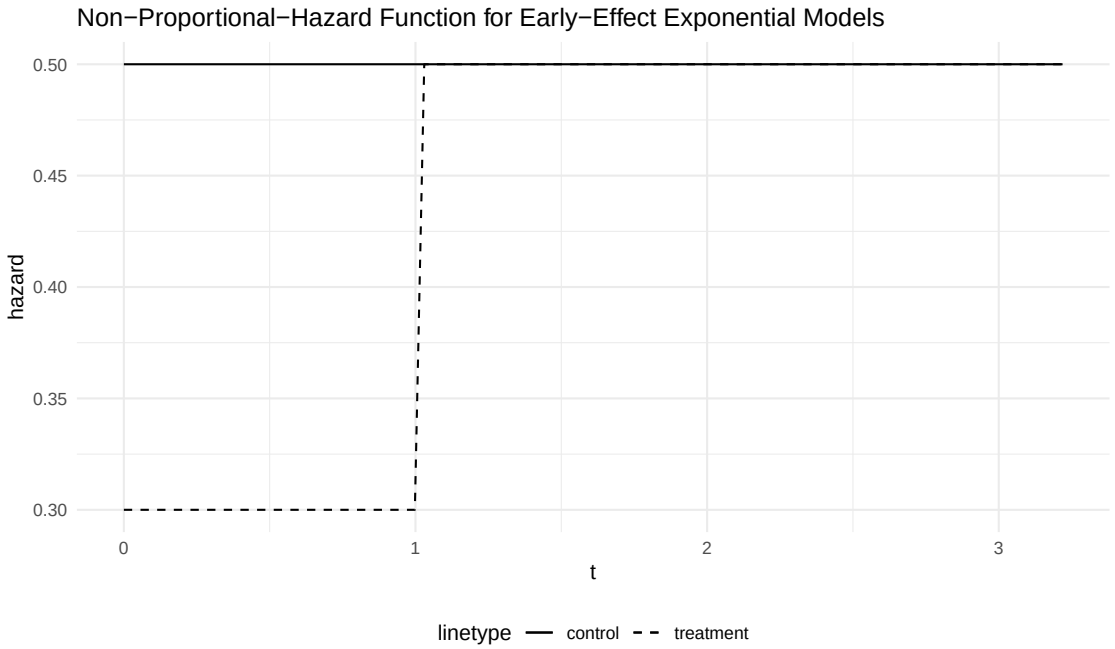
$$h(t|x=1) = \begin{cases} \lambda_0 & t \geq 1 \\ \lambda_1 & t < 1 \end{cases}$$

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:




```
early_pw_param_df = expand.grid(iteration = c(1:50), n = c(100, 200),
            lambda0 = c(0.8, 0.9), lambda1 = c(0.6, 0.7))

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.

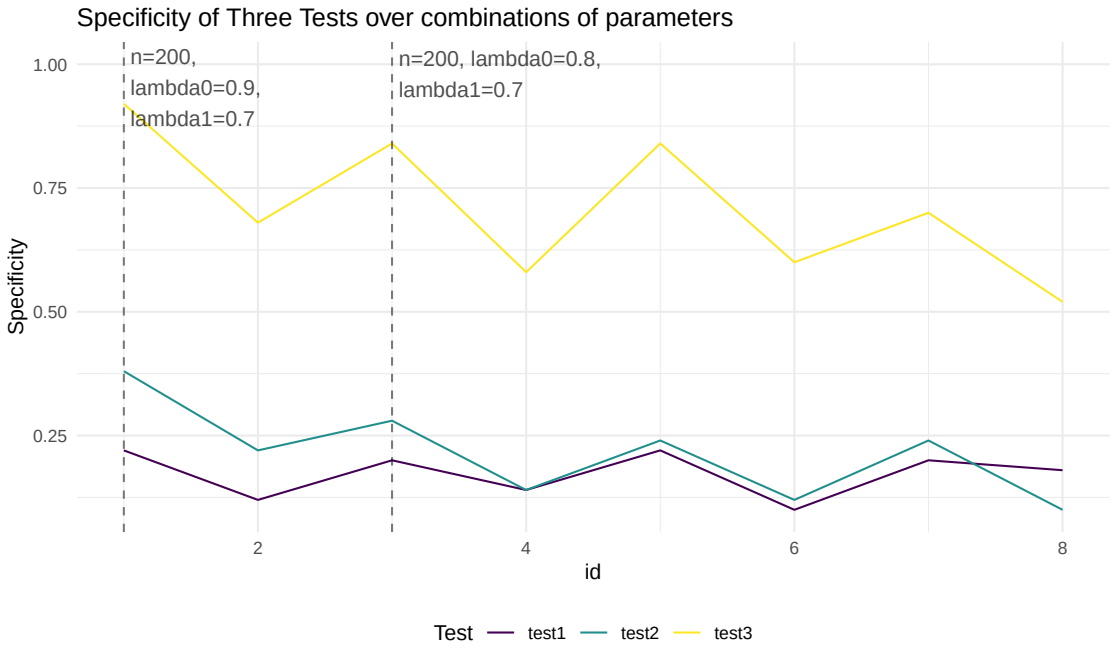
```
early_pw_df = early_pw_df |>
  mutate(specificity = map(simulations, specif_func)) |>
  unnest(specificity) |>
  select(-simulations)
```

```
kable(early_pw_df, caption = "Specificity of 3 Log-Rank Tests based on NPH Assumption (Early)")
```

Table 4: Specificity of 3 Log-Rank Tests based on NPH Assumption (Early)

n	lambda0	lambda1	test1_specificity	test2_specificity	test3_specificity
200	0.9	0.7	0.22	0.38	0.92
100	0.9	0.7	0.12	0.22	0.68
200	0.8	0.7	0.20	0.28	0.84
100	0.8	0.7	0.14	0.14	0.58
200	0.9	0.6	0.22	0.24	0.84
100	0.9	0.6	0.10	0.12	0.60
200	0.8	0.6	0.20	0.24	0.70
100	0.8	0.6	0.18	0.10	0.52

```
early_pw_iddf = early_pw_df |>
  mutate(id = seq_len(nrow(early_pw_df))) |>
  select(id, everything()) |>
  pivot_longer(cols = starts_with("test"), names_to = "Test", values_to = "Specificity") |>
  mutate(Test = str_remove(Test, "_specificity"))

early_pw_spe_graph = ggplot(early_pw_iddf, aes(x = id, y = Specificity, color = Test)) +
  geom_line() +
  geom_vline(xintercept = 3, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 3.05, y = 1, label = "n=200, lambda0=0.8,\nlambda1=0.7", vjust = 0.7, hjust = 0,
  geom_vline(xintercept = 1, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 1.05, y = 1, label = "n=200,\nlambda0=0.9,\nlambda1=0.7", vjust = 0.8, hjust = 0
  labs(
    x = "id",
    y = "Specificity",
    title = "Specificity of Three Tests over combinations of parameters",
    color = "Test"
  )

early_pw_spe_graph
```



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:

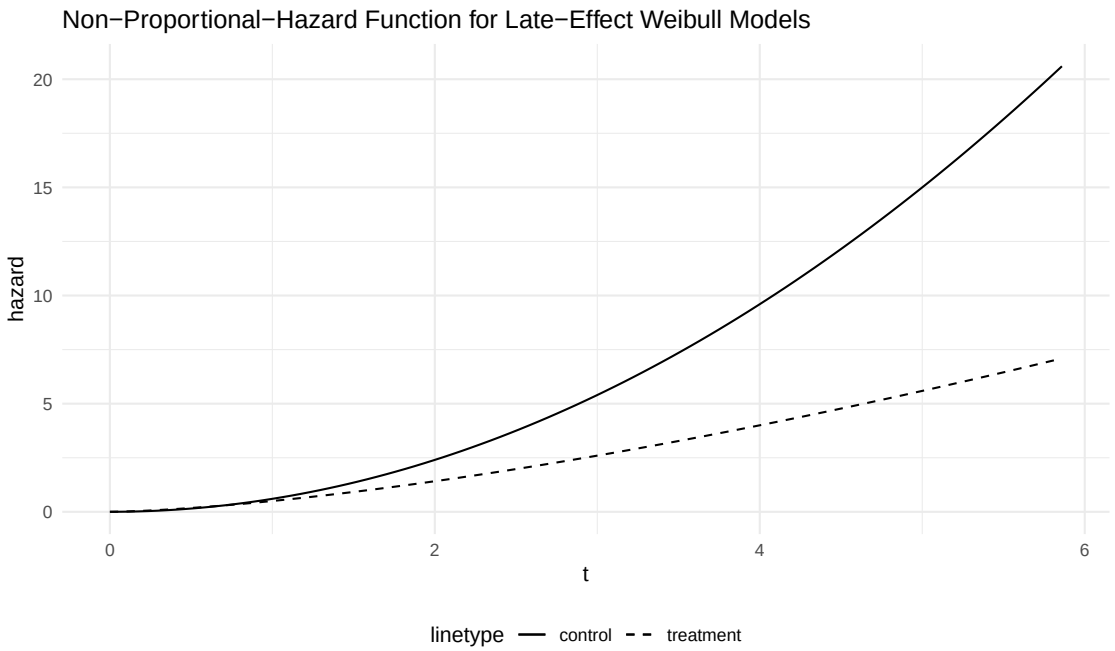
$$h(t|x=1) = \lambda\gamma_1 t^{(\gamma_1-1)}.$$

We can derive that

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

Late Effect

The hazard function under $\lambda = 0.2$, $\gamma_0 = 3$, $\gamma_1 = 2.5$ 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 = 1 / 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)

  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))
}
```

```
late_wei_param_df =
  expand.grid(iteration = c(1:50), n = c(100, 200),
              lambda = c(0.2, 0.4), gamma0 = c(4, 5),
              gamma1 = c(2.5, 3))

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(specificity = map(simulations, specif_func)) |>
  unnest(specificity) |>
  select(-simulations)
```

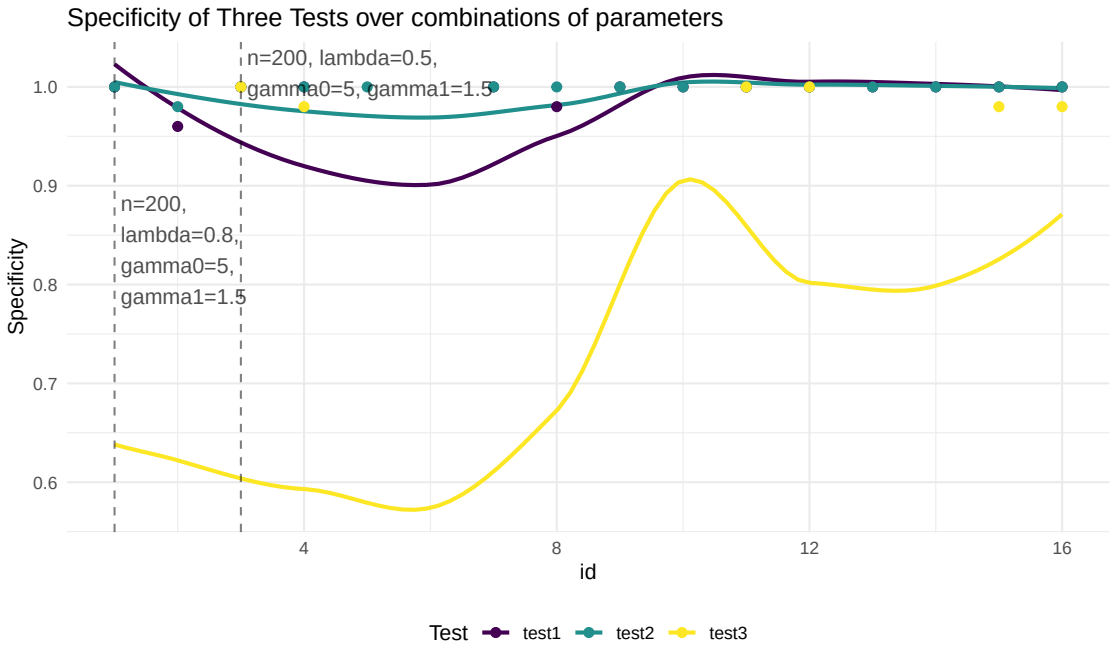
```
kable(late_wei_df, caption = "Specificity of 3 Log-Rank Tests based on NPH Assumption")
```

Table 5: Specificity of 3 Log-Rank Tests based on NPH Assumption

n	lambda	gamma0	gamma1	test1_specificity	test2_specificity	test3_specificity
200	0.4	5	3.0	1.00	1.00	0.64
100	0.4	5	3.0	0.96	0.98	0.30
200	0.2	5	3.0	1.00	1.00	1.00
100	0.2	5	3.0	1.00	1.00	0.98
200	0.4	4	3.0	0.88	1.00	0.26
100	0.4	4	3.0	0.74	0.88	0.26
200	0.2	4	3.0	1.00	1.00	0.86
100	0.2	4	3.0	0.98	1.00	0.64
200	0.4	5	2.5	1.00	1.00	0.86
100	0.4	5	2.5	1.00	1.00	0.78
200	0.2	5	2.5	1.00	1.00	1.00
100	0.2	5	2.5	1.00	1.00	1.00
200	0.4	4	2.5	1.00	1.00	0.64
100	0.4	4	2.5	1.00	1.00	0.32
200	0.2	4	2.5	1.00	1.00	0.98
100	0.2	4	2.5	1.00	1.00	0.98

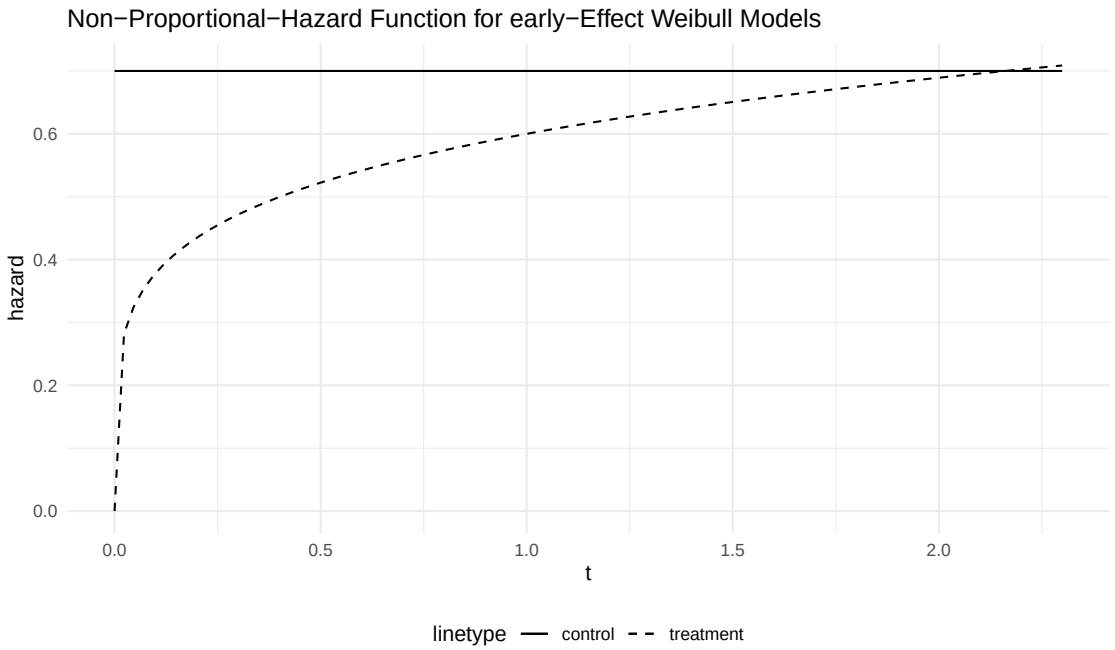
```
late_wei_iddf = late_wei_df |>
  mutate(id = seq_len(nrow(late_wei_df))) |>
  select(id, everything()) |>
  pivot_longer(cols = starts_with("test"), names_to = "Test", values_to = "Specificity") |>
  mutate(Test = str_remove(Test, "_specificity"))

late_wei_spe_graph = ggplot(late_wei_iddf, aes(x = id, y = Specificity, color = Test)) +
  #geom_line() +
  geom_smooth(se=FALSE) +
  labs(
    x = "id",
    y = "Specificity",
    title = "Specificity of Three Tests over combinations of parameters",
    color = "Test"
  )
late_wei_spe_graph = late_wei_spe_graph +
  geom_point(data = subset(late_wei_iddf, Specificity > 0.9), aes(color = Test), size = 2)

late_wei_spe_graph = late_wei_spe_graph +
  geom_point(data = subset(late_wei_iddf, Specificity < 0.2), aes(color = Test), size = 1.5)

late_wei_spe_graph = late_wei_spe_graph +
  geom_vline(xintercept = 3, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 3.1, y = 1, label = "n=200, lambda=0.5,\ngamma0=5, gamma1=1.5", vjust = 0.2, hjus
  geom_vline(xintercept = 1, linetype = "dashed", color = "black", alpha = 0.5) +
  annotate("text", x = 1.1, y = 1, label = "n=200,\nlambda=0.8,\ngamma0=5,\ngamma1=1.5", vjust = 2, hjus

late_wei_spe_graph
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



Early Effect

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.

```
weiexp_sim_func = function(n, lambda0, lambda1 = 0.5, gamma)
{
  # 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)
  t1 = (- log(u1) / lambda1) ^ (1 / gamma)

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

```
early_wei_param_df =
  expand.grid(iteration = c(1:50), n = c(100, 200),
              lambda0 = c(0.8, 1), lambda1 = 0.5, gamma = c(1.1, 1.2))

early_wei_results =
  mapply(weiexp_sim_func, n = early_wei_param_df$n,
         lambda0 = early_wei_param_df$lambda0,
         lambda1 = early_wei_param_df$lambda1, gamma = early_wei_param_df$gamma)

early_wei_df = tibble()

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

early_wei_df = early_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
early_wei_df = early_wei_df |>
  mutate(specificity = map(simulations, specif_func)) |>
  unnest(specificity) |>
  select(-simulations)
```

```r
kable(early_wei_df, caption = "Specificity of 3 Log-Rank Tests based on NPH Assumption")
```

Table 6: Specificity of 3 Log-Rank Tests based on NPH Assumption

n	lambda0	lambda1	gamma	test1_specificity	test2_specificity	test3_specificity
200	1.0	0.5	1.2	1.00	0.98	1.00
100	1.0	0.5	1.2	0.98	0.82	1.00
200	0.8	0.5	1.2	0.86	0.54	0.94
100	0.8	0.5	1.2	0.66	0.28	0.76
200	1.0	0.5	1.1	1.00	1.00	1.00
100	1.0	0.5	1.1	0.98	0.86	0.94
200	0.8	0.5	1.1	0.96	0.86	0.96
100	0.8	0.5	1.1	0.74	0.66	0.70

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