flexsurv: a platform for parametric survival modelling in R

Christopher H. Jackson
MRC Biostatistics Unit, Cambridge, UK
chris.jackson@mrc-bsu.cam.ac.uk

Abstract

flexsurv is an R package for fully-parametric modelling of survival data. Any parametric time-to-event distribution may be fitted if the user supplies at minimum a probability density or hazard function. Many survival distributions are built in, including standard models models and the three and four-parameter generalized gamma and F distributions. Any parameter of the distribution can be modelled as a linear or log-linear function of covariates. Another built-in model is the spline model of Royston and Parmar, in which both baseline survival and covariate effects can be arbitrarily flexible parametric functions of time. The main model-fitting function, `flexsurvreg`, uses the familiar syntax of `survreg` from the standard **survival** package. Censoring or left-truncation are specified in `Surv` objects. Estimates and confidence intervals for any function of the model parameters can be printed or plotted. **flexsurv** also enhances the **mstate** package (Putter et al) by providing cumulative hazards for fully-parametric multi-state models. This article explains the methods and design principles of the package, giving several worked examples of its use.

Keywords:˜survival.

1. Motivation and design

The Cox model for survival data is ubiquitous in medical research, since the effects of predictors can be estimated without needing to supply a baseline survival distribution that might be inaccurate. However, fully-parametric models have many advantages, and even the originator of the Cox model has expressed a preference for parametric modelling (see Reid 1994). Fully-specified models help to understand the change in hazard through time, and help with prediction and extrapolation. For example, the mean survival $E(T) = \int_0^\infty S(t)$, used in health economic evaluations (Latimer 2013), needs the survivor function $S(t)$ to be fully-specified for all times t.

flexsurv allows parametric distributions of arbitrary complexity to be fitted to survival data, gaining the convenience of parametric modelling, while avoiding the risk of model misspecification. Built-in choices include splines with any number of knots (Royston and Parmar 2002) and 3–4 parameter generalized gamma and F distribution families. Any user-defined model may be employed by supplying at minimum an R function to compute the probability density or hazard, and ideally also its cumulative form. Any parameters may be modelled in terms of covariates, and any function of the parameters may be printed or plotted in model

summaries.

flexsurv is intended as a general platform for survival modelling in R. The `survreg` function in the R package **survival** (Therneau 2014) only supports two-parameter (location/scale) distributions, though users can supply their own distributions if they can be parameterised in this form. Many other contributed R packages can fit survival models, e.g. **eha** (Broström 2014), **VGAM** (Yee and Wild 1996), though these are either limited to specific distribution families, not specifically designed for survival analysis, or (**ActuDistns** Nadarajah and Bakar 2013) contain only the definitions of distribution functions. **flexsurv** enables distribution functions provided by such packages to be used as survival models.

It is similar in spirit to the Stata packages **stpm2** (Lambert and Royston 2009) for spline-based survival modelling, and **stgenreg** (Crowther and Lambert 2013) for fitting survival models with user-defined hazard functions using numerical integration. Though in **flexsurv**, numerical integration can be avoided if the analytic cumulative distribution or hazard can be supplied, and optimisation can also be speeded by supplying analytic derivatives. **flexsurv** also has features for multi-state modelling and interval censoring, and general output reporting. It employs functional programming to work with user-defined or existing R functions.

2. General parametric survival model

2.1. Definitions

The general model that **flexsurv** fits has probability density function

$$f(t|\mu(\mathbf{z}), \boldsymbol{\alpha}(\mathbf{z})), \quad t \geq 0 \tag{1}$$

The cumulative distribution function $F(t)$, survivor function $S(t) = 1 - F(t)$, cumulative hazard $H(t) = -\log S(t)$ and hazard $h(t) = f(t)/S(t)$ are also defined (suppressing the conditioning for clarity). $\mu = \alpha_0$ is the parameter of primary interest, which usually governs the mean or location of the distribution. Other parameters $\boldsymbol{\alpha} = \alpha_1, \ldots, \alpha_R$ are called "ancillary" and determine the shape, variance or higher moments.

Covariates All parameters may depend on a vector of covariates $\mathbf{z}$ through link-transformed linear models $g_0(\mu) = \gamma_0 + \boldsymbol{\beta}_0'\mathbf{z}$ and $g_r(\alpha_r) = \gamma_r + \boldsymbol{\beta}_r'\mathbf{z}$. $g()$ will typically be log() if the parameter is defined to be positive, or the identity function if the parameter is unrestricted.

Suppose that the location, but not the ancillary parameters, depends on covariates. If the hazard function factorises as $h(t|\alpha, \mu(\mathbf{z})) = \mu(\mathbf{z})h_0(t|\alpha)$, then this is a *proportional hazards* (PH) model, so that the hazard ratio between two groups (defined by different values of $\mathbf{z}$) is constant over time.

Alternatively, if $S(t|\mu(\mathbf{z}), \alpha) = S(\mu(\mathbf{z})t|\alpha)$ then we have an *accelerated failure time* (AFT) model, so that the effect of covariates is to speed or slow the passage of time. For example, doubling the value of a covariate with coefficient $\beta = \log(2)$ would give half the expected survival time.

Data and likelihood Let $t_i : i = 1, \ldots, n$ be a sample of times from individuals i. Let $c_i = 1$ if t_i is an observed death time, or $c_i = 0$ if t_i is a right-censoring time, thus the true

death time is known only to be greater than t_i. Also let s_i be corresponding left-truncation (or delayed-entry) times, meaning that individual i is only observed conditionally on having survived up to s_i, thus $s_i = 0$ if there is no left-truncation. This facilitates time-dependent covariates (§3) and multi-state models (§5). Additionally let t_i^{max} be left-censoring times. If there is no left-censoring then these are infinite, so that $S(t_i^{max}) = 0$; or if the ith death time is interval-censored then $c_i = 0$ and t_i^{max} is finite.

The likelihood for the parameters $\boldsymbol{\theta} = \{\boldsymbol{\gamma}, \boldsymbol{\beta}\}$ in model (1), given the corresponding data vectors, is

$$l(\{\boldsymbol{\theta}\}|\mathbf{t}, \mathbf{c}, \mathbf{s}, \mathbf{t}^{max}) = \left\{ \prod_{i:\ c_i=1} f_i(t_i) \prod_{i:\ c_i=0} (S_i(t_i) - S_i(t_i^{max})) \right\} / \prod_i S_i(s_i) \qquad (2)$$

The individuals are independent, so that **flexsurv** does not currently support frailty, clustered or random effects models (see §6).

An example dataset used throughout this paper is from 686 patients with primary node positive breast cancer, available in the package as `bc`. This was originally provided with `stpm` (Royston 2001), and analysed in much more detail by Royston and Parmar (2002) and Sauerbrei and Royston (1999).

3. Model fitting syntax

The main model-fitting function is called `flexsurvreg`. Its first argument is an R *formula* object. The left hand side of the formula gives the response as a survival object, using the `Surv` function from the **survival** package. Here, this indicates that the response variable is `recyrs`, which represents observed death or censoring times when the variable `censrec` is 1 or 0 respectively. The covariate `group` is a factor representing a prognostic score, with three levels `"Good"` (the baseline), `"Medium"` and `"Poor"`. All of these variables are in the data frame `bc`.

```
> library(flexsurv)
> fs1 <- flexsurvreg(Surv(recyrs, censrec) ~ group, data=bc, dist="weibull")
```

If the argument `dist` is a string, this denotes a built-in survival distribution. In this case we fit a Weibull survival model. Printing the fitted model object gives estimates and confidence intervals for the model parameters and other useful information. Note that these are the *same parameters* as represented by the R distribution function `dweibull`: the `shape` α and the `scale` μ of the survivor function $S(t) = \exp(-(t/\mu)^\alpha)$, and `group` has a linear effect on $\log(\mu)$.

```
> fs1

Call:
flexsurvreg(formula = Surv(recyrs, censrec) ~ group, data = bc,     dist = "weibull")

Estimates:
             data mean  est      L95%     U95%     se       exp(est)  L95%
```

```
shape             NA     1.3797   1.2548   1.5170   0.0668       NA        NA
scale             NA    11.4229   9.1818  14.2110   1.2728       NA        NA
groupMedium   0.3338    -0.6136  -0.8623  -0.3649   0.1269   0.5414    0.4222
groupPoor     0.3324    -1.2122  -1.4583  -0.9661   0.1256   0.2975    0.2326
              U95%
shape             NA
scale             NA
groupMedium   0.6943
groupPoor     0.3806

N = 686,  Events: 299,  Censored: 387
Total time at risk: 2113.425
Log-likelihood = -811.9419, df = 4
AIC = 1631.884
```

The same model can be fitted using `survreg` in **survival**:

```
> survreg(Surv(recyrs, censrec) ~ group, data=bc, dist="weibull")

Call:
survreg(formula = Surv(recyrs, censrec) ~ group, data = bc, dist = "weibull")

Coefficients:
(Intercept) groupMedium   groupPoor
  2.4356168  -0.6135892  -1.2122137

Scale= 0.7248206

Loglik(model)= -811.9   Loglik(intercept only)= -873.2
        Chisq= 122.53 on 2 degrees of freedom, p= 0
n= 686
```

The maximised log-likelihoods are the same, however the parameterisation is different: the first coefficient `(Intercept)` reported by `survreg` is $\log(\mu)$, and `survreg`'s `"scale"` is `dweibull`'s (thus `flexsurvreg`)'s 1 / `shape`. The covariate effects $\boldsymbol{\beta}$, however, have the same "accelerated failure time" interpretation, as linear effects on $\log(\mu)$. The multiplicative effects $\exp(\boldsymbol{\beta})$ are printed in the output as `exp(est)`.

Alternative censoring or truncation mechanisms If we also had left-truncation times in a variable called `start`, the response would be `Surv(start,recyrs,censrec)`. Or if all responses were interval-censored between lower and upper bounds `tmin` and `tmax`, then we would write `Surv(tmin,tmax,type="interval2")`.

Note that just as in `survreg`, time-dependent covariates can be represented through left-truncation. For each individual there would be multiple records, each corresponding to an interval where the covariate is assumed to be constant. The response would be of the form `Surv(start,stop,censrec)`, where `start` and `stop` are the limits of each interval, and `censrec` indicates whether a death was observed at `stop`.

3.1. Built-in survival models

`flexsurvreg`'s currently built-in distributions are listed in Table 1. In each case, the probability density $f()$ and parameters of the fitted model are taken from an existing R function of the same name but beginning with the letter `d`. For the Weibull, exponential (`dexp`), gamma (`dgamma`) and log-normal (`dlnorm`), the density functions are provided with standard installations of R. These density functions, and the corresponding cumulative distribution function (with the first letter `d` replaced by `p`) are used internally in `flexsurvreg` to compute the likelihood.

flexsurv provides some additional survival distributions, including a Gompertz distribution with unrestricted shape parameter (`dist="gompertz"`), and the three- and four-parameter families described below. For all built-in distributions, **flexsurv** also defines functions beginning `h` giving the hazard, and `H` for cumulative hazard.

Generalized gamma This three-parameter distribution includes the Weibull, gamma and log-normal as special cases. The original parameterisation from Stacy (1962) is available as `dist="gengamma.orig"`, however the newer parameterisation (Prentice 1974) is preferred: `dist="gengamma"`. This has parameters (μ,σ,q), and survivor function

$$\begin{array}{ll} 1 - I(\gamma, u) & (q > 0) \\ 1 - \Phi(z) & (q = 0) \end{array}$$

where $I(a, x) = \int_0^x x^{a-1} \exp(-x)/\Gamma(a)$ is the incomplete gamma function (the cumulative gamma distribution with shape a and scale 1), Φ is the standard normal cumulative distribution, $u = \gamma \exp(|q|z)$, $z = (\log(t) - \mu)/\sigma$, and $\gamma = q^{-2}$. The Prentice (1974) parameterisation extends the original one to include a further class of models with negative q, and survivor function $I(\gamma, u)$, where z is replaced by $-z$. This stabilises estimation when the distribution is close to log-normal, since $q = 0$ is no longer near the boundary of the parameter space. In R notation, [1] the parameter values corresponding to the three special cases are

```
dgengamma(x, mu, sigma, Q=0)     ==  dlnorm(x, mu, sigma)
dgengamma(x, mu, sigma, Q=1)     ==  dweibull(x, shape=1/sigma, scale=exp(mu))
dgengamma(x, mu, sigma, Q=sigma) ==  dgamma(x, shape=1/sigma^2,
                                            rate=exp(-mu) / sigma^2)
```

The generalized gamma model is fitted to the breast cancer survival data. `fs2` is an AFT model, where only the parameter μ depends on the prognostic covariate `group`. In a second model `fs3`, the first ancillary parameter `sigma` (α_1) also depends on this covariate, giving a model with a time-dependent effect that is neither PH nor AFT. The second ancillary parameter `Q` is still common between prognostic groups.

```
> fs2 <- flexsurvreg(Surv(recyrs, censrec) ~ group, data=bc, dist="gengamma")
> fs3 <- flexsurvreg(Surv(recyrs, censrec) ~ group + sigma(group),
+                    data=bc, dist="gengamma")
```

[1] The parameter called q here and in previous literature is called Q in `dgengamma` and related functions, since the first argument of a cumulative distribution function is conventionally named `q`, for quantile, in R.

	Parameters	Density R function	dist
Exponential	rate	dexp	"exp"
Weibull	shape, scale	dweibull	"weibull"
Gamma	shape, rate	dgamma	"gamma"
Log-normal	meanlog, sdlog	dlnorm	"lnorm"
Gompertz	shape, rate	dgompertz	"gompertz"
Generalized gamma (Prentice 1975)	mu, sigma, Q	dgengamma	"gengamma"
Generalized gamma (Stacy 1962)	shape, scale, k	dgengamma.orig	"gengamma.orig"
Generalized F (stable)	mu, sigma, Q, P	dgenf	"genf"
Generalized F (original)	mu, sigma, s1, s2	dgenf.orig	"genf.orig"

Table 1: Built-in parametric survival distributions in **flexsurv**

Table 3 compares all the models fitted to the breast cancer data, showing absolute fit to the data as measured by the maximised -2× log likelihood $-2LL$, number of parameters p, and Akaike's information criterion $-2LL + 2p$ which estimates predictive ability.

Generalized F This four-parameter distribution includes the generalized gamma, and also the log-logistic, as special cases. The variety of hazard shapes that can be represented is discussed by Cox (2008). It is provided here in alternative "original" (`dist="genf.orig"`) and "stable" parameterisations (`dist="genf"`) as presented by Prentice (1975). See `help(GenF)` and `help(GenF.orig)` in the package documentation for the exact definitions.

3.2. Plotting outputs

The `plot()` method for `flexsurvreg` objects is used as a quick check of model fit. By default, this draws a Kaplan-Meier estimate of the survivor function $S(t)$, one for each combination of categorical covariates, or just a single "population average" curve if there are no categorical covariates. The corresponding estimates from the fitted model are overlaid. Fitted values from further models can be added with the `lines()` method.

`scale="hazard"` can be used to plot hazards from parametric models against kernel density estimates (obtained from **muhaz**, Hess and Gentleman 2010; Mueller and Wang 1994). This shows more clearly why the Weibull model is inadequate: the hazard must be increasing or decreasing — while the generalized gamma can represent the increase and subsequent decline in hazard seen in the data.

Similarly, `scale="cumhaz"` plots cumulative hazards. Confidence intervals are produced by simulating a large sample from the asymptotic normal distribution of the maximum likelihood estimates of $\{\boldsymbol{\beta}_r : r = 0, \ldots, R$, via the function `normboot.flexsurvreg`.

In this example, there is only a single categorical covariate, and the `plot` and `summary` methods return one observed and fitted trajectory for each level of that covariate. For more complicated models, users should specify exactly what covariate values they want summaries for, rather than relying on the default [2]. This is done by supplying the `newdata` argument, a data frame

[2]If there are only factor covariates, all combinations are plotted. If there are any continuous covariates, these methods by default return a "population average" curve, with the linear model design matrix set to its average values, including the 0/1 contrasts defining factors, which doesn't represent a meaningful covariate combination.

```
> plot(fs1, col="gray", lwd.obs=2)
> lines(fs2, col="red", lty=2)
> lines(fs3, col="red")
```

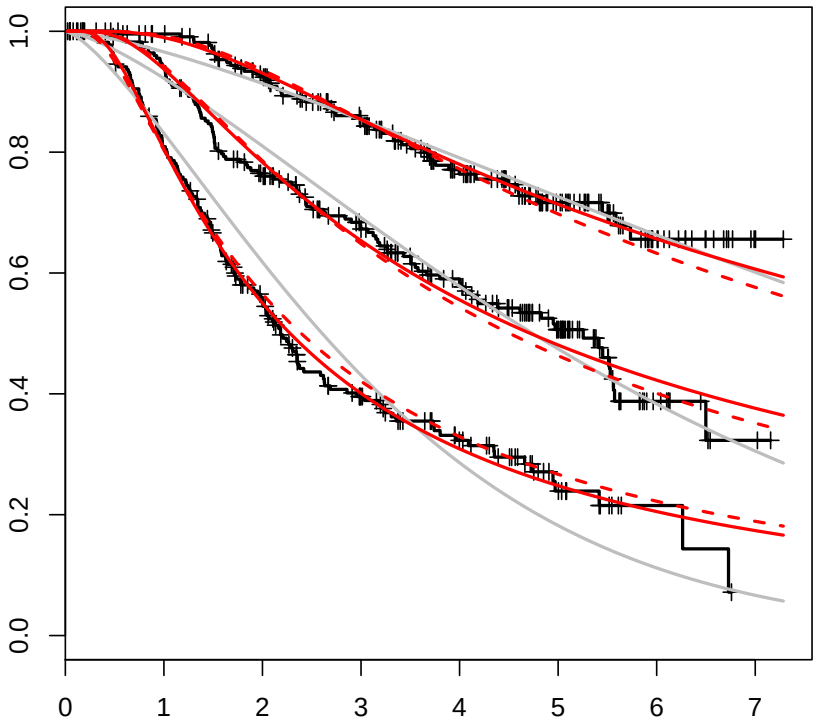

Figure 1: Estimated survival from parametric models and Kaplan-Meier estimates.

or list containing covariate values, just as in standard R functions like `predict.lm`.

For more than casual plots, it is advised to set up the axes beforehand, and use the `lines()` method. Or for even more flexibility, the data underlying the plots is available from the `summary.flexsurvreg()` method.

3.3. Custom model summaries

Any function of the parameters of a fitted model can be summarised or plotted by supplying the argument `fn` to `summary.flexsurvreg` or `plot.flexsurvreg`. This should be an R function, with mandatory first two arguments `t` representing time, and `start` representing a left-truncation point (so that the result is conditional on survival up to that time). The remaining arguments must be the parameters of the survival distribution. For example, median survival under the Weibull model `fs1` can be summarised as follows

```
> plot(fs1, type="hazard", col="gray", lwd.obs=2)
> lines(fs2, type="hazard", col="red", lty=2)
> lines(fs3, type="hazard", col="red")
```

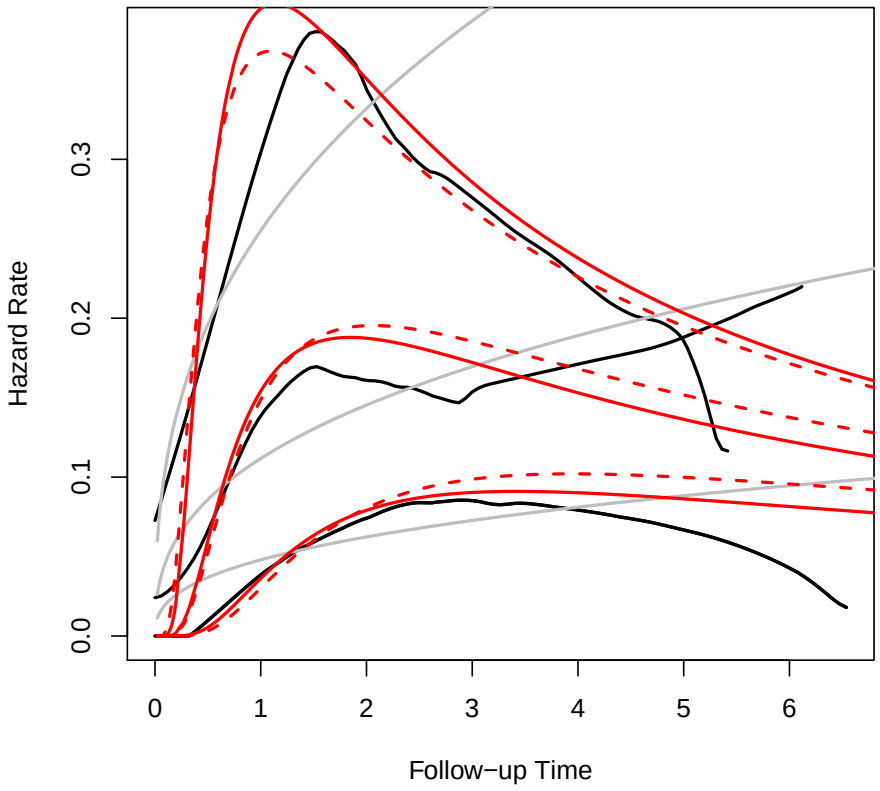

Figure 2: Estimated hazards from parametric models and kernel density estimates.

```
> median.weibull <- function(t, start, shape, scale) {
+     qweibull(0.5, shape=shape, scale=scale)
+ }
> summary(fs1, fn=median.weibull, t=1, B=10000)

group=Good
  time     est      lcl      ucl
1    1 8.75794 7.152142 10.82977

group=Medium
  time      est      lcl      ucl
1    1 4.741585 4.108802 5.443883

group=Poor
```

```
  time      est      lcl      ucl
1    1 2.605819 2.316351 2.937252
```

Although the median of the Weibull has an analytic form as $\mu \log(2)^{1/\alpha}$, the form of the code given here generalises to other distributions. The argument `t` is not used in `median.weibull`, because the median is a time-constant function of the parameters, unlike the survival or hazard. `10000` random samples are drawn to produce a slightly more precise confidence interval than the default — users should adjust this until the desired level of precision is obtained. A useful future extension of the package would be to allow users to supply derivatives of their custom summary function, so that the delta method can be used to obtain approximate confidence intervals without simulation.

3.4. Computation

The likelihood is maximised in `flexsurvreg` using the optimisation methods available through the standard R `optim` function. By default, this is the `"BFGS"` method ((Nash 1990)) using the analytic derivatives of the likelihood with respect to the model parameters, if these are available, to improve the speed of convergence to the maximum. These are built-in for the exponential, Weibull and Gompertz. For custom distributions, the user can optionally supply functions with names beginning `"DLd"` and `"DLS"` respectively (e.g. `DLdweibull,DLSweibull`) to calculate the derivatives of the log density and log survivor functions with respect to the transformed parameters γ.

Initial values are difficult: ideally two would come from moments of the distribution, then defaults that reduce to simpler distributions. example

3.5. Custom survival distributions

flexsurv is not limited to its built-in distributions. Any survival model of the form (1–2) can be fitted if we can provide either the density function $f()$ or the hazard $h()$. Many contributed R packages provide probability density and cumulative distribution functions for positive distributions. Though survival models may be more naturally characterised by their hazard function, representing the changing risk of death through time. For example, for survival following major surgery we may want a "U-shaped" hazard curve, representing a high risk soon after the operation, which then decreases, but increases naturally as survivors grow older.

To supply a custom distribution, the `dist` argument to `flexsurvreg` is defined to be an R list object, rather than a character string. The list has the following elements.

`name` Name of the distribution. For example, if this is `"llogis"` then there is assumed to be at least either

- a function called `dllogis` to compute the probability density, or
- `hllogis` to compute the hazard.

Ideally there will also be a function called `pllogis` for the cumulative distribution (if `d` is given), or `H` for the cumulative hazard (to complement `h`).

These functions must be *vectorised*, and the density function must also accept an argument `log`, which when `TRUE`, returns the log density. See the examples below.

`pars` Character vector naming the parameters of the distribution $\mu, \alpha_1, ..., \alpha_R$. These must match the arguments of the R distribution function or functions.

`location` Character: quoted name of the location parameter μ. The location parameter will not necessarily be the first one, e.g. in `dweibull` the `scale` comes after the `shape`.

`transforms` A list of functions $g()$ which transform the parameter from its natural range to the real line, for example, `c(log,identity)` [3]

`inv.transforms` List of corresponding inverse functions.

`inits` A function which provides plausible initial values of the parameters for maximum likelihood estimation. This is optional, but if not provided, then each call to `flexsurvreg` must have an `inits` argument containing a vector of initial values, which is inconvenient. Implausible initial values may produce a likelihood of zero, and a fatal error message (`initial value in 'vmmin' is not finite`) from the optimiser.

Each distribution will ideally have a heuristic for initialising parameters from summaries of the data. For example, since the median of the Weibull is $\mu \log(2)^{1/\alpha}$, a sensible estimate of μ will commonly be the median log uncensored survival time divided by $\log(2)$, with $\alpha = 1$, assuming that in practice the true value of α is not often far from 1. Then we define the function, of one argument `t` assumed to be the uncensored survival times, returning the initial values for the Weibull `shape` and `scale` respectively.

```
inits = function(t) c(1, median(t[t>0]) / log(2))
```

More complicated initial value functions may use other data such as the covariate values and censored observations: for an example, see the function `flexsurv.splineinits` in the package source that computes initial values for spline models (§4.1).

Example: Using functions from a contributed package The following custom model uses the log-logistic distribution functions (`dllogis` and `pllogis`) available in the package **eha**. The survivor function is $S(t|\mu, \alpha) = 1/(1 + (t/\mu)^\alpha)$, so that the odds $(1 - S(t))/S(t)$ of having died are a linear function of log time.

```
> library(eha)
> custom.llogis <- list(name="llogis",  pars=c("shape","scale"), location="scale",
+                       transforms=c(log, log), inv.transforms=c(exp, exp),
+                       inits=function(t){ c(1, median(t)) })
> fs4 <- flexsurvreg(Surv(recyrs, censrec) ~ group, data=bc, dist=custom.llogis)
```

This fits the breast cancer data better than the Weibull, since it can represent a peaked hazard, but less well than the generalized gamma (Table 3).

[3] This is a *list*, not an *atomic vector* of functions, so if the distribution only has one parameter, we should write `transforms=c(log)` or `transforms=list(log)`, not `transforms=log`

Example: Wrapping functions from a contributed package Sometimes there may be probability density and similar functions in a contributed package, but in a different format. For example, **eha** also provides a three-parameter Gompertz-Makeham distribution with hazard $h(t|\mu, \alpha_1, \alpha_2) = \alpha_2 + \alpha_1 \exp(t/\mu)$. The shape parameters α_1, α_2 are provided to `dmakeham` as a vector argument of length two. However, `flexsurvreg` expects distribution functions to have one argument for each parameter. Therefore we write our own functions that wrap around the third-party functions.

```
> dmakeham3 <- function(x, shape1, shape2, scale, ...)  {
+     dmakeham(x, shape=c(shape1, shape2), scale=scale, ...)
+ }
> pmakeham3 <- function(q, shape1, shape2, scale, ...)  {
+     pmakeham(q, shape=c(shape1, shape2), scale=scale, ...)
+ }
```

`flexsurvreg` also requires these functions to be *vectorized*, as the standard distribution functions in R are. That is, we can supply a vector of alternative values for one or more arguments, and expect a vector of the same length to be returned. The R base function `Vectorize` can be used to do this here.

```
> dmakeham3 <- Vectorize(dmakeham3)
> pmakeham3 <- Vectorize(pmakeham3)
```

and this allows us to write, for example,

```
> pmakeham3(c(0, 1, 1, Inf), 1, c(1, 1, 2, 1), 1)

[1] 0.0000000 0.9340120 0.9757244 1.0000000
```

We could then use `dist=list(name="makeham3", pars=c("shape1","shape2","scale"),...)` in a `flexsurvreg` model, though in the breast cancer example, the second shape parameter is poorly identifiable.

Example: Changing the parameterisation of a distribution We may want to fit a Weibull model like `fs1`, but parameterised as $S(t) = \exp(-\mu t^{\alpha})$, so that the covariate effects reported in the printed `flexsurvreg` object can be interpreted as hazard ratios or log hazard ratios without any further transformation. Here instead of the density and cumulative distribution functions, we provide the hazard and cumulative hazard.[4]

```
> detach("package:eha")
> hweibullPH <- function(x, shape, scale = 1, log=FALSE){
+     hweibull(x, shape=shape, scale=scale^{-1/shape}, log=log)
+ }
> HweibullPH <- function(x, shape, scale=1, log=FALSE){
+     Hweibull(x, shape=shape, scale=scale^{-1/shape}, log=log)
```

[4]The **eha** package needs to be detached first so that **flexsurv**'s built-in `hweibull` is used, which returns `NaN` if the parameter values are zero, rather than failing as **eha**'s does.

```
+ }
> custom.weibullPH <- list(name="weibullPH",
+                          pars=c("shape","scale"), location="scale",
+                          transforms=c(log, log), inv.transforms=c(exp, exp),
+                          inits = function(t){
+                              c(1, median(t[t>0]) / log(2))
+                          })
> fs6 <- flexsurvreg(Surv(recyrs, censrec) ~ group, data=bc, dist=custom.weibullPH)
> 1 / fs1$res["scale","est"]^fs1$res["shape","est"]

[1] 0.03472474

> 1 / exp(fs1$res["groupMedium","est"]) ^ fs1$res["shape","est"]

[1] 2.331564
```

The fitted model is the same as `fs1`, therefore the maximised likelihood is the same, and the parameter estimates of `fs1` can be transformed to those of `fs6` as shown.

A slightly more complicated example is given in the package vignette `flexsurv-examples` of constructing a proportional hazards generalized gamma model.

Example: Omitting the cumulative distribution or hazard If there is no analytic form for $F(t)$ or $H(t)$ as the integral of the density or hazard respectively, then **flexsurv** can compute these internally by numerical integration, as in **stgenreg** (Crowther and Lambert 2013). The default options of the built-in R routine `integrate` for adaptive quadrature are used, though these may be changed using the `integ.opts` argument to `flexsurvreg`. Models specified this way will take much longer to fit, by an order of magnitude. An example is given in §4.2.

4. Arbitrary-dimension models

flexsurv also supports models where the number of parameters is arbitrary. In the models discussed previously, the number of parameters in the model family is fixed (e.g. three for the generalized gamma). In this section, the model complexity can be chosen by the user, given the model family. We may want to represent more irregular hazard curves by more flexible functions, or use bigger models if a bigger sample size makes it feasible to estimate more parameters.

4.1. Royston and Parmar spline model

In the spline-based survival model of Royston and Parmar (2002), a transformation $g(S(t, z))$ of the survival function is modelled as a natural cubic spline function of log time, $x = \log(t)$, plus linear effects of covariates z. This is available here as the function `flexsurvspline`, and is also available in the Stata package `stpm2` (Lambert and Royston 2009) (and historically `stpm`, Royston (2001, 2004)).

Model	$g(S(t,\mathbf{z}))$	In `flexsurvspline`	With $m=0$
Proportional hazards	$\log(-\log(S(t,\mathbf{z})))$ (log cumulative hazard)	`scale="hazard"`	Weibull `shape` γ_1, `scale` $\exp(-\gamma_0/\gamma_1)$
Proportional odds	$\log(S(t,\mathbf{z})^{-1}-1)$ (log cumulative odds)	`scale="odds"`	Log-logistic `shape` γ_1, `scale` $\exp(-\gamma_0/\gamma_1)$
Normal / probit	$\Phi^{-1}(S(t,\mathbf{z}))$ (inverse normal CDF, `qnorm`)	`scale="normal"`	Log-normal `meanlog` $-\gamma_0/\gamma_1$, `sdlog` $1/\gamma_1$

Table 2: Alternative modelling scales for `flexsurvspline`, and equivalent distributions for $m=0$ (with parameter definitions as in the R `d` functions referred to elsewhere in the paper)

$$g(S(t,z)) = s(x,\boldsymbol{\gamma})$$

Typically we use $g(S(t,\mathbf{z}) = \log(-\log(S(t,\mathbf{z}))) = \log(H(t,\mathbf{z}))$, the log cumulative hazard, giving a proportional hazards model.

Spline parameterisation The complexity of the model, thus the dimension of $\boldsymbol{\gamma}$, is governed by the number of *knots* m in the spline function $s()$. Natural cubic splines are piecewise cubic polynomials defined to be continuous, with continuous first and second derivatives at the knots, and also constrained to be linear beyond boundary knots k_{min}, k_{max}. As well as the boundary knots there may be up to $m \geq 0$ *internal* knots $k_1, \ldots k_m$. Various spline parameterisations exist — the one used here is from Royston and Parmar (2002).

$$s(x,\boldsymbol{\gamma}) = \gamma_0 + \gamma_1 x + \gamma_2 v_1(x) + \ldots + \gamma_{m+1} v_m(x) \tag{3}$$

where $v_j(x)$ is the jth *basis* function

$$v_j(x) = (x-k_j)_+^3 - \lambda_j (x-k_{min})_+^3 - (1-\lambda_j)(x-k_{max})_+^3, \qquad \lambda_j = \frac{k_{max}-k_j}{k_{max}-k_{min}}$$

and $(x-a)_+ = max(0, x-a)$. If $m=0$ then there are only two parameters γ_0, γ_1. In fact if $g()$ is the log cumulative hazard, this is equivalent to a Weibull model. Table 2 explains two further choices of $g()$, and the parameter values and distributions they simplify to for $m=0$. The probability density and cumulative distribution functions for this model are available as `dsurvspline` and `psurvspline`.

Covariates on spline parameters Covariates can be placed on any parameter γ through a linear model (with identity link function). Most straightforwardly, we can let the intercept γ_0 vary with covariates $\mathbf{z}$, giving a proportional hazards or odds model (depending on $g()$).

$$g(S(t,z)) = s(x,\boldsymbol{\gamma}) + \boldsymbol{\beta}^T \mathbf{z}$$

The spline coefficients $\gamma_j : j = 1, 2 \ldots$, the "ancillary parameters", may also be modelled as linear functions of covariates $\mathbf{z}$, as

$$\gamma_j(\mathbf{z}) = \gamma_{j0} + \gamma_{j1} z_1 + \gamma_{j2} z_2 + \dots$$

giving a model where the effects of covariates are arbitrarily flexible functions of time: a non-proportional hazards or odds model.

Spline models in flexsurv The package provides the function `flexsurvspline` to fit this general model. Internal knots are chosen by default from quantiles of the log uncensored death times, or users can supply their own knot locations in the `knots` argument to `flexsurvspline`. Initial values for numerical likelihood maximisation are chosen using the method described by Royston and Parmar (2002) of Cox regression combined with transforming an empirical survival estimate.

For example, the best-fitting model for the breast cancer dataset identified in Royston and Parmar (2002), a proportional odds model with one internal spline knot, is

```
> sp1 <- flexsurvspline(Surv(recyrs, censrec) ~ group, data=bc, k=1,
+                       scale="odds")
```

A further model where the first ancillary parameter also depends on the prognostic group, giving a time-varying odds ratio, is fitted as

```
> sp2 <- flexsurvspline(Surv(recyrs, censrec) ~ group + gamma1(group),
+                       data=bc, k=1, scale="odds")
```

These models give qualitatively similar results to the generalized gamma in this dataset (Figure **??**), and have similar predictive ability as measured by AIC (Table 3). Though in general, an advantage of spline models is that extra flexibility is available where necessary.

Note that the log hazard ratios under the proportional hazards spline model are practically the same as under a standard Cox model.

```
> sp3 <- flexsurvspline(Surv(recyrs, censrec) ~ group, data=bc, k=1, scale="hazard")
> sp3$res[c("groupMedium","groupPoor"),c("est","se")]
```

```
                 est        se
groupMedium 0.8334517 0.1712042
groupPoor   1.6111788 0.1640933
```

```
> cox3 <- coxph(Surv(recyrs, censrec) ~ group, data=bc)
> coef(summary(  cox3))[,c("coef","se(coef)")]
```

```
                coef  se(coef)
groupMedium 0.8401002 0.1713926
groupPoor   1.6180720 0.1645443
```

```
> res <- t(sapply(list(fs1, fs2, fs3, fs4, sp1, sp2),
+                 function(x)rbind(-2*x$loglik, x$npars, x$AIC)))
> rownames(res) <- c("Weibull","Generalized gamma",
+                    "Generalized gamma (time-varying effects)","Log-logistic",
+                    "Spline (sp1)", "Spline (sp2: time-varying effects)")
> colnames(res) <- c("-2 log likelihood","Parameters","AIC")


> res

                                       -2 log likelihood Parameters      AIC
Weibull                                         1623.884          4 1631.884
Generalized gamma                               1575.137          5 1585.137
Generalized gamma (time-varying effects)        1572.434          7 1586.434
Log-logistic                                    1598.105          4 1606.105
Spline (sp1)                                    1577.964          5 1587.964
Spline (sp2: time-varying effects)              1574.848          7 1588.848
```

Table 3: Comparison of parametric survival models fitted to the breast cancer data

4.2. Implementing general-dimension models

The spline model above is an example of the general parametric form (1), but the number of parameters, $R+1$ in (1), $m+2$ in (3), is arbitrary. **flexsurv** has the tools to deal with any model of this form. `flexsurvspline` works internally by building a custom distribution and then calling `flexsurvreg`. Similar models may in principle be built by users using the same method. This relies on a functional programming trick.

Creating distribution functions dynamically The R distribution functions supplied to custom models are expected to have a fixed number of arguments, including one for each scalar parameter. However, the distribution functions for the spline model (e.g. `dsurvspline`) have an argument `gamma` representing the vector of parameters γ, whose length is determined by the user through the choice of the number of knots. Just as the *scalar parameters* of conventional distribution functions can be supplied as *vector arguments* (as explained in §3.5), similarly, the vector parameters of spline-like distribution functions can be supplied as *matrix arguments*, representing alternative parameter values.

To convert a spline-like distribution function into the correct form, **flexsurv** provides the utility `unroll.function`. This converts a function with one (or more) vector parameters (matrix arguments) to a function with an arbitrary number of scalar parameters (vector arguments). For example, the 5-year survival probability for the baseline group under the model `sp1` is

```
> gamma <- sp1$res[c("gamma0","gamma1","gamma2"),"est"]
> 1 - psurvspline(5, gamma=gamma, knots=sp1$knots)

[1] 0.6896969
```

An alternative function to compute this can be built by `unroll.function`. We tell it that the vector parameter `gamma` should be provided instead as three scalar parameters named `gamma0`,`gamma1`,`gamma2`. The resulting function `pfn` is in the correct form for a custom `flexsurvreg` distribution.

```
> pfn <- unroll.function(psurvspline, gamma=0:2)
> 1 - pfn(5, gamma0=gamma[1], gamma1=gamma[2], gamma2=gamma[3], knots=sp1$knots)

[1] 0.6896969
```

Users wishing to fit a new spline-like model with a known number of parameters could just as easily write distribution functions specific to that number of parameters, and use the methods in §3.5. However the `unroll.function` method is intended to simplify the process of extending the **flexsurv** package to new classes of models. The intention is that wrappers similar to `flexsurvspline` could be included in the package in the future for useful classes of models.

Example: splines on alternative scales An alternative to the Royston-Parmar spline model is to model the log *hazard* as a function of time instead of the log cumulative hazard. An advantage explained by Royston and Lambert (2011) is that when there are when there are multiple time-dependent effects, time-dependent hazard ratios can be interpreted independently of the values of other covariates. Crowther and Lambert (2013) demonstrate this using the Stata `stgenreg` package.

This can also be implemented in `flexsurvreg` using `unroll.function`. A disadvantage of this model is that the cumulative hazard (hence the survivor function) has no analytic form, therefore to compute the likelihood, the hazard function needs to be integrated numerically. This is done automatically in `flexsurvreg` (just as in `stgenreg`) if the cumulative hazard is not supplied.

Firstly, a function must be written to compute the hazard as a function of time `x`, the vector of parameters `gamma` (which can be supplied as a matrix argument so the function can give a vector of results), and a vector of knot locations.

```
> hsurvspline.lh <- function(x, gamma, knots){
+     if(!is.matrix(gamma)) gamma <- matrix(gamma, nrow=1)
+     lg <- nrow(gamma)
+     nret <- max(length(x), lg)
+     gamma <- apply(gamma, 2, function(x)rep(x,length=nret))
+     x <- rep(x, length=nret)
+     loghaz <- rowSums(basis(knots, log(x)) * gamma)
+     exp(loghaz)
+ }
```

The equivalent function is then created for a three-knot example of this model (one internal and two boundary knots) that has arguments `gamma0`,`gamma1` and `gamma2` corresponding to the three columns of `gamma`,

```
> hsurvspline.lh3 <- unroll.function(hsurvspline.lh, gamma=0:2)
```

A custom distribution list, referring to this function, is then constructed to supply to `flexsurvreg`.

```
> custom.hsurvspline.lh3 <- list(
+     name = "survspline.lh3",
+     pars = c(paste0("gamma",0:2)),
+     location = c("gamma0"),
+     transforms = rep(c(identity), 3), inv.transforms=rep(c(identity), 3)
+     )
```

To complete the model, we place the internal knot at the median uncensored log survival time, and boundary knots at the minimum and maximum (the default knots used by `flexsurvspline`). These are passed to `hsurvspline.lh` through the `aux` argument of `flexsurvreg`.

```
> dtime <- log(bc$recyrs)[bc$censrec==1]
> ak <- list(knots=quantile(dtime, c(0, 0.5, 1)))
```

Initial values must be provided in the call to `flexsurvreg`, since the custom distribution list did not include an `inits` component. For this example, "default" initial values of zero suffice, but the permitted values of γ_2 are fairly tightly constrained (from -0.5 to 0.5 here) using the `"L-BFGS-B"` bounded optimiser from R's `optim` (Nash 1990). Without the constraint, extreme values of γ_2, visited by the optimiser, cause the numerical integration of the hazard function to fail.

```
> fs3 <- flexsurvreg(Surv(recyrs, censrec) ~ group, data=bc, aux=ak,
+                    inits=c(0, 1, 0.3, 0, 0), dist=custom.hsurvspline.lh3,
+                    method="L-BFGS-B", lower=c(-Inf,-Inf,-0.5), upper=c(Inf,Inf,0.5),
+                    control=list(trace=1,REPORT=1))
```

This takes around ten minutes to converge, so is not evaluated here, though the fit is poorer than the equivalent spline model for the cumulative hazard. The 95% confidence interval for γ_2 of (0.16, 0.37) is firmly within the constraint.

Other arbitrary-dimension models Another potential application is to fractional polynomials (Royston and Altman 1994). These are of the form $\sum_{m=1}^{M} \alpha_m x^{p_m} log(x)^n$ where the power p_m is in the standard set $\{2, -1, -0.5, 0, 0.5, 1, 2, 3\}$ (except that $log(x)$ is used instead of x^0), and n is a non-negative integer. They are similar to splines in that they can give arbitrarily close approximations to a nonlinear function, such as a hazard curve, and are particularly useful for building interpretable models for continuous predictors in regression models. See e.g. Sauerbrei, Royston, and Binder (2007), and several other publications by the same authors, for applications and discussion of their advantages over splines. The R package **gamlss** (Rigby and Stasinopoulos 2005) has a function to construct a fractional polynomial basis that might be employed in **flexsurv** models.

Polyhazard models (Louzada-Neto 1999) are another potential use of this technique. These express an overall hazard as a sum of latent cause-specific hazards, each one typically from the same class of distribution, e.g. a *poly-Weibull* model if they are all Weibull. For example, a U-shaped hazard curve following surgery may be the sum of early hazards from surgical mortality and later deaths from natural causes. However, such models may not always be

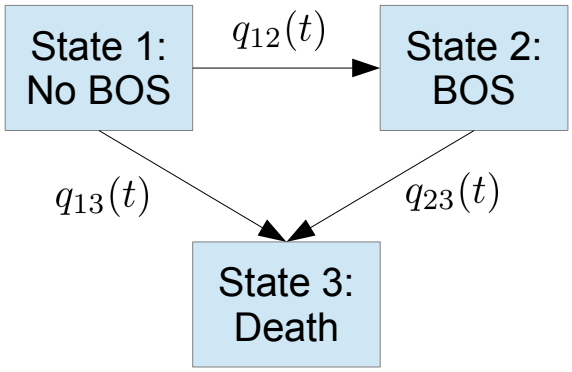

Figure 3: Three-state multi-state model for bronchiolitis obliterans syndrome (BOS)

identifiable without external information to fix or constrain the parameters of particular hazards (Demiris, Lunn, and Sharples 2011).

5. Multi-state models

A *multi-state model* represents how an individual moves between multiple states through time. Survival analysis is a special case with two states, "alive" and "dead". *Competing risks* are a further special case, where there are multiple causes of death, that is, multiple possible destination states for the same starting state.

Instead of a single event time, there may now be a series of event times $t_1, \ldots, t_n$ for an individual. The last of these may be an observed or right-censored event time. Given that an individual is in state $S(t)$ at time t, their next state, and the time of the change, are governed by a set of *transition intensities*

$$q_{rs}(t, \mathbf{z}(t), \mathcal{F}_t) = \lim_{\delta t \to 0} P(S(t + \delta t) = s | S(t) = r, \mathbf{z}(t), \mathcal{F}_t,)/\delta t$$

for states $r, s = 1, \ldots, R$, which for a survival model are equivalent to the hazard $h(t)$. The intensity represents the instantaneous risk of moving from state r to state s. It may depend on the time t since the start of the process, patient characteristics $\mathbf{z}(t)$, and possibly also the "history" of the process up to that time, $\mathcal{H}_t$, the states previously visited or the length of time spent in them.

For illustration, consider a very simple three-state example, previously studied by Heng, Sharples, McNeil, Stewart, Wreghitt, and Wallwork (1998). Recipients of lung transplants are are risk of bronchiolitis obliterans syndrome (BOS). This was defined as a decrease in lung function to below 80% of a baseline value defined in the six months following transplant. A three-state "illness-death" model represents the risk of developing BOS, the risk of dying before developing BOS, and the risk of death after BOS. BOS is assumed to be irreversible, so there are only three allowed transitions (Figure 3), each with an intensity function $q_{rs}(t)$.

Alternative time scales In semi-Markov (clock-reset) models, $q_{rs}(t)$ depends on the time t since entry into the current state, but otherwise, the time since the beginning of the process is forgotten. Any software to fit survival models can also fit this kind of multi-state model.

In an inhomogeneous Markov (clock-forward) model, t represents the time since the beginning of the process, but the intensity $q_{rs}(t)$ does not depend further on $\mathcal{H}_t$. Again any survival modelling software can be used, with the additional requirement that it can deal with left-truncation or *counting process* data.

Competing risks models can be fitted as semi-Markov models, since there is at most one transition for each individual, so that the time since the beginning of the process equals the time spent in the current state, therefore no left-truncation is necessary.

5.1. Representing multi-state data as survival data

Putter, Fiocco, and Geskus (2007) discuss how to implement multi-state models using standard survival modelling software — an overview is given here. For each permitted $r \to s$ transition in the multi-state model, there is a corresponding "survival" (time-to-event) model, with hazard rates defined by $q_{rs}(t)$. For a patient who moves into state r at time t_j, their next event at t_{j+1} is defined by the model structure (FIGURE) to be one of a set of competing events $s_1, \ldots, s_{n_r}$. This implies there are n_r corresponding survival models for this state r, and $\sum_r n_r$ time-to-event models over all states r. In the BOS example, there are $n_1 = 2, n_2 = 1$ and $n_3 = 0$ possible transitions from states 1, 2 and 3 respectively.

The data to inform the n_r models from state r consists of an indicator for whether the transition to the corresponding state $s_1, \ldots, s_{n_r}$ is observed or censored at t_{j+1}, coupled with:

- (for a semi-Markov model) the time elapsed $dt_j = t_{j+1} - t_j$ from state r entry to state s entry. This data informs the "survival" model for the $r \to s$ transition.
- (for an inhomogeneous Markov model) the start and stop time (t_j, t_{j+1}). The $r \to s$ model is fitted to the right-censored time t_{j+1} from the *start of the process*, but is conditional on not experiencing the $r \to$ transition until after the state r entry time. In other words, the $r \to s$ transition model is *left-truncated* at the state r entry time.

The **mstate** R package (de˜Wreede, Fiocco, and Putter 2010, 2011) has a utility `msprep` to produce data of this form from "wide-format" datasets where rows represent individuals, and times of different events appear in different columns, and **msm** (Jackson 2011) has a similar utility `msm2Surv` for producing the required form given longitudinal data where rows represent state observations.

The outcomes of two patients in the BOS data are given by

```
> bosms3[18:22,]

An object of class 'msdata'

Data:
   id from to    Tstart     Tstop time status trans     years
18  7    1  2 0.0000000 0.1697467   62      1     1 0.1697467
19  7    1  3 0.0000000 0.1697467   62      0     2 0.1697467
20  7    2  3 0.1697467 0.6297057  168      1     3 0.4599589
21  8    1  2 0.0000000 8.1615332 2981      0     1 8.1615332
22  8    1  3 0.0000000 8.1615332 2981      1     2 8.1615332
```

Each row represents an observed (`status=1`) or censored (`status=0`) transition time for one of three time-to-event models indicated by the categorical variable `trans` (defined as a factor). Times are expressed in days, with time 0 representing six months after transplant. Values of `trans` of 1, 2, 3 correspond to no BOS→BOS, no BOS→death and BOS→death respectively. The first row indicates that the patient (`id` 7) moved from state 1 (no BOS) to state 2 (BOS) at 62 days, but (second row) that this is also interpreted as a censored time from state 1 to state 3, potential death before BOS onset. This patient then died, given bu the third row with `status` 1 for `trans` 3. Patient 8 died before BOS onset, therefore at day 2981 their potential BOS onset is censored (fourth row), but their death before BOS is observed (fifth row).

5.2. Fitting parametric multi-state models

Two semi-Markov multi-state models are fitted using `flexsurvreg`. The first is a simple time-homogeneous Markov model where all transition intensities are constant through time, equivalent to exponential times to the next event, but with a different intensity q_{rs} for each transition type `trans`. In the second, the time to the next transition has a Weibull distribution, with a different shape and scale for each transition type.

```
> bosms3$years <- bosms3$time / 365.25
> bexp <- flexsurvreg(Surv(years, status) ~ trans, data=bosms3, dist="exp")
> bwei <- flexsurvreg(Surv(years, status) ~ trans + shape(trans), data=bosms3,
+                     dist="weibull")
```

The equivalent semi-parametric Cox model is also fitted using `coxph` from the **survival** package. This specifies a different baseline hazard for each transition type through a function `strata` in the formula, so since there are no other covariates, it is essentially non-parametric. Note that the `strata` function is not currently understood by `flexsurvreg` — the user must say explicitly what parameters, if any, vary with the transition type, as in `bwei`.

```
> bcox <- coxph(Surv(years, status) ~ strata(trans), data=bosms3)
```

In all cases, if there were other covariates, they could simply be included in the model formula. Typically covariate effects will vary with the transition type, so that an interaction term with `trans` would be included, and some post-processing would be needed to combine the main effects and interaction terms into an easily-interpretable effect (such as the hazard ratio for the r, s transition). Alternatively, **mstate** has a utility `expand.covs` to add transition-specific covariates to the data.

5.3. Obtaining cumulative transition-specific hazards

The `mstate` package enables *semi-parametric* multi-state modelling. Like the example `bcox`, models must be fitted with `coxph`, which also provides a semi-parametric estimates of each transition-specific baseline hazard (de˜Wreede *et˜al.* 2010, 2011). These imply piecewise-constant estimates of each cumulative $r \to s$ transition-specific hazard function $H_{rs}(t) = \int_0^t q_{rs}(u)du$. These estimates, and their covariances, are provided by **mstate**'s function `msfit`, and form the basis of prediction from the model. In this case,

```
> library(mstate)
> tmat <- rbind(c(NA,1,2),c(NA,NA,3),c(NA,NA,NA))
> mcox <- msfit(bcox, trans=tmat)
```

`tmat` describes the transition structure. This is a matrix of integers whose r, s entry is i if the ith transition type is r, s, and has `NA`s on the diagonal and where the r, s transition is disallowed.

flexsurv provides an analogous function `msfit.flexsurvreg` to produce cumulative hazards from *fully-parametric* multi-state models in the same format. This is simply a wrapper around `summary.flexsurvreg(...,type="cumhaz")`, as previously mentioned in §3.2. The difference from **mstate**'s method is that hazard estimates can be produced for any grid of times `t`, at any level of detail and even beyond the range of the data, since the model is fully parametric. The argument `newdata` can be used in the same way to specify a desired covariate category, though in this example there are no covariates in addition to the transition type. The name of the (factor) covariate indicating the transition type can also be supplied through the `tvar` argument, in this case it is the default, `"trans"`.

```
> mexp <- msfit.flexsurvreg(bexp, t=seq(0,150,1), trans=tmat)
> mwei <- msfit.flexsurvreg(bwei, t=seq(0,150,1), trans=tmat)
```

These can be plotted (Figure 4) to show the fit of the parametric models compared to the non-parametric estimates. Both models appear to fit adequately, though give diverging extrapolations after around 6 years when the data become sparse. The Weibull model has an improved AIC of 1091.11644350468, compared to 1099.29192817275 for the exponential model.

5.4. Prediction from multi-state models

Since `msfit.flexsurvreg` returns transition hazards in the same format as **mstate**'s `msfit` method, the functions of **mstate** for prediction can then be used. For example, the *transition probabilities* of the multi-state model are the probabilities of occupying each state s at time $t > t_0$, given that the individual is in state r at time t_0.

$$P(t_0, t) = P(S(t) = s | S(t_0) = r)$$

For an inhomogeneous Markov model, these are related to the transition intensities via the Kolmogorov forward equation

$$\frac{dP(t_0, t)}{dt} = P(t_0, t)Q(t)$$

with initial condition $P() = I$ (Cox and Miller 1965). An approximate solution (e.g. Aalen, Borgan, and Gjessing 2008) is given by a product integral

$$P(t_0, t) = \prod_{i=0}^{m-1} \{I + Q(t_i)du\}$$

where a fine grid of times $t_0, t_1, \ldots, t_m = t$ is chosen to span the prediction interval, and $Q(t_i)du$ is the increment in the cumulative hazard matrix between times t_i and t_{i+1}. Q may also depend on other covariates, as long as these are known in advance.

```
> plot(mcox, xlab="Years after baseline", lwd=3, xlim=c(0,14))
> cols <- c("black","red","green")
> for (i in 1:3){
+     lines(mexp$Haz$time[mexp$Haz$trans==i], mexp$Haz$Haz[mexp$Haz$trans==i],
+           type="s", col=cols[i], lty=2, lwd=2)
+     lines(mwei$Haz$time[mwei$Haz$trans==i], mwei$Haz$Haz[mwei$Haz$trans==i],
+           type="s", col=cols[i], lty=3, lwd=2)
+ }
> legend("topleft", inset=c(0,0.2), c("Non-parametric","Exponential","Weibull"),
+        lty=c(1,2,3), lwd=c(3,2,2), bty="n")
```

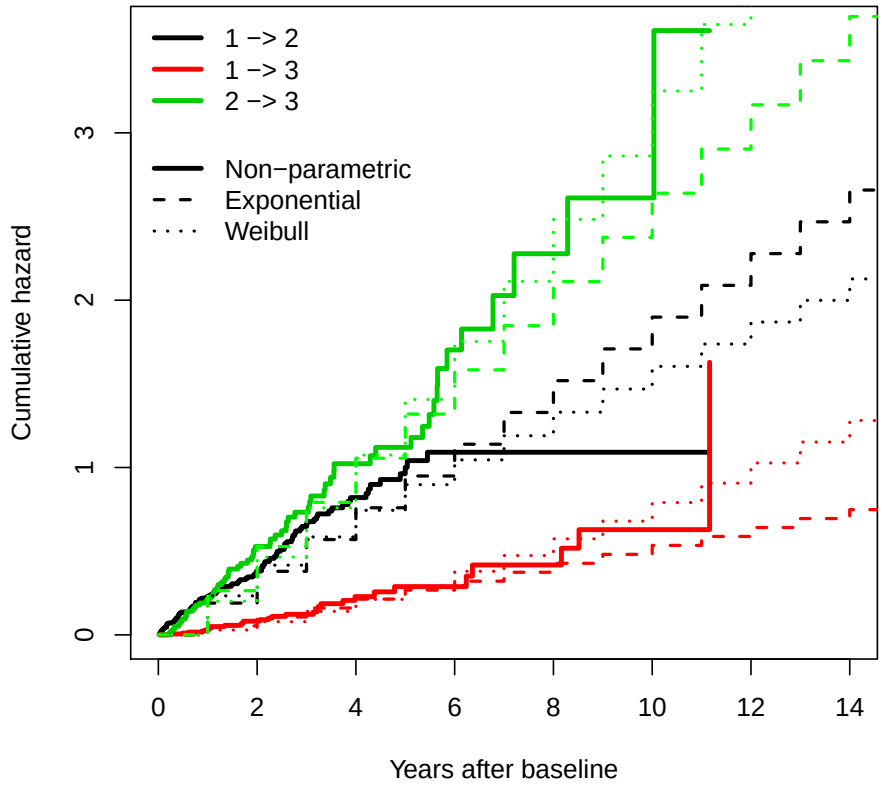

Figure 4: Cumulative hazards for three transitions in the BOS multi-state model, under a non-parametric and two parametric models.

In **mstate** these can be calculated with the `probtrans` function, applied to the cumulative hazards returned by `msfit`. For the semi-parametric models used in **mstate**, the time grid is naturally defined by the observed survival times, giving the Aalen-Johansen estimator Andersen, Borgan, Gill, and Keiding (1993). For parametric models, using a similar discrete-time approximation was suggested by Cook and Lawless (2014), and can be made arbitrarily accurate by choosing a finer resolution for the grid of times. This is achieved by passing the object returned by `msfit.flexsurvreg` to `probtrans` in **mstate**.

An alternative approach, not yet investigated in **flexsurv**, would be to solve the Kolmogorov forward equation numerically, the approach taken by Titman (2011).

For semi-Markov models, these equations do not apply, since the transition intensity matrix $Q(t)$ is not known in advance at time t, but depends on when the transitions occur between time t_0 and t. $P(t_0, t)$ can then be estimated by simulation.

EXAMPLE OF ONE OR MORE OF THESE

5.5. Multi-state models for panel data

Note the contrast between the models discussed in this section, and multi-state models for *panel data*, that is, observations of the state of the process at a series of times (Kalbfleisch and Lawless 1985). In panel data, we do not necessarily know know the time of each transition, or even whether transitions of a certain type have occurred at all between a pair of observations. Multi-state models for this type of data (and also for the exact event time data discussed above) can be fitted with the **msm** package for R (Jackson 2011), but are restricted to (piecewise) exponentially-distributed event times.

6. Potential extensions

Models (Hougaard 1995) where multiple survival times are assumed to be correlated within groups, sometimes called (shared) frailty models, would be a useful extension. See, e.g. Crowther, Look, and Riley (2014) for a recent application based on parametric survival models. These might be implemented by exploiting tractability for specific distributions, such as gamma frailties, or by adjusting standard errors to account for clustering, as implemented in `survreg`. More complex random effects models would require numerical integration, and (Crowther *et˜al.* 2014) provide Stata software based on Gauss-Hermite quadrature. Alternatively, a probabilistic modelling language such as Stan (Stan Development Team 2014) or BUGS (Lunn, Jackson, Best, Thomas, and Spiegelhalter 2012) would be naturally suited to complex extensions such as random effects on multiple parameters or multiple hierarchical levels.

Relative survival models (Nelson, Lambert, Squire, and Jones 2007), often used in cancer epidemiology, are a typical application of the Stata packages mentioned in this article, but have not yet been attempted with **flexsurv**. This wouldn't be conceptually difficult, requiring a known offset to be added to the hazard function, representing the hazard for a reference population.

flexsurv is intended as a platform for parametric survival modelling. Extensions of the software to deal with different models may be written by users themselves, through the facilities described in §3.5 and §4.2. These might then be included in the package as built-in distri-

butions, or at least demonstrated in the package's other vignette `flexsurv-examples`. Each new class of models would ideally come with

- guidance on what situations the model is useful for, e.g. what shape of hazards it can represent
- some intuitive interpretation of the model parameters, their plausible values in typical situations, and potential identifiability problems. This would also help with choosing initial values for numerical maximum likelihood estimation, ideally through an `inits` function in the custom distribution list (§3.5).

A. Acknowledgements

Thanks to Milan Bouchet-Valat for help with implementing covariates on ancillary parameters, Andrea Manca for motivating the development of the package, and all users of the package who have reported bugs and given suggestions.

References

Aalen O, Borgan O, Gjessing H (2008). *Survival and event history analysis: a process point of view*. Springer.

Andersen PK, Borgan O, Gill RD, Keiding N (1993). *Statistical models based on counting processes*. Springer, New York.

Broström G (2014). *eha: Event History Analysis*. R package version 2.4-1, URL http://CRAN.R-project.org/package=eha.

Cook RJ, Lawless JF (2014). "Statistical issues in modeling chronic disease in cohort studies." *Statistics in Biosciences*, **6**(1), 127–161.

Cox C (2008). "The generalized F distribution: An umbrella for parametric survival analysis." *Statistics in Medicine*, **27**, 4301–4312.

Cox DR, Miller HD (1965). *The Theory of Stochastic Processes*. Chapman and Hall, London.

Crowther MJ, Lambert PC (2013). "stgenreg: A Stata package for general parametric survival analysis." *Journal of Statistical Software*, **53**, 1–17.

Crowther MJ, Look MP, Riley RD (2014). "Multilevel mixed effects parametric survival models using adaptive Gauss–Hermite quadrature with application to recurrent events and individual participant data meta-analysis." *Statistics In Medicine*.

de˜Wreede L, Fiocco M, Putter H (2010). "The `mstate` package for estimation and prediction in non-and semi-parametric multi-state and competing risks models." *Computer Methods and Programs in Biomedicine*, **99**(3), 261–274.

de˜Wreede LC, Fiocco M, Putter H (2011). "mstate: an R package for the analysis of competing risks and multi-state models." *J Stat Softw*, **38**, 1–30.

Demiris N, Lunn D, Sharples L (2011). "Survival extrapolation using the poly-Weibull model." *Statistical Methods in Medical Research*.

Heng D, Sharples L, McNeil K, Stewart S, Wreghitt T, Wallwork J (1998). "Bronchiolitis Obliterans Syndrome: Incidence, Natural History, Prognosis, and Risk Factors." *The Journal of Heart and Lung Transplantation*, **17**(12), 1255–1263.

Hess K, Gentleman R (2010). *muhaz: Hazard Function Estimation in Survival Analysis*. R package version 1.2.5, URL http://CRAN.R-project.org/package=muhaz.

Hougaard P (1995). "Frailty models for survival data." *Lifetime Data Analysis*, **1**(3), 255–273.

Jackson CH (2011). "Multi-State Models for Panel Data: The msm Package for R." *Journal of Statistical Software*, **38**(8).

Kalbfleisch J, Lawless J (1985). "The Analysis of Panel Data under a Markov Assumption." *Journal of the American Statistical Association*, **80**(392), 863–871.

Lambert PC, Royston P (2009). "Further development of flexible parametric models for survival analysis." *Stata Journal*, **9**(2), 265.

Latimer NR (2013). "Survival analysis for economic evaluations alongside clinical trials — extrapolation with patient-level data inconsistencies, limitations, and a practical guide." *Medical Decision Making*, **33**(6), 743–754.

Louzada-Neto F (1999). "Polyhazard models for lifetime data." *Biometrics*, **55**, 1281–1285.

Lunn D, Jackson C, Best N, Thomas A, Spiegelhalter D (2012). *The BUGS Book: a practical introduction to Bayesian analysis*. CRC Press.

Mueller HG, Wang JL (1994). "Hazard rates estimation under random censoring with varying kernels and bandwidths." *Biometrics*, **50**, 61–76.

Nadarajah S, Bakar SAA (2013). "A new R package for actuarial survival models." *Computational Statistics*, **28**(5), 2139–2160.

Nash JC (1990). *Compact numerical methods for computers: linear algebra and function minimisation*. CRC Press.

Nelson CP, Lambert PC, Squire IB, Jones DR (2007). "Flexible parametric models for relative survival, with application in coronary heart disease." *Statistics in medicine*, **26**(30), 5486–5498.

Prentice RL (1974). "A log gamma model and its maximum likelihood estimation." *Biometrika*, **61**(3), 539–544.

Prentice RL (1975). "Discrimination among some parametric models." *Biometrika*, **62**(3), 607–614.

Putter H, Fiocco M, Geskus RB (2007). "Tutorial in biostatistics: competing risks and multi-state models." *Statistics in Medicine*, **26**, 2389–2430.

Reid N (1994). "A conversation with Sir David Cox." *Statistical Science*, pp. 439–455.

Rigby RA, Stasinopoulos DM (2005). "Generalized additive models for location, scale and shape,(with discussion)." *Applied Statistics*, **54**, 507–554.

Royston P (2001). "Flexible parametric alternatives to the Cox model, and more." *Stata Journal*, **1**(1), 1–28.

Royston P (2004). "Flexible parametric alternatives to the Cox model: update." *The Stata Journal*, **4**(1), 98–101.

Royston P, Altman DG (1994). "Regression using fractional polynomials of continuous covariates: parsimonious parametric modelling." *Applied Statistics*, pp. 429–467.

Royston P, Lambert PC (2011). "Flexible parametric survival analysis using Stata: beyond the Cox model." *Stata Press books*.

Royston P, Parmar M (2002). "Flexible parametric proportional-hazards and proportional-odds models for censored survival data, with application to prognostic modelling and estimation of treatment effects." *Statistics in Medicine*, **21**(1), 2175–2197.

Sauerbrei W, Royston P (1999). "Building multivariable prognostic and diagnostic models: transformation of the predictors by using fractional polynomials." *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, **162**(1), 71–94.

Sauerbrei W, Royston P, Binder H (2007). "Selection of important variables and determination of functional form for continuous predictors in multivariable model building." *Statistics in medicine*, **26**(30), 5512–5528.

Stacy EW (1962). "A generalization of the gamma distribution." *Annals of Mathematical Statistics*, (33), 1187–92.

Stan Development Team (2014). *Stan Modeling Language Users Guide and Reference Manual, Version 2.4*. URL http://mc-stan.org/.

Therneau T (2014). "A Package for Survival Analysis in S." R package version 2.37-7. http://CRAN.R-project.org/package=survival.

Titman AC (2011). "Flexible nonhomogeneous Markov models for panel observed data." *Biometrics*, **67**(3), 780–787.

Yee TW, Wild CJ (1996). "Vector generalized additive models." *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 481–493.