14

Survival analysis

The analysis of lifetimes is an important topic within biology and medicine in particular but also in reliability analysis with engineering applications. Such data are often highly nonnormally distributed, so that the use of standard linear models is problematic.

Lifetime data are often *censored:* You do not know the exact lifetime, only that it is longer than a given value. For instance, in a cancer trial, some people are lost to follow-up or simply live beyond the study period. It is an error to ignore the censoring in the statistical analysis, sometimes with extreme consequences. Consider, for instance, the case where a new treatment is introduced towards the end of the study period, so that nearly all the observed lifetimes will be cut short.

14.1 Essential concepts

Let X be the true lifetime and T a censoring time. What you observe is the minimum of X and T together with an indication of whether it is one or the other. T can be a random variable or a fixed time depending on context, but if it is random, then it should generally be *noninformative* for the methods we describe here to be applicable. Sometimes "dead from other causes" is considered a censoring event for the mortality of a given

P. Dalgaard, *Introductory Statistics with R*,
DOI: 10.1007/978-0-387-79054-1_14,

disease, and in those cases it is particularly important to ensure that these other causes are unassociated with the disease state.

The *survival function* $S(t)$ measures the probability of being alive at a given time. It is really just 1 minus the cumulative distribution function for X, $1 - F(t)$.

The *hazard function* or *force of mortality* $h(t)$ measures the (infinitesimal) risk of dying within a short interval of time t, given that the subject is alive at time t. If the lifetime distribution has density f, then $h(t) = f(t)/S(t)$. This is often considered a more fundamental quantity than (say) the mean or median of the survival distribution and is used as a basis for modelling.

14.2 Survival objects

We use the package `survival`, written by Terry Therneau and ported to R by Thomas Lumley. The package implements a large number of advanced techniques. For the present purposes, we use only a small subset of it.

To load `survival`, use

```
> library(survival)
```

(This may produce a harmless warning about masking the `lung` data set from the `ISwR` package.)

The routines in `survival` work with objects of class `"Surv"`, which is a data structure that combines times and censoring information. Such objects are constructed using the `Surv` function, which takes two arguments: an observation time and an event indicator. The latter can be coded as a logical variable, a 0/1 variable, or a 1/2 variable. The latter coding is not recommended since `Surv` will assume 0/1 coding if all values are 1.

Actually, `Surv` can also be used with three arguments for dealing with data that have a start time as well as an end time ("staggered entry") and also interval censored data (where you know that an event happened between two dates, as happens, for instance, in repeated testing for a disease) can be handled.

We use the data set `melanom` collected by K. T. Drzewiecki and reproduced in Andersen et al. (1991). The data become accessible as follows:

```
> attach(melanom)
> names(melanom)
```

```
[1] "no"     "status" "days"   "ulc"    "thick"  "sex"
```

The variable `status` is an indicator of the patient's status by the end of the study: 1 means "dead from malignant melanoma", 2 means "alive on January 1, 1978", and 3 means "dead from other causes". The variable `days` is the observation time in days, `ulc` indicates (1 for present and 2 for absent) whether the tumor was ulcerated, `thick` is the thickness in 1/100 mm, and `sex` contains the gender of the patient (1 for women and 2 for men).

We want to create a `Surv` object in which we consider the values 2 and 3 of the `status` variable as censorings. This is done as follows:

```
> Surv(days, status==1)
  [1]   10+   30+   35+   99+  185   204   210   232   232+  279
 [11]  295   355+  386   426   469   493+  529   621   629   659
 [21]  667   718   752   779   793   817   826+  833   858   869
...
[181] 3476+ 3523+ 3667+ 3695+ 3695+ 3776+ 3776+ 3830+ 3856+ 3872+
[191] 3909+ 3968+ 4001+ 4103+ 4119+ 4124+ 4207+ 4310+ 4390+ 4479+
[201] 4492+ 4668+ 4688+ 4926+ 5565+
```

Associated with the `Surv` objects is a print method that displays the objects in the format above, with a '+' marking censored observations. For example, `10+` means that the patient did not die from melanoma within 10 days and was then unavailable for further study (in fact, he died from other causes), whereas `185` means that the patient died from the disease a little over half a year after his operation.

Notice that the second argument to `Surv` is a logical vector; `status==1` is `TRUE` for those who died of malignant melanoma and `FALSE` otherwise.

14.3 Kaplan–Meier estimates

The Kaplan–Meier estimator allows the computation of an estimated survival function in the presence of right-censoring. It is also called the *product-limit estimator* because one way of describing the procedure is that it multiplies together conditional survival curves for intervals in which there are either no censored observations or no deaths. This becomes a step function where the estimated survival is reduced by a factor $(1 - 1/R_t)$ if there is a death at time t and a population of R_t is still alive and uncensored at that time.

Computing the Kaplan–Meier estimator for the survival function is done with a function called `survfit`. In its simplest form, it takes just a single

argument, namely a `Surv` object. It returns a `survfit` object. As described above, we consider "dead from other causes" a kind of censoring and do as follows:

```
> survfit(Surv(days,status==1))
Call: survfit(formula = Surv(days, status == 1))

      n  events  median 0.95LCL 0.95UCL
    205      57     Inf     Inf     Inf
```

As is seen, using `survfit` by itself is not very informative (just as the printed output of a "bare" `lm` is not). You get a couple of summary statistics and an estimate of the median survival, and in this case the latter is not even interesting because the estimate is infinite. The survival curve does not cross the 50% mark before all patients are censored.

To see the actual Kaplan–Meier estimate, use `summary` on the `survfit` object. We first save the `survfit` object into a variable, here named `surv.all` because it contains the raw survival function for all patients without regard to patient characteristics.

```
> surv.all <- survfit(Surv(days,status==1))
> summary(surv.all)
Call: survfit(formula = Surv(days, status == 1))

 time n.risk n.event survival std.err lower 95% CI upper 95% CI
  185    201       1    0.995 0.00496        0.985        1.000
  204    200       1    0.990 0.00700        0.976        1.000
  210    199       1    0.985 0.00855        0.968        1.000
  232    198       1    0.980 0.00985        0.961        1.000
  279    196       1    0.975 0.01100        0.954        0.997
  295    195       1    0.970 0.01202        0.947        0.994
...
 2565     63       1    0.689 0.03729        0.620        0.766
 2782     57       1    0.677 0.03854        0.605        0.757
 3042     52       1    0.664 0.03994        0.590        0.747
 3338     35       1    0.645 0.04307        0.566        0.735
```

This contains the values of the survival function at the event times. The censoring times are not displayed but are contained in the `survfit` object and can be obtained by passing `censored=T` to `summary` (see the help page for `summary.survfit` for such details).

The Kaplan–Meier estimate is the step function whose jump points are given in `time` and whose values right after a jump are given in `survival`. Additionally, both an estimate of the standard error of the curve and a (pointwise) confidence interval for the true curve are given.

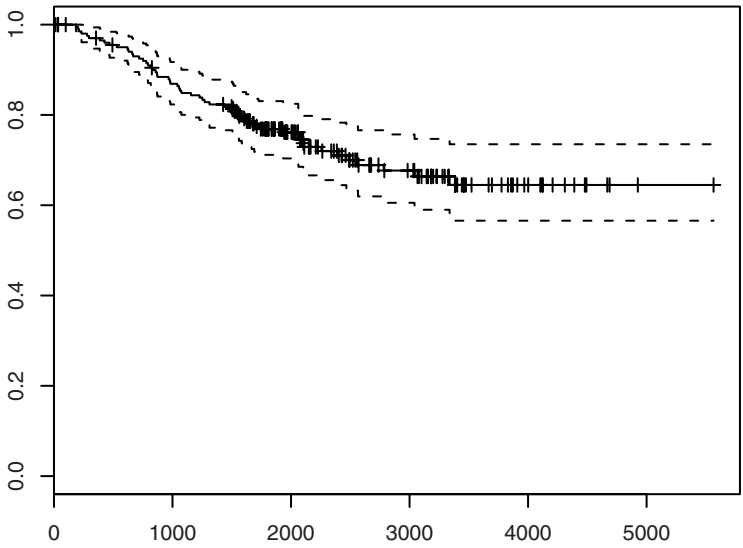

Figure 14.1. Kaplan–Meier plot for melanoma data (all observations).

Normally, you would be more interested in showing the Kaplan–Meier estimate graphically than numerically. To do this (Figure 14.1), you simply write

```
> plot(surv.all)
```

The markings on the curve indicate censoring times, and the bands give approximate confidence intervals. If you look closely, you will see that the bands are not symmetrical around the estimate. They are constructed as a symmetric interval on the log scale and transformed back to the original scale.

It is often useful to plot two or more survival functions on the same plot so that they can be directly compared (Figure 14.2). To obtain survival functions split by gender, do the following:

```
> surv.bysex <- survfit(Surv(days,status==1)~sex)
> plot(surv.bysex)
```

That is, you use a model formula as in `lm` and `glm`, specifying that the survival object generated from `day` and `status` should be described by `sex`. Notice that there are no confidence intervals on the curves. These are

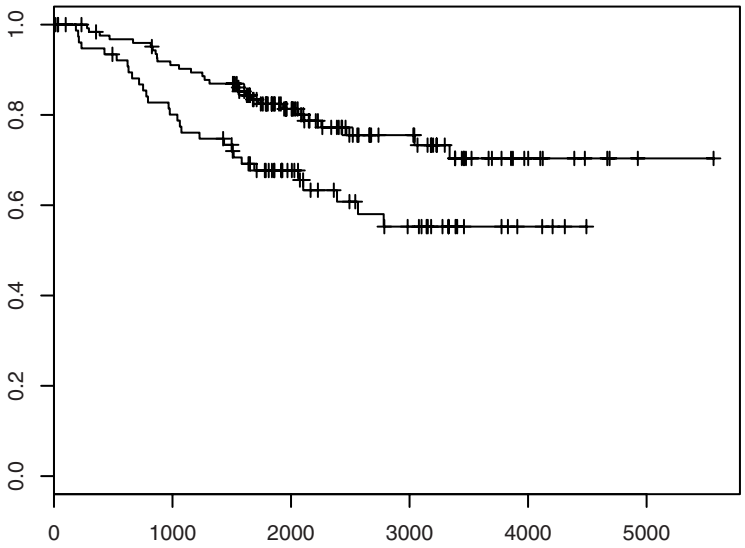

Figure 14.2. Kaplan–Meier plots for melanoma data, grouped by gender.

turned off when there are two or more curves because the display easily becomes confusing. They can be turned on again by passing `conf.int=T` to `plot`, in which case it can be recommended to use separate colours for the curves, as in

```
> plot(surv.bysex, conf.int=T, col=c("black","gray"))
```

Similarly, you can avoid plotting the confidence bands in the single-sample case by setting `conf.int=F`. If you want the bands but at a 99% confidence level, you should pass `conf.int=0.99` to `survfit`. Notice that the level of confidence is an argument to the fitting function (which needs it to compute the confidence limits), whereas the decision to plot the bands is controlled by a similarly named argument to plot.

14.4 The log-rank test

The log-rank test is used to test whether two or more survival curves are identical. It is based on looking at the population at each death time and computing the expected number of deaths in proportion to the number of

individuals at risk in each group. This is then summed over all death times and compared with the observed number of deaths by a procedure similar (but not identical) to the χ^2 test. Notice that the interpretation of "expected" and "observed" is slightly peculiar: If the difference in mortality is sufficiently large, then you can easily "expect" the same individuals to die several times over the course of the trial. If the population is observed to extinction with no censoring, then the observed number of deaths will equal the group size by definition and the expected values will contain all the random variation.

The log-rank test is formally nonparametric since the distribution of the test statistic depends only on the assumption that the groups have the same survival function. However, it can also be viewed as a model-based test under the assumption of *proportional hazards* (see Section 14.1). You can set up a semiparametric model in which the hazard itself is unspecified but it is assumed that the hazards are proportional between groups. Testing that the proportionality factors are all unity then leads to a log-rank test. The log-rank test will work best against this class of alternatives.

Computation of the log-rank test is done by the function `survdiff`. This actually implements a whole family of tests specified by a parameter ρ, allowing various nonproportional hazards alternatives to the null hypothesis, but the default value of $\rho = 0$ gives the log-rank test.

```
> survdiff(Surv(days,status==1)~sex)
Call:
survdiff(formula = Surv(days, status == 1) ~ sex)

        N Observed Expected (O-E)^2/E (O-E)^2/V
sex=1 126       28     37.1      2.25      6.47
sex=2  79       29     19.9      4.21      6.47

 Chisq= 6.5  on 1 degrees of freedom, p= 0.011
```

The specification is using a model formula as for linear and generalized linear models. However, the test can deal only with grouped data, so if you specify multiple variables on the right-hand side it will work on the grouping of data generated by all combinations of predictor variables. It also makes no distinction between factors and numerical codes. The same is true of `survfit`.

It is also possible to specify stratified analyses, in which the observed and expected value calculations are carried out separately within a stratification of the data set. For instance, you can compute the log-rank test for a gender effect stratified by ulceration as follows:

```
> survdiff(Surv(days,status==1)~sex+strata(ulc))
Call:
```

```
survdiff(formula = Surv(days, status == 1) ~ sex + strata(ulc))

        N Observed Expected (O-E)^2/E (O-E)^2/V
sex=1 126       28     34.7      1.28      3.31
sex=2  79       29     22.3      1.99      3.31

 Chisq= 3.3  on 1 degrees of freedom, p= 0.0687
```

Notice that this makes the effect of `sex` appear less significant. A possible explanation might be that males seek treatment when the disease is in a more advanced state than women do, so that the gender difference is reduced when adjusted for a measure of disease progression.

14.5 The Cox proportional hazards model

The proportional hazards model allows the analysis of survival data by regression models similar to those of `lm` and `glm`. The scale on which linearity is assumed is the log-hazard scale. Models can be fitted via the maximization of *Cox's likelihood*, which is not a true likelihood but it can be shown that it may be used as one. It is calculated in a manner similar to that of the log-rank test, as the product of conditional likelihoods of the observed death at each death time.

As a first example, consider a model with the single regressor `sex`:

```
> summary(coxph(Surv(days,status==1)~sex))
Call:
coxph(formula = Surv(days, status == 1) ~ sex)

  n= 205
     coef exp(coef) se(coef)   z     p
sex 0.662      1.94    0.265 2.5 0.013

    exp(coef) exp(-coef) lower .95 upper .95
sex      1.94      0.516      1.15      3.26

Rsquare= 0.03   (max possible= 0.937 )
Likelihood ratio test= 6.15  on 1 df,   p=0.0131
Wald test            = 6.24  on 1 df,   p=0.0125
Score (logrank) test = 6.47  on 1 df,   p=0.0110
```

The `coef` is the estimated logarithm of the hazard ratio between the two groups, which for convenience is also given as the actual hazard ratio `exp(coef)`. The line following that also gives the inverted ratio (swapping the groups) and confidence intervals for the hazard ratio. Finally, three overall tests for significant effects in the model are given. These are all equivalent in large samples but may differ somewhat in small-sample

cases. Notice that the Wald test is identical to the z test based on the estimated coefficient divided by its standard error, whereas the score test is equivalent to the log-rank test (as long as the model involves only a simple grouping).

A more elaborate example, involving a continuous covariate and a stratification variable, is

```
> summary(coxph(Surv(days,status==1)~sex+log(thick)+strata(ulc)))
Call:
coxph(formula = Surv(days, status == 1) ~ sex + log(thick) +
    strata(ulc))

  n= 205

            coef exp(coef) se(coef)    z      p
sex         0.36      1.43    0.270 1.33 0.1800
log(thick) 0.56      1.75    0.178 3.14 0.0017

            exp(coef) exp(-coef) lower .95 upper .95
sex              1.43      0.698     0.844      2.43
log(thick)       1.75      0.571     1.234      2.48

Rsquare= 0.063   (max possible= 0.9 )
Likelihood ratio test= 13.3  on 2 df,   p=0.00130
Wald test            = 12.9  on 2 df,   p=0.00160
Score (logrank) test = 13.0  on 2 df,   p=0.00152
```

It is seen that the significance of the sex variable has been further reduced.

The Cox model assumes an underlying baseline hazard function with a corresponding survival curve. In a stratified analysis, there will be one such curve for each stratum. They can be extracted by using survfit on the output of coxph and of course be plotted using the plot method for survfit objects (Figure 14.3):

```
> plot(survfit(coxph(Surv(days,status==1)~
+              log(thick)+sex+strata(ulc))))
```

Be aware that the default for survfit is to generate curves for a pseudo-individual for which the covariates are at their mean values. In the present case, that would correspond to a tumor thickness of 1.86 mm and a gender of 1.39 (!). Notice that we have been sloppy in not defining sex as a factor variable, but that would not actually give a different result (coxph subtracts the means of the regressors before fitting, so a 1/2 coding is the same as 0/1, which is what a factor with treatment contrasts gives you). However, you can use the newdata argument of survfit to specify a data frame for which you want to calculate survival curves.

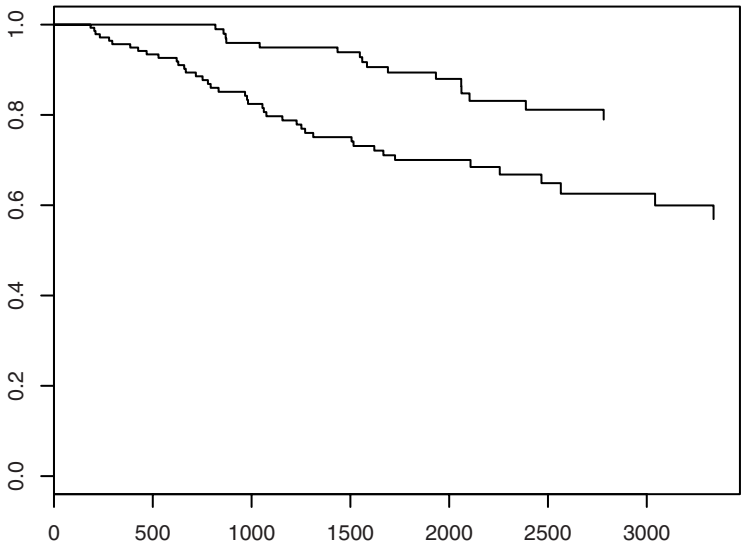

Figure 14.3. Baseline survival curves (ulcerated and nonulcerated tumors) in stratified Cox regression.

14.6 Exercises

14.1 In the `graft.vs.host` data set, estimate the survival function for patients with or without GVHD. Test the hypothesis that the survival is the same in both groups. Extend the analysis by including the other explanatory variables.

14.2 With the Cox model in the last section of the text, generate a plot with estimated survival curves for men with nonulcerated tumors of thicknesses 0.1, 0.2, and 0.5 mm (three curves in one plot). Hint: `survfit` objects can be indexed with `[]` to extract individual strata.

14.3 Fit Cox models to the `stroke` data with `age` and `sex` as predictors and with `sex` alone. Explain the difference.

14.4 With the split data from Exercise 10.4, you can fit a Cox model with *delayed entry* to the stroke data; `help(Surv)` shows how to set up the `Surv` object in that case. Refit the model(s) from the previous exercise.