Monday, April 13

Probability Density, Survival, and Hazard Functions

Let T be a continuous random variable that is time-till-event. Four related functions are used to describe the distribution of T.

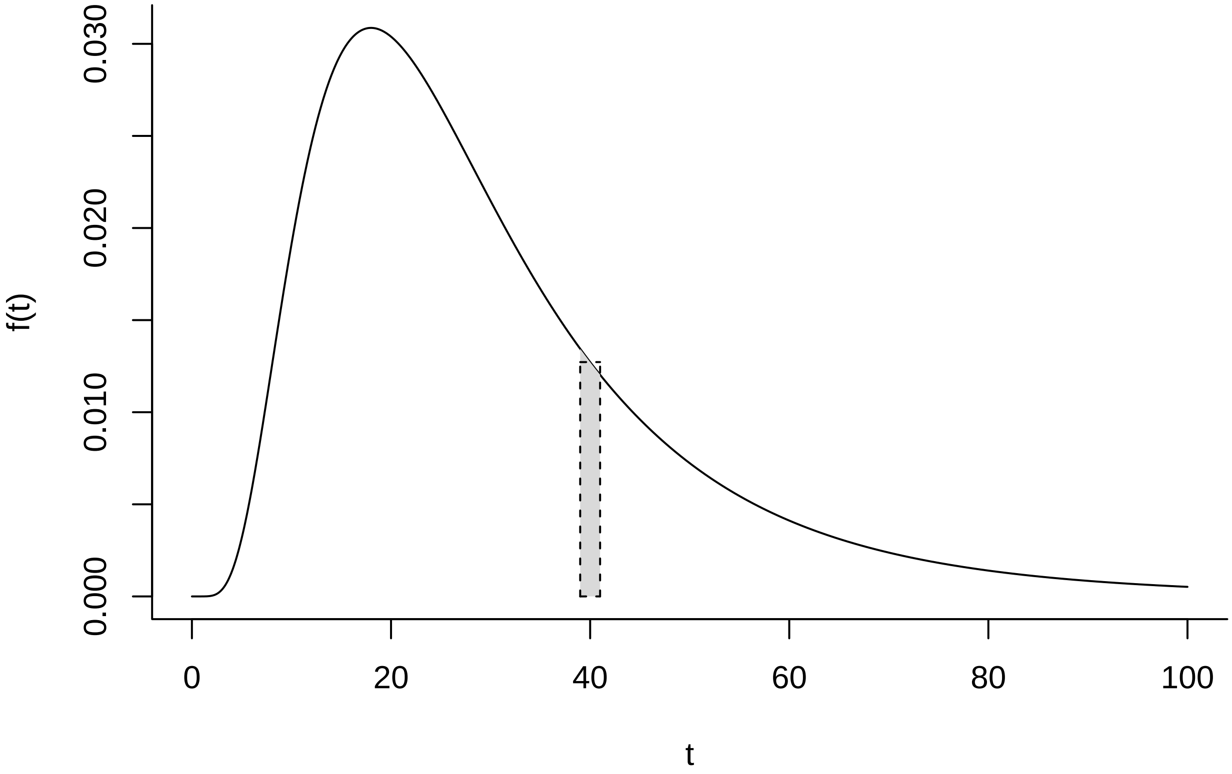
The Probability Density Function

The *probability density function* of T is

$$f(t) = \lim_{\delta t \to 0} \frac{P(t \leq T < t + \delta t)}{\delta t}.$$

If δ is relatively small then $P(t \leq T < t + \delta t) \approx f(t)\delta t$ and so $f(t) \approx P(t \leq T < t + \delta t)/(\delta t)$ and thus $f(t)$ is approximately *proportional* to the probability that T is between t and $t + \delta t$. *So $f(t)$ is approximately proportional to the probability that the event will happen "near" t.*

For the distribution below, the probability that T is *approximately* 40 (say, between 39 and 41) equals the area under the curve and between 39 and 41. This probability is *approximated* by the rectangle, which has area $wf(40)$, where $w = 2$ is the width of the rectangle and $f(40)$ is the height of the rectangle. So the probability that T is approximately 40 is *proportional to* $f(40)$.

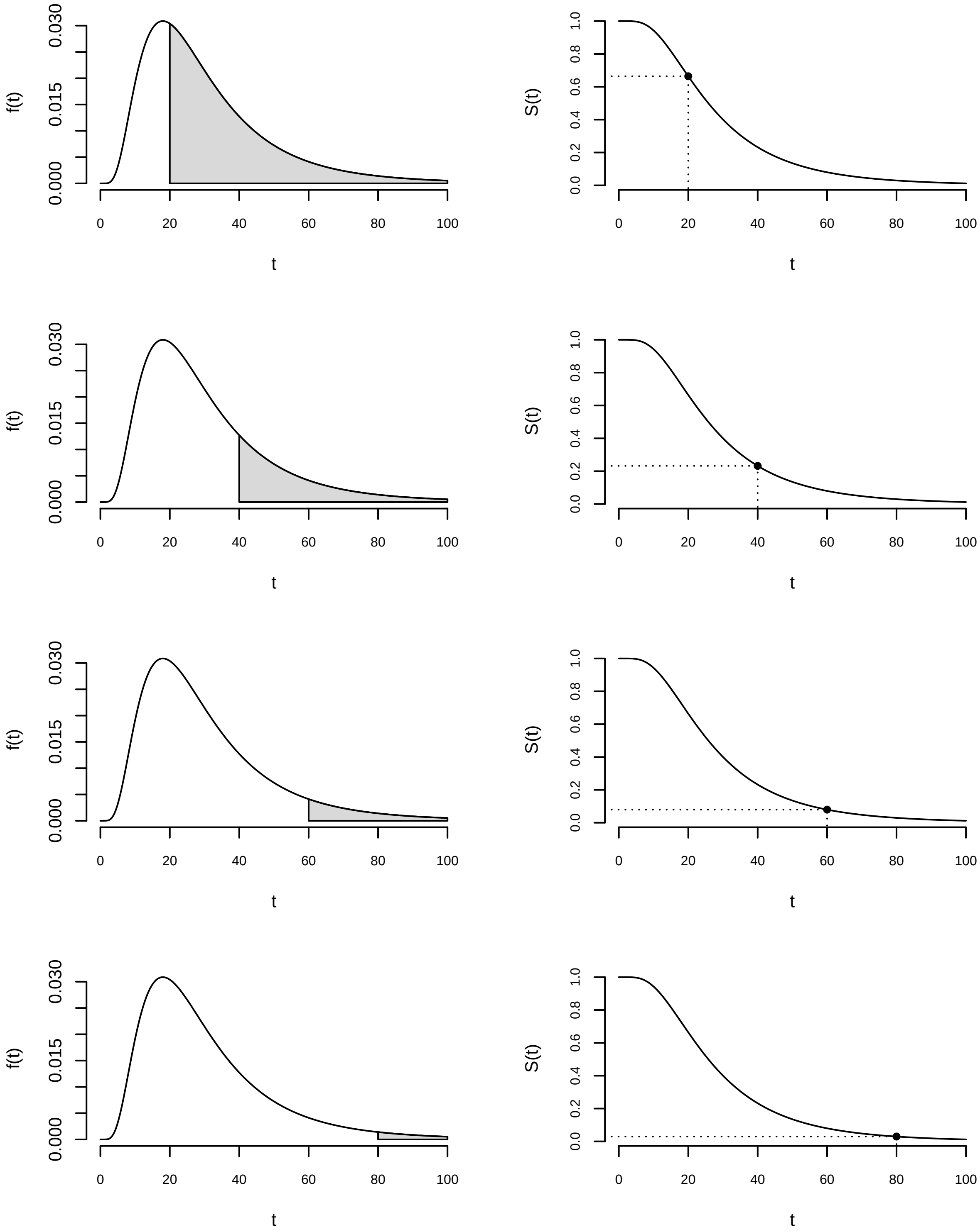


The Survival Function

The *survival function* is

$$S(t) = P(T \geq t).$$

It equals the area under $f(t)$ and between t and ∞. The area under $S(t)$ equals $E(T)$ if $S(0) = 1$ and $S(\infty) = 1$.



Thus

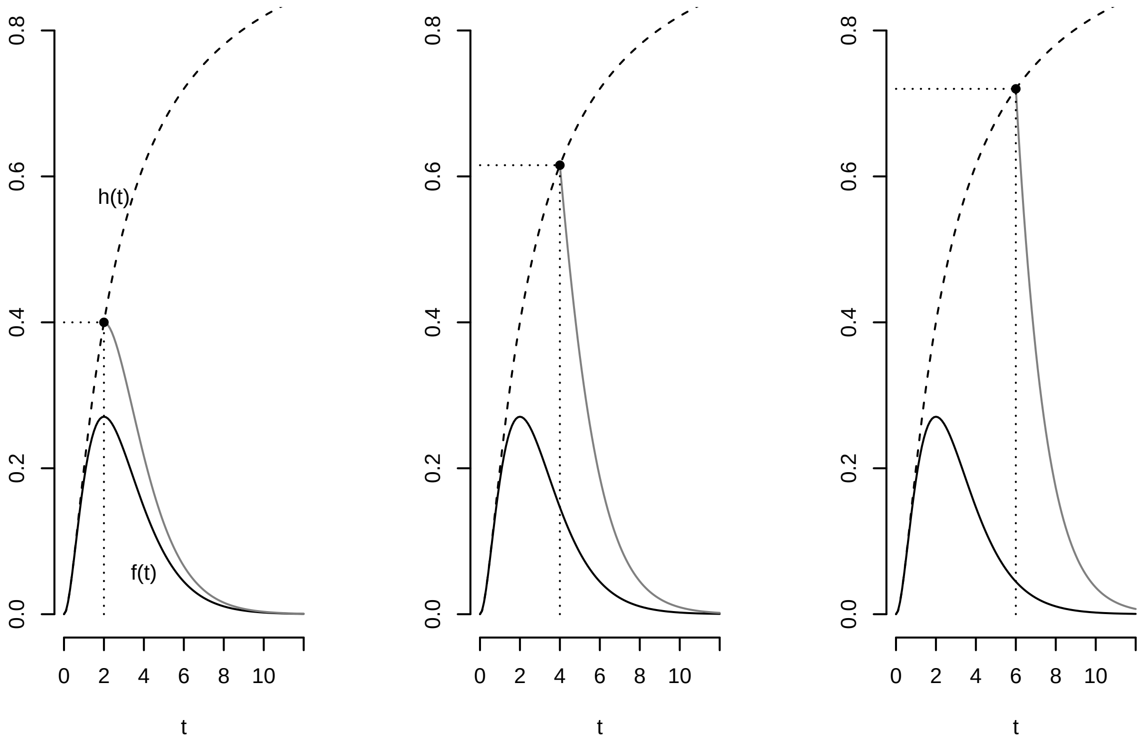
$$S(t) = \int_t^\infty f(z)dz.$$

The Hazard Function

The *hazard function* is

$$h(t) = \lim_{\delta t \to 0} \frac{P(t \le T < t + \delta t | T \ge t)}{\delta t} = \frac{f(t)}{S(t)}.$$

If δ is relatively small then $h(t)$ is approximately *proportional* to the probability that $t \le T < t + \delta t$ *given* survival up to t — i.e., $T \ge t$. *So* $h(t)$ *is approximately proportional to the probability of the event happening at near time t if it has not yet happened.*



Distributions and Hazard Functions

A wide variety of distributions can be used for parametric survival models such as AFT models. Below is a list of just some of those distributions. One of the more noticeable differences between them is the shape of their hazard functions.

1. *Log-normal.* The distribution of $\log(T_i)$ is normal. Single-peaked hazard function. Known as `lognormal` by `survreg` and `flexsurvreg`, and also `lnorm` by `flexsurvreg`.
2. *Log-logistic.* The distribution of $\log(T_i)$ is logistic. Single-peaked or decreasing hazard function. Known as `loglogistic` by `survreg` and `flexsurvreg` and `llogis` by `flexsurvreg`.
3. *Gamma.* Monotonic or flat hazard function. Known as `gamma` to `flexsurvreg`.
4. *Weibull.* Monotonic or flat hazard function. Known as `weibull` to both `survreg` and `flexsurvreg`.
5. *Exponential.* Flat hazard function ("memoryless"). Known as `exp` to `flexsurvreg` but also as a special case of `weibull` if `scale = 1` with `survreg`.
6. *Gompertz.* Increasing hazard function. Known as `gompertz` to `flexsurvreg`.

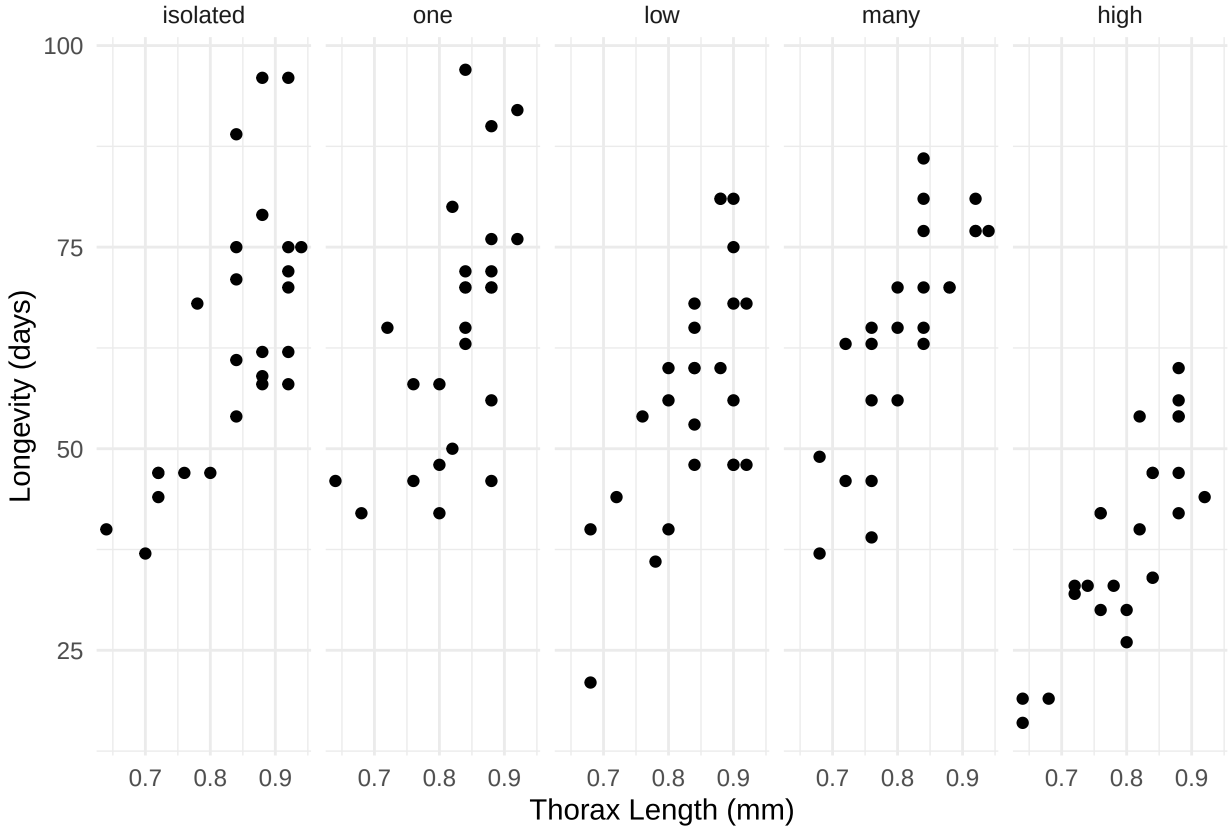
7. *Generalized gamma*. Monotonic, single-peaked, and "bathtub" hazard functions. The exponential, Weibull, gamma, and log-normal are special cases. Known as `gengamma` to `flexsurvreg`.
8. *Generalized F*. Single-peaked or decreasing. Known as `genf` to `flexsurvreg`.

Estimating and Plotting Hazard Functions

The `summary` function can be used to estimate the hazard function based on a `flexsurvreg` model object.

Example: Consider data from an experiment on the effects of sexual activity on the lifespan of the male fruitfly. Thorax length was used as a covariate.

```
library(faraway)
p <- ggplot(fruitfly, aes(x = thorax, y = longevity)) +
  geom_point() + facet_wrap(~ activity, ncol = 5) +
  labs(x = "Thorax Length (mm)", y = "Longevity (days)") +
  theme_minimal()
plot(p)
```

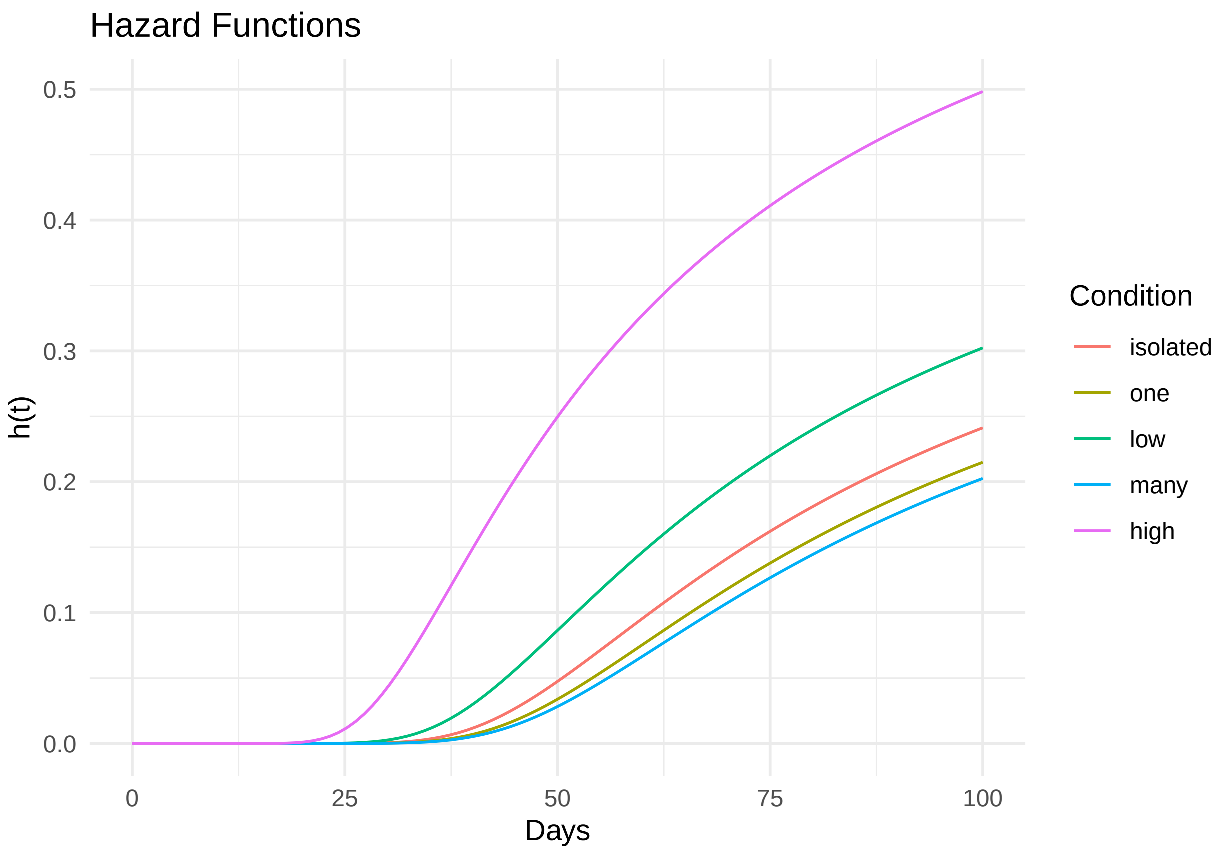


```
m <- flexsurvreg(Surv(longevity) ~ activity + thorax,
  data = fruitfly, dist = "gamma")

d <- data.frame(activity = unique(fruitfly$activity), thorax = 0.8)
d <- summary(m, newdata = d, t = seq(0, 100, length = 100),
 type = "hazard", tidy = TRUE)
head(d)
```

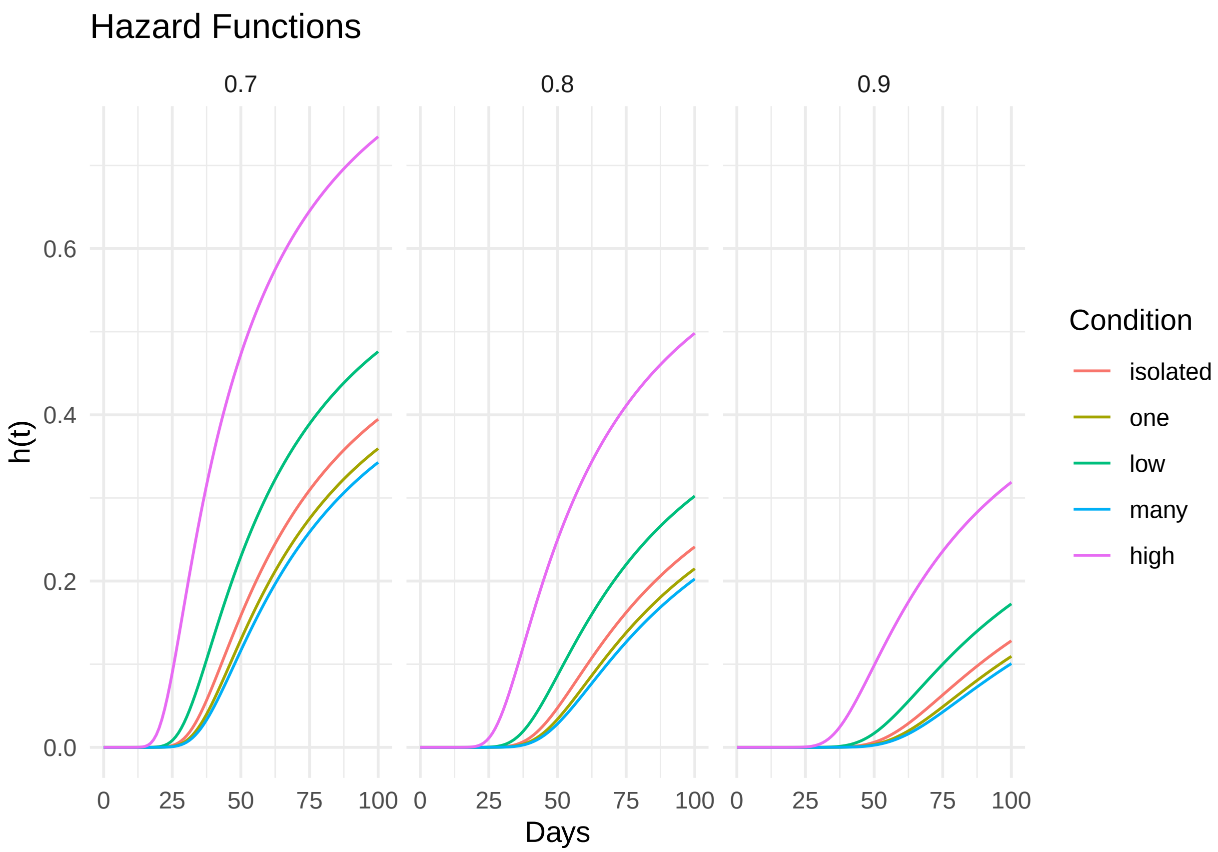
```
    time        est        lcl        ucl activity thorax
```

```
1 0.000 0.000e+00 0.000e+00 0.000e+00     many    0.8
2 1.010 1.218e-39 3.973e-51 1.140e-30     many    0.8
3 2.020 1.900e-31 2.059e-40 1.869e-24     many    0.8
4 3.030 9.683e-27 2.748e-34 6.895e-21     many    0.8
5 4.040 1.845e-23 4.539e-30 2.099e-18     many    0.8
6 5.051 5.806e-21 8.271e-27 1.635e-16     many    0.8
```

```r
p <- ggplot(d, aes(x = time, y = est, color = activity)) +
  geom_line() + theme_minimal() +
  labs(x = "Days", y = "h(t)", color = "Condition", title = "Hazard Functions")
plot(p)
```

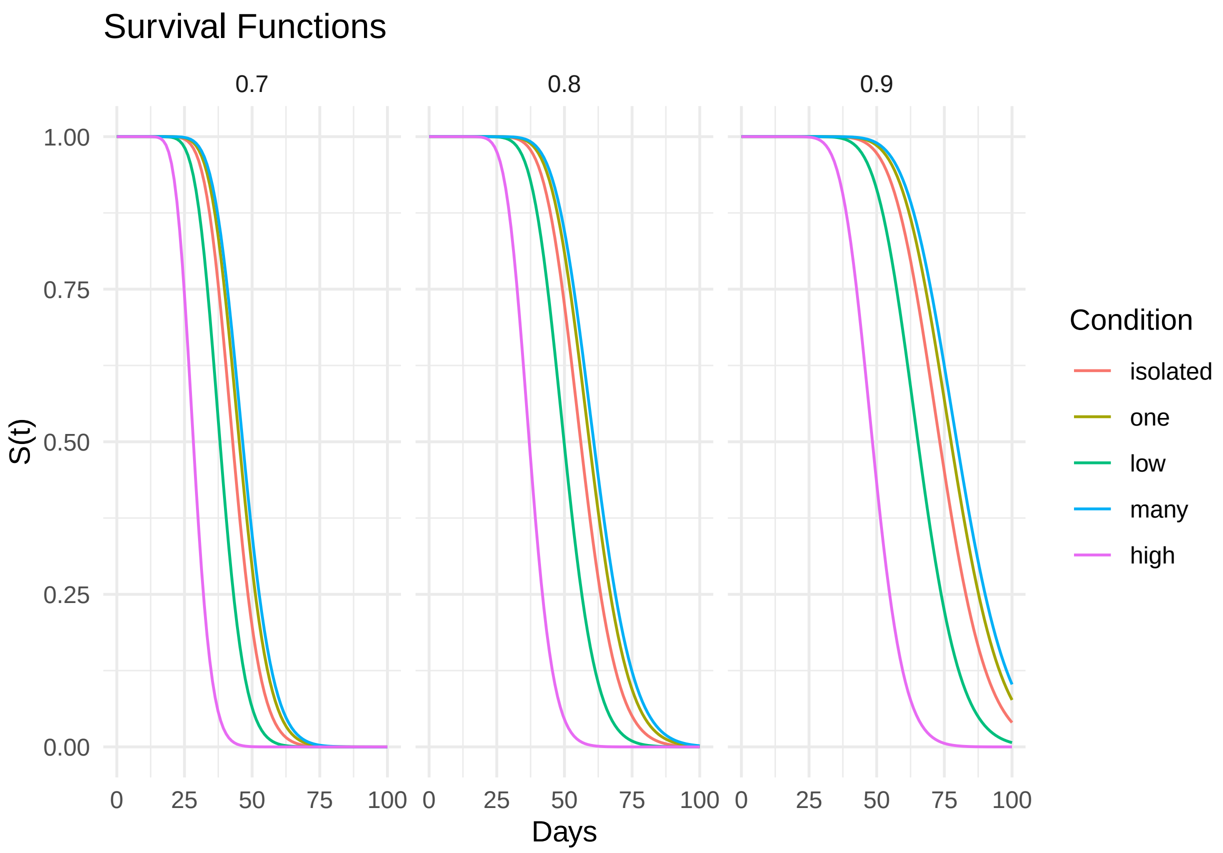


```r
d <- expand.grid(activity = unique(fruitfly$activity), thorax = c(0.7,0.8,0.9))
d <- summary(m, newdata = d, t = seq(0, 100, length = 100),
  type = "hazard", tidy = TRUE)

p <- ggplot(d, aes(x = time, y = est, color = activity)) +
  geom_line() + theme_minimal() +
  labs(x = "Days", y = "h(t)", color = "Condition", title = "Hazard Functions") +
  facet_wrap(~ thorax)
plot(p)
```



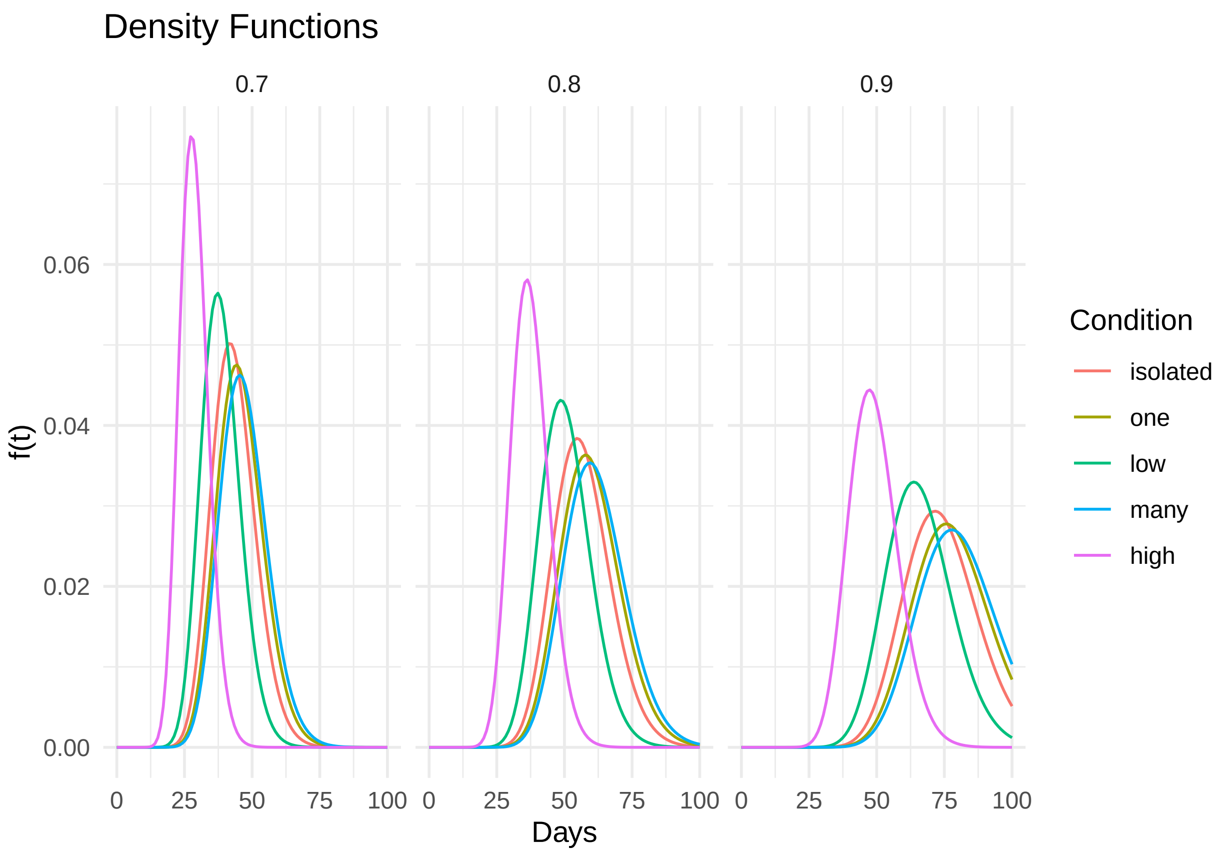
For comparison here are the *survival functions*.

```
d <- expand.grid(activity = unique(fruitfly$activity), thorax = c(0.7,0.8,0.9))
d <- summary(m, newdata = d, t = seq(0, 100, length = 100),
  type = "survival", tidy = TRUE)

p <- ggplot(d, aes(x = time, y = est, color = activity)) +
  geom_line() + theme_minimal() +
  labs(x = "Days", y = "S(t)", color = "Condition", title = "Survival Functions") +
  facet_wrap(~ thorax)
plot(p)
```



And here are the *probability density functions*.

```
d <- expand.grid(activity = unique(fruitfly$activity), thorax = c(0.7,0.8,0.9))
d <- summary(m, newdata = d, t = seq(0, 100, length = 100),
  fn = function(t, ...) dgamma(t, ...), tidy = TRUE)

p <- ggplot(d, aes(x = time, y = est, color = activity)) +
  geom_line() + theme_minimal() +
  labs(x = "Days", y = "f(t)", color = "Condition", title = "Density Functions") +
  facet_wrap(~ thorax)
plot(p)
```



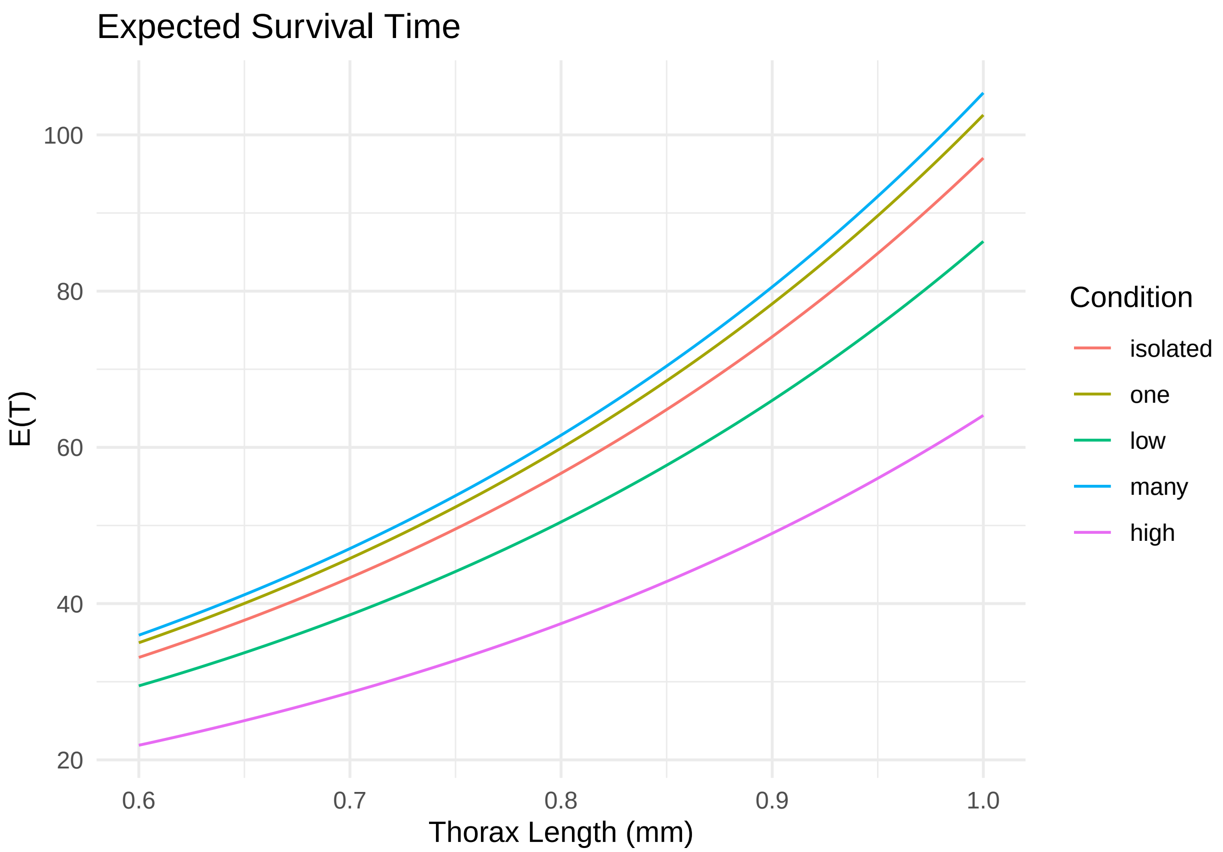
Note that we can adapt this to other distributions by adding a `d` to the beginning of the distribution name recognized by `flexsurvreg`. This include log-normal (`dlnorm`), log-logistic (`dllogis`), gamma (`dgamma`), Weibull (`dweibull`), exponential (`dexp`), Gompertz (`dgompertz`), generalized gamma (`dgengamma`), and generalized F (`dgenf`).

Finally we can also plot the *expected* survival time. This is analogous to using `predict` with `type = response` in a GLM.

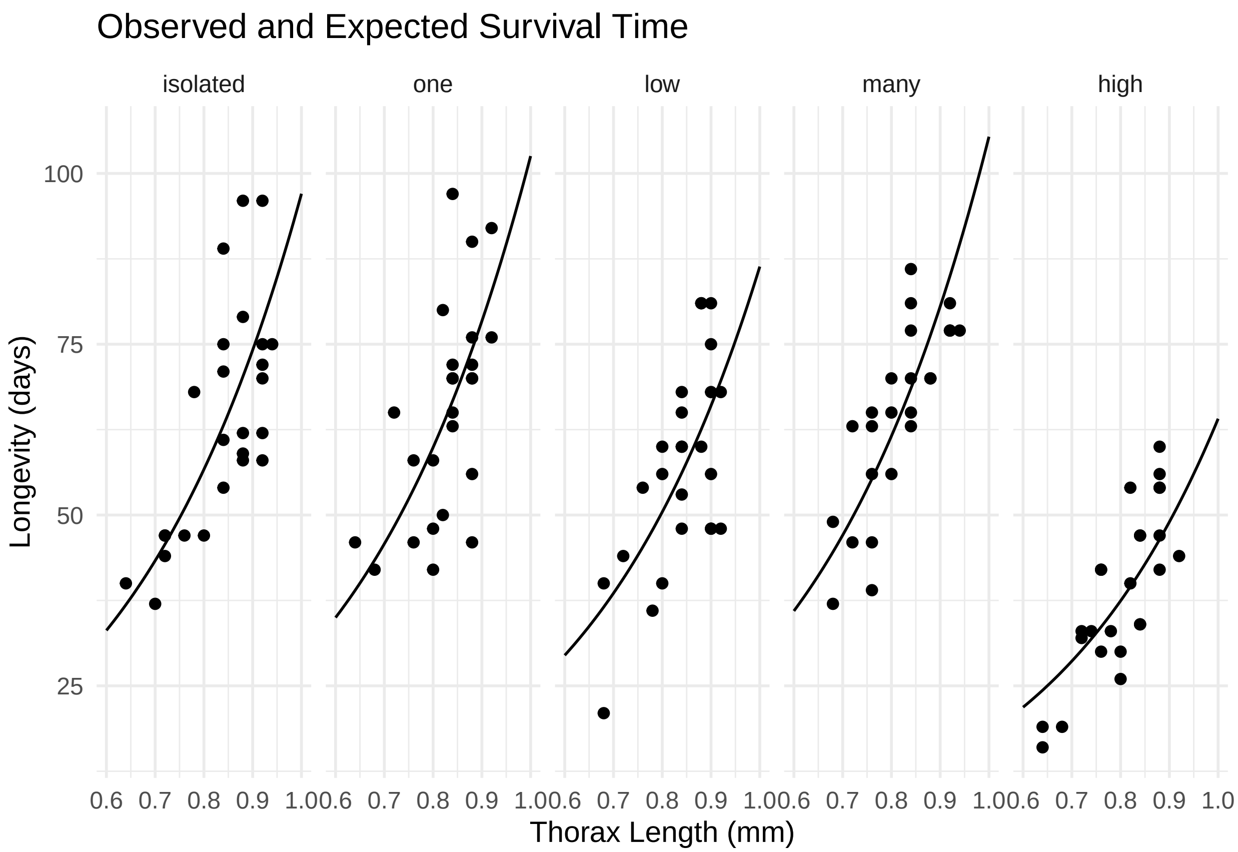
```
d <- expand.grid(activity = unique(fruitfly$activity),
  thorax = seq(0.6, 1.0, length = 100))
d <- summary(m, newdata = d, type = "mean", tidy = TRUE)
head(d)
```

```
    est   lcl   ucl activity thorax
1 35.96 31.96 40.88     many  0.600
2 33.11 29.19 37.70 isolated  0.600
3 34.99 30.59 39.78      one  0.600
4 29.47 25.88 33.57      low  0.600
5 21.87 19.30 24.62     high  0.600
6 36.35 32.34 41.26     many  0.604
```

```
p <- ggplot(d, aes(x = thorax, y = est, color = activity)) +
  geom_line() + theme_minimal() +
  labs(x = "Thorax Length (mm)", y = "E(T)", color = "Condition",
    title = "Expected Survival Time")
plot(p)
```



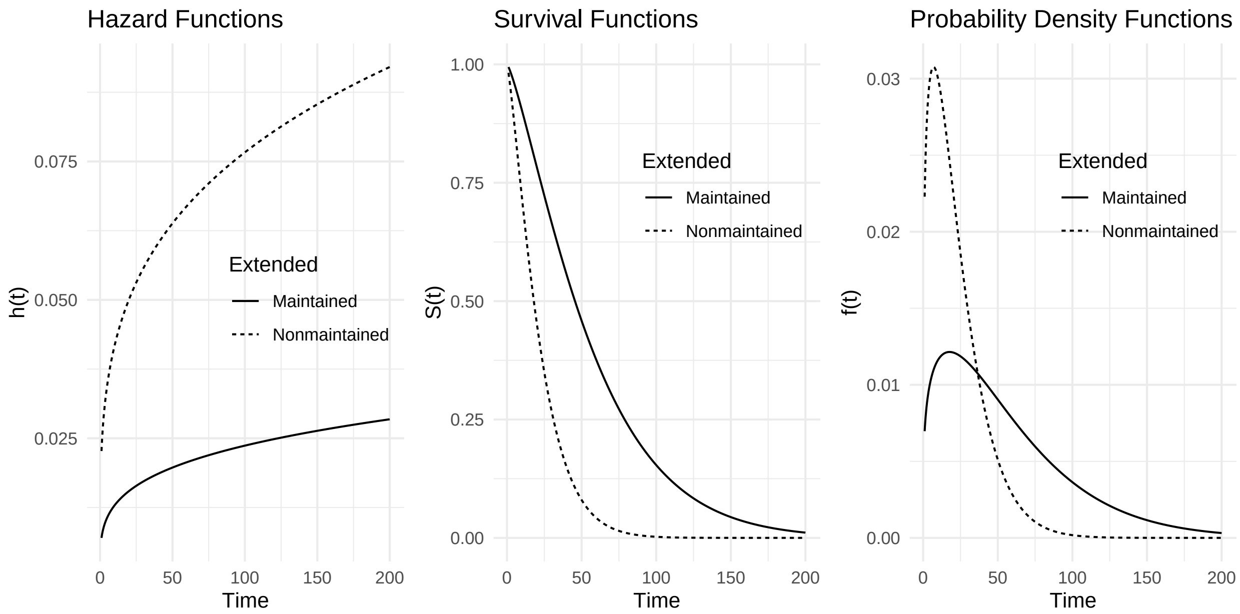
```
p <- ggplot(fruitfly, aes(x = thorax, y = longevity)) +
  geom_point() + facet_wrap(~ activity, ncol = 5) +
  labs(x = "Thorax Length (mm)", y = "Longevity (days)",
   title = "Observed and Expected Survival Time") +
  theme_minimal() + geom_line(aes(y = est), data = d)
plot(p)
```



Example: Consider an AFT model for the `leukemia` data. Note that patients are either of remission (not censored) or still in remission (right-censored).

```
library(survival)
leukemia$status <- factor(leukemia$status, labels = c("in","out"))

m <- flexsurvreg(Surv(time, status == "out") ~ x, dist = "weibull", data = leukemia)

# create plot of hazard functions
d <- data.frame(x = c("Maintained","Nonmaintained"))
d <- summary(m, newdata = d, t = seq(1, 200, length = 1000),
  type = "hazard", tidy = TRUE)

p <- ggplot(d, aes(x = time, y = est)) +
  geom_line(aes(linetype = x)) + theme_minimal() +
  labs(x = "Time", y = "h(t)", linetype = "Extended",
    title = "Hazard Functions") +
  theme(legend.position = "inside", legend.position.inside = c(0.7, 0.5))
p.h <- p

# create plot of survival functions
d <- data.frame(x = c("Maintained","Nonmaintained"))
d <- summary(m, newdata = d, t = seq(1, 200, length = 1000),
  type = "survival", tidy = TRUE)

p <- ggplot(d, aes(x = time, y = est)) +
  geom_line(aes(linetype = x)) + theme_minimal() +
```

```
  labs(x = "Time", y = "S(t)", linetype = "Extended",
    title = "Survival Functions") +
  theme(legend.position = "inside", legend.position.inside = c(0.7, 0.7))
p.s <- p

# create plot of probability density functions
d <- data.frame(x = c("Maintained","Nonmaintained"))
d <- summary(m, newdata = d, t = seq(1, 200, length = 1000),
  fn = function(t, ...) dweibull(t, ...), tidy = TRUE)

p <- ggplot(d, aes(x = time, y = est)) +
  geom_line(aes(linetype = x)) + theme_minimal() +
  labs(x = "Time", y = "f(t)", linetype = "Extended",
    title = "Probability Density Functions") +
  theme(legend.position = "inside", legend.position.inside = c(0.7, 0.7))
p.d <- p

# put the plots together into one plot
cowplot::plot_grid(p.h, p.s, p.d, ncol = 3)
```



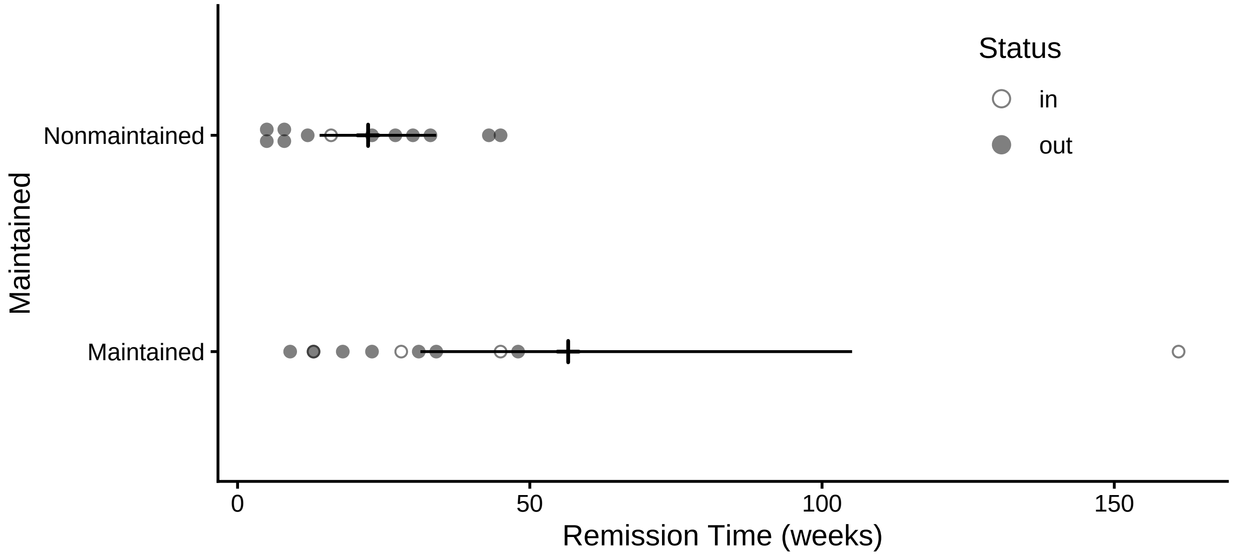
We can also plot the raw data with the estimated expected survival times and confidence intervals for the estimated expected survival time.

```
d <- summary(m, newdata = data.frame(x = c("Maintained","Nonmaintained")),
 type = "mean", tidy = TRUE)
d
```

```
    est   lcl    ucl             x
1 56.57 31.28 105.14    Maintained
2 22.33 14.04  33.99 Nonmaintained
```

```
p <- ggplot(leukemia, aes(x = x, y = time)) +
  geom_dotplot(aes(fill = status), stackdir = "center", binaxis = "y",
    binwidth = 1, dotsize = 2, alpha = 0.5) + coord_flip() +
  scale_fill_manual(name = "Status", values = c("white","black")) +
```

```
  geom_pointrange(aes(y = est, ymin = lcl, ymax = ucl),
    shape = 3, data = d) +
  labs(x = "Maintained", y = "Remission Time (weeks)") +
  theme_classic() +
  theme(legend.position = "inside", legend.position.inside = c(0.8, 0.8))
plot(p)
```



A very useful feature of the **flexsurv** package is that a user can program their own distribution for use with the functions therein.

Proportional Hazards Models

Let $h_0(t)$ be the "baseline" hazard function (i.e., the hazard function when all $x_j = 0$). A *proportional hazards model* has the form

$$h_i(t) = h_0(t)e^{\beta_1 x_{i1}}e^{\beta_2 x_{i2}} \cdots e^{\beta_k x_{ik}},$$

so that $h_i(t) \propto e^{\beta_1 x_{i1}}e^{\beta_2 x_{i2}} \cdots e^{\beta_k x_{ik}}$. Thus increasing x_j by one changes the hazard function by a factor of e^{β_j}. This is the *hazard ratio*. For example, the hazard ratio for x_1 is

$$\frac{h_0(t)e^{\beta_1(x_1+1)}e^{\beta_2 x_2} \cdots e^{\beta_k x_k}}{h_0(t)e^{\beta_1 x_1}e^{\beta_2 x_2} \cdots e^{\beta_k x_k}} = e^{\beta_1},$$

since $e^{\beta_1(x_1+1)} = e^{\beta_1 x_1}e^{\beta_1}$.

Parametric Proportional Hazards Models

AFT models with a Weibull distribution (or exponential, which is a special case of the Weibull distribution) are also proportional hazards models. Consider the AFT model,

$$\log T_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + \sigma\epsilon_i,$$

and the proportional hazards model

$$h_i(t) = h_0(t)\exp(\beta_1^* x_{i1} + \beta_2^* x_{i2} + \cdots + \beta_k^* x_{ik}),$$

where in both cases T_i has a *Weibull* distribution. It can be shown that the models are equivalent with

$$\beta_j^* = -\beta_j/\sigma.$$

The hazard ratios are $e^{\beta_j^*}$.

An AFT model with a Weibull distribution is the *only* AFT model that is also a proportional hazards model. Other proportional hazards models exist, but none of the them are AFT models.

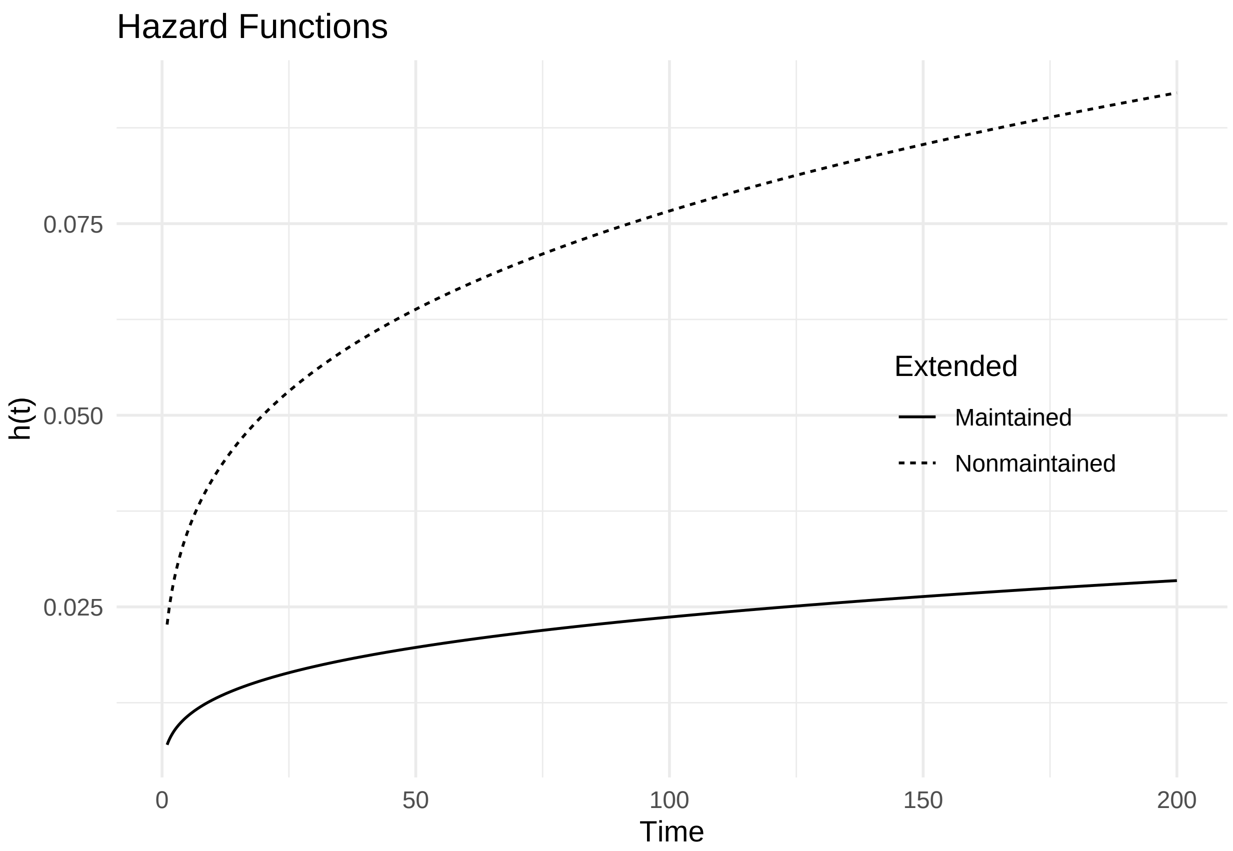
Example: Use `dist = "weibullPH"` with `flexsurvreg` for a parameterization of the Weibull distribution that gives hazard ratios.

```
m <- flexsurvreg(Surv(time, status == "out") ~ x, dist = "weibullPH", data = leukemia)
print(m)
```

```
Call:
flexsurvreg(formula = Surv(time, status == "out") ~ x, data = leukemia,
    dist = "weibullPH")

Estimates:
                data mean  est       L95%      U95%      se        exp(est)  L95%      U95%
shape                 NA   1.264295  0.891546  1.792889  0.225328        NA        NA        NA
scale                 NA   0.005544  0.000739  0.041565  0.005698        NA        NA        NA
xNonmaintained  0.521739   1.174962  0.149832  2.200092  0.523035  3.238021  1.161640  9.025845

N = 23,  Events: 18,  Censored: 5
Total time at risk: 678
Log-likelihood = -80.52, df = 3
AIC = 167
```

The proportionality can be seen when plotting the hazard functions.

```
d <- data.frame(x = c("Maintained","Nonmaintained"))
d <- summary(m, newdata = d, t = seq(1, 200, length = 1000),
  type = "hazard", tidy = TRUE)

p <- ggplot(d, aes(x = time, y = est)) +
  geom_line(aes(linetype = x)) + theme_minimal() +
  labs(x = "Time", y = "h(t)", linetype = "Extended", title = "Hazard Functions") +
  theme(legend.position = "inside", legend.position.inside = c(0.8, 0.5))
plot(p)
```



Example: Consider a Weibull proportional hazards model for the `motors` data.

```
m <- flexsurvreg(Surv(time, cens) ~ temp, data = MASS::motors, dist = "weibullPH")
print(m)
```

```
Call:
flexsurvreg(formula = Surv(time, cens) ~ temp, data = MASS::motors, 
    dist = "weibullPH")

Estimates: 
       data mean  est        L95%       U95%       se         exp(est)  L95%       U95%     
shape        NA   2.99e+00   1.96e+00   4.56e+00   6.42e-01         NA         NA         NA
scale        NA   6.34e-22   1.46e-30   2.76e-13   6.43e-21         NA         NA         NA
temp   1.82e+02   1.36e-01   7.92e-02   1.92e-01   2.87e-02   1.15e+00   1.08e+00   1.21e+00

N = 40,  Events: 17,  Censored: 23
Total time at risk: 140654
Log-likelihood = -147.4, df = 3
AIC = 300.7
```

Here we have that

$$h_{x+1}(t) = e^{\beta_1^*} h_x(t),$$

where $h_x(t)$ and $h_{x+1}(t)$ represent the hazard functions at temperatures of x and $x+1$, respectively. The estimated hazard ratio is $e^{\hat{\beta}_1^*} = 1.15$.

```
d <- summary(m, newdata = data.frame(temp = seq(110, 150, by = 10)),
  t = seq(0, 8000, length = 1000), type = "hazard", tidy = TRUE, ci = FALSE)
```

```r
p <- ggplot(d, aes(x = time, y = est, color = factor(temp))) +
  geom_line() + theme_minimal() +
  theme(legend.position = "inside", legend.position.inside = c(0.2, 0.6)) +
  labs(x = "Hours", y = "h(t)", color = "Temperature", title = "Hazard Functions")
plot(p)
```