Simulating simple and complex survival data

Stata UK User Group Meeting
Cass Business School
11th September 2014

Michael J. Crowther

University of Leicester

Department of Health Sciences
University of Leicester, UK
michael.crowther@le.ac.uk

NHS National Institute for Health Research

Outline

1. Background
2. Motivating dataset
3. Simulating survival times from standard distributions
4. A general algorithm for generating survival times
5. Discussion

Background

- Simulation studies are conducted to assess the performance of current and novel statistical models in pre-defined scenarios
- Guidelines for the reporting of simulation studies in medical research have been published (Burton et al., 2006)
- Many simulation studies involving survival data use the exponential or Weibull models
- Often in clinical trials and population based studies, at least one turning point in the baseline hazard function is observed

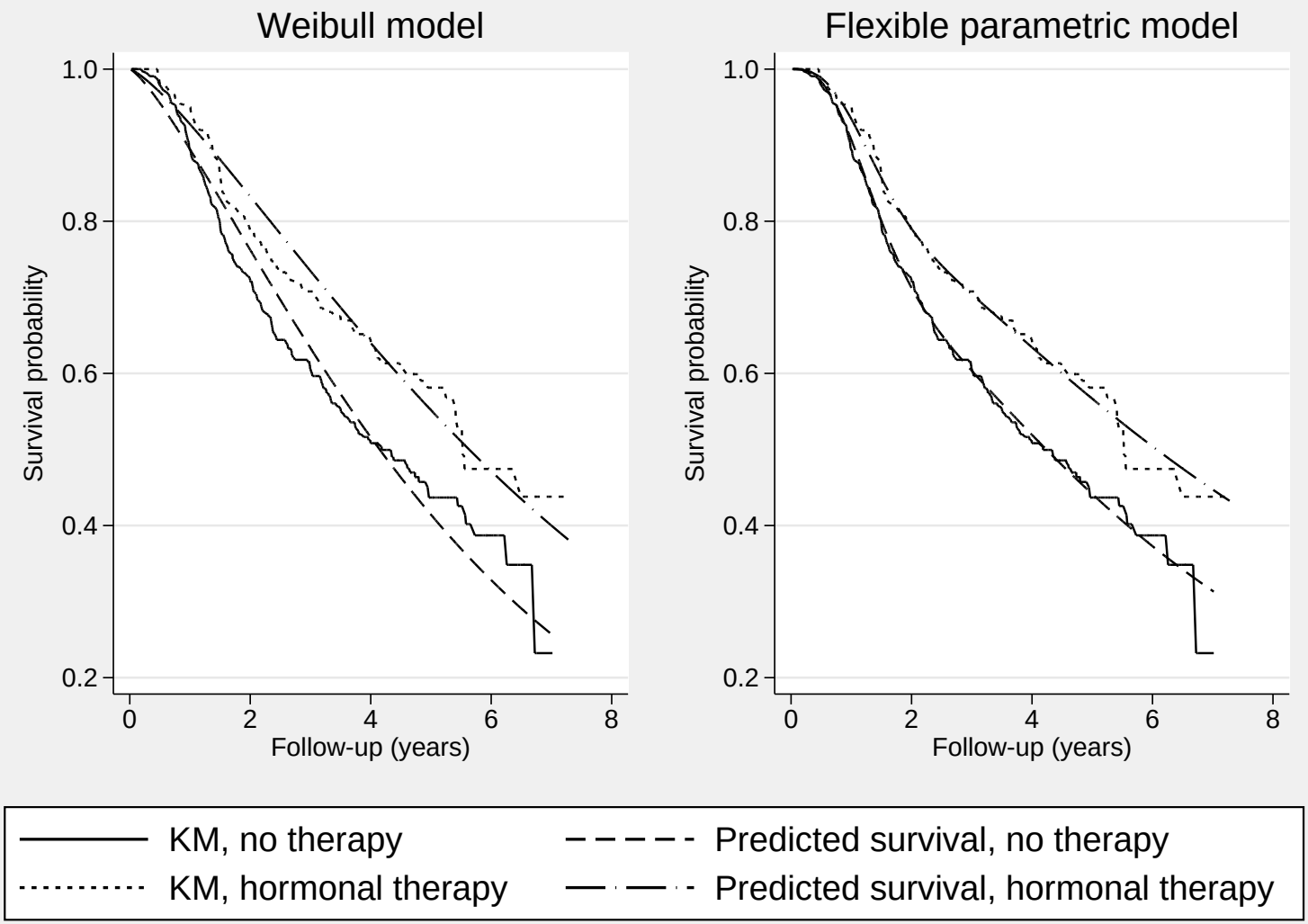
Motivating dataset

- `webuse brcancer`
- 686 women diagnosed with breast cancer in Germany
- 246 were randomised to receive hormonal therapy and 440 to receive a placebo
- Outcome of interest is recurrence-free survival, with 299 patients experiencing the event

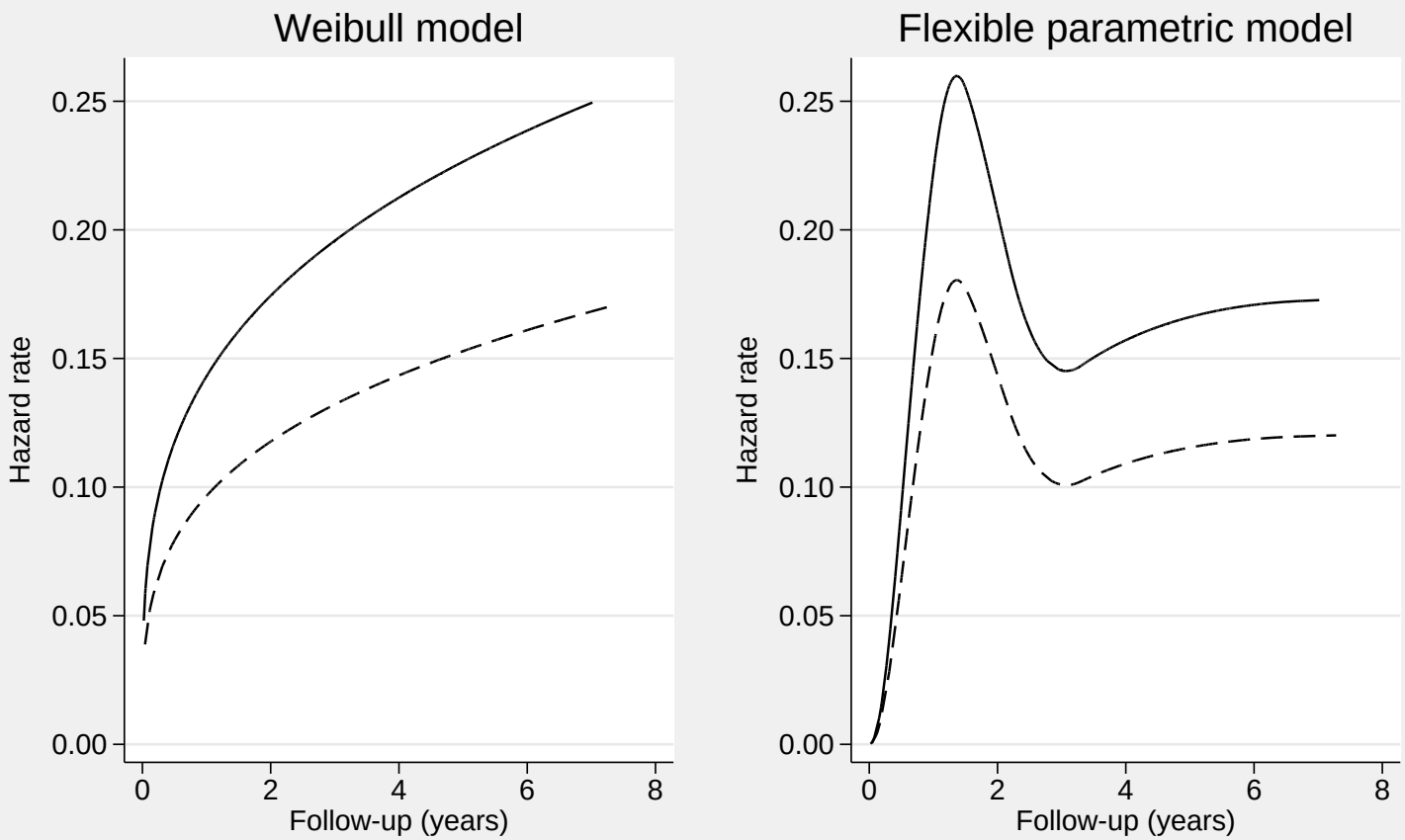
Analysis

- Weibull proportional hazards model
- Flexible parametric model with 5 degrees of freedom
- Treatment included in both models

Fitted survival functions



Fitted hazard functions



Simulating survival times

Bender et al. (2005) provided a simple and efficient method to simulate survival times from standard parametric distributions

$$h(t|X) = h_0(t)\exp(X\beta), \qquad H(t|X) = H_0(t)\exp(X\beta)$$

$$S(t|X) = \exp[-H(t|X)], \qquad F(t|X) = 1 - \exp[-H(t|X)]$$

If we let T be the simulated survival time

$$F(T|X) = 1 - \exp[-H(T|X)] = u, \qquad \text{where} \quad u \sim U(0,1)$$

and

$$S(T|X) = 1 - u \qquad (\text{or equivalently} \quad = u)$$

This can then simply be re-arranged and solved for T

$$T = H_0^{-1}[-\log(u)\exp(-X\beta)]$$

For example in Stata

```
. //simulate 1000 survival times
. set obs 1000
obs was 0, now 1000
. //set seed for reproducibility
. set seed 398894
. //get uniform draws, representing centiles
. gen u = runiform()
. //geenrated a binary treatment group indicator
. gen treatment = runiform()>0.5
. //Weibull baseline parameters
. local lambda = 0.1
. local gamma = 1.2
. //treatment effect
. local loghr = 0.7
. //simulate survival times from Weibull PH model
. gen stimes = (-log(u)/(`lambda´*exp(`loghr´*treatment)))^(1/`gamma´)
```

survsim (from SSC)

survsim *newvarname1* [*newvarname2*] [, *options*]

- distribution(exp|gomp|weib)
- lambda(#), gamma(#)
- covariates(*varname* # [*varname* #] ...)
- tde(*varname* # [*varname* #] ...)
- maxtime(#)

```
.  survsim stime event, dist(weib) lambda(0.1)
> gamma(1.2) cov(treatment 0.7)
```

Recent use of survival simulation

- Paul Lambert and I recently proposed a general parametric framework for survival analysis, implemented in `stgenreg` (Crowther and Lambert, 2013b, 2014)
- Reviews raised questions about benefits/pitfalls compared to the Cox model
- We set out to compare the efficiency of the Kaplan-Meier estimate of survival with a parametric function using splines, when data is sparse in the right tail

Core of simulation program

```
. //simulate from a Weibull distribution
. survsim stime died, lambda(0.2) gamma(1.3) maxt(5)
. //censoring times
. gen cens = runiform()*6
. replace died = 0 if cens<stime
. replace stime = cens if cens<stime
. stset stime, f(died=1)
. //KM estimate
. sts gen s1 = s sells = se(lls) lb = lb(s) ub = ub(s)
. //Fit parametric model
. stgenreg, loghaz([xb]) xb(#rcs(df(3)))
. //Get predicted survival at 4 and 5 years
. range t45 4 5 2
. predict surv, survival timevar(t45) ci
```

Results

Table : Bias and mean squared error of $\log(-\log(S(t)))$ at 4 and 5 years.

Time		Kaplan-Meier	Parametric model
4 years	Bias	-0.0019	-0.0038
	MSE	0.1251	0.1100
5 years	Bias	0.0066	0.0063
	MSE	0.1565	0.1481

Median # events = 101

Median # events in final year = 5

Results

Table : Bias and mean squared error of $\log(-\log(S(t)))$ at 4 and 5 years.

Time		Kaplan-Meier	Parametric model
4 years	Bias	-0.0019	-0.0038
	MSE	0.1251	0.1100
5 years	Bias	0.0066	0.0063
	MSE	0.1565	0.1481

Median # events = 101

Median # events in final year = 5

Results

Table : Bias and mean squared error of $\log(-\log(S(t)))$ at 4 and 5 years.

Time		Kaplan-Meier	Parametric model
4 years	Bias	-0.0019	-0.0038
	MSE	0.1251	0.1100
5 years	Bias	0.0066	0.0063
	MSE	0.1565	0.1481

Median # events = 101

Median # events in final year = 5

Benefits of the Bender et al. (2005) approach

- Extremely easy to implement
- Quite often we simulate survival times and then apply Cox models $->$ baseline hazard from which we simulate is irrelevent
- What if we wish to simulate from a more complex and biologically plausible underlying hazard function?
- There is a growing interest in parametric survival models (Royston and Lambert, 2011; Crowther and Lambert, 2014)

Limitations with simulating survival times from standard distributions with proportional hazards

$$T = H_0^{-1}[-\log(u)\exp(-X\beta)]$$

- Must be able to integrate the hazard function in order to calculate the cumulative hazard function
- We then must be able to invert the cumulative hazard function to obtain the simulated survival time

Simulating from a more complex baseline hazard function

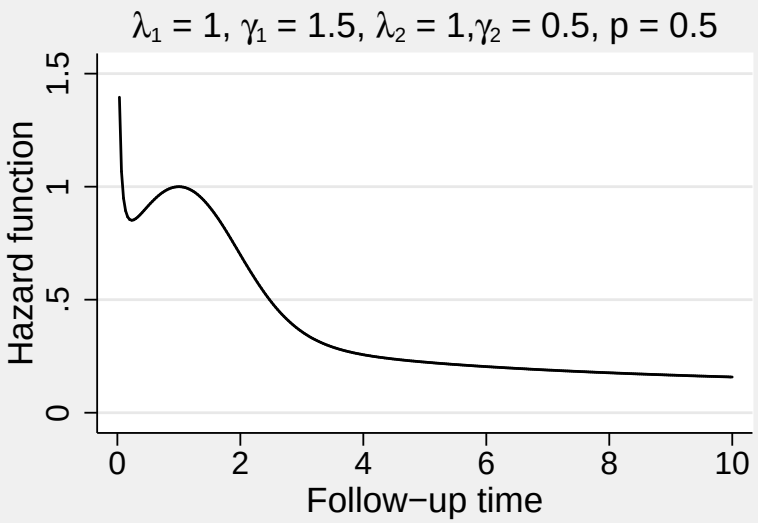
We can use a mixture of parametric distributions

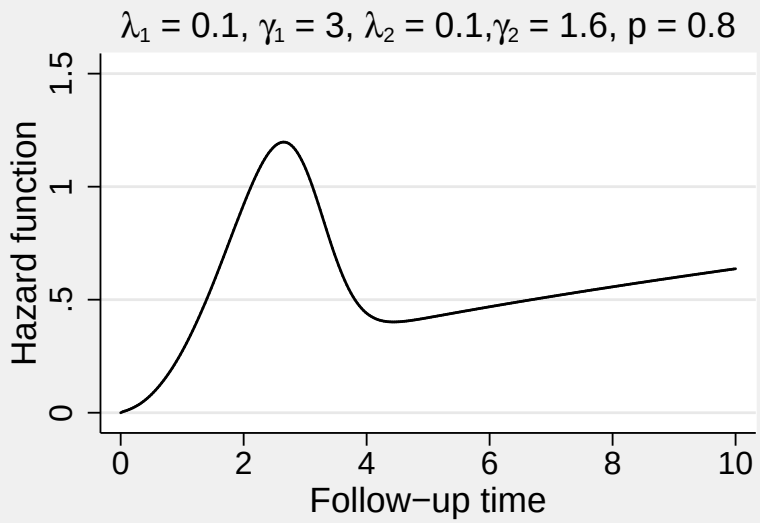
$$S_0(t) = pS_{01}(t) + (1-p)S_{02}(t) \tag{1}$$

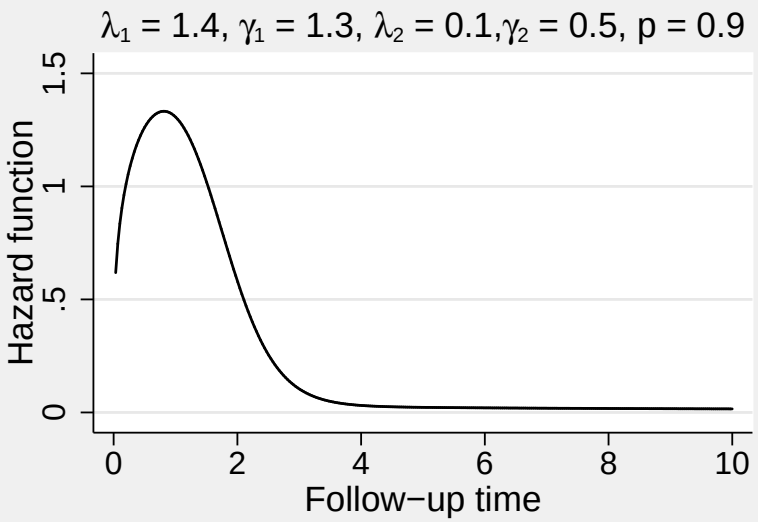
For example a 2-component mixture Weibull

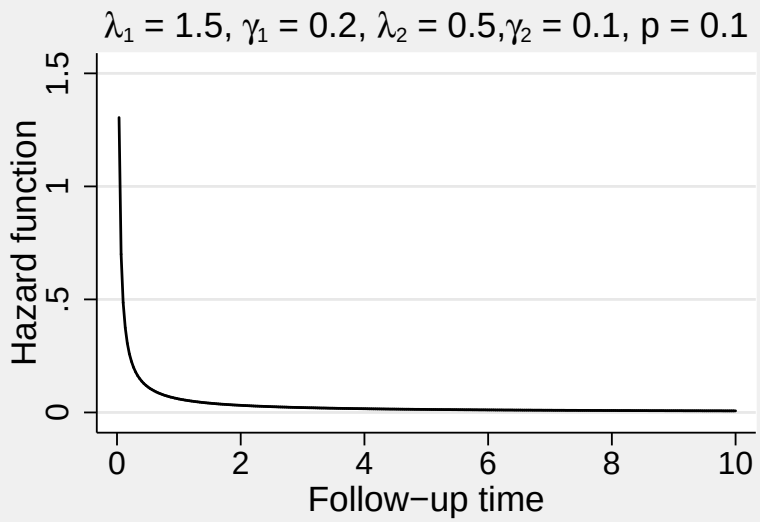
$$S_0(t) = p\exp(-\lambda_1 t^{\gamma_1}) + (1-p)\exp(-\lambda_2 t^{\gamma_2}) \tag{2}$$

with $0 \leq p \leq 1$, and $\lambda_1, \lambda_2, \gamma_1, \gamma_2 > 0$


$\lambda_1 = 1, \gamma_1 = 1.5, \lambda_2 = 1, \gamma_2 = 0.5, p = 0.5$
Hazard function
0
.5
1
1.5
0
2
4
6
8
10
Follow−up time


$\lambda_1 = 0.1, \gamma_1 = 3, \lambda_2 = 0.1, \gamma_2 = 1.6, p = 0.8$
Hazard function
0
.5
1
1.5
0
2
4
6
8
10
Follow−up time


$\lambda_1 = 1.4, \gamma_1 = 1.3, \lambda_2 = 0.1, \gamma_2 = 0.5, p = 0.9$
Hazard function
0
.5
1
1.5
0
2
4
6
8
10
Follow−up time


$\lambda_1 = 1.5, \gamma_1 = 0.2, \lambda_2 = 0.5, \gamma_2 = 0.1, p = 0.1$
Hazard function
0
.5
1
1.5
0
2
4
6
8
10
Follow−up time

Incorporating proportional hazards gives us a survival function

$$S(t) = [p\exp(-\lambda_1 t^{\gamma_1}) + (1-p)\exp(-\lambda_2 t^{\gamma_2})]^{\exp(X\beta)} \quad (3)$$

This model is implemented in the `stmix` command from SSC. Attempting to apply the inversion method, gives

$$S(t) = u, \quad \text{where} \quad u \sim U(0,1) \quad (4)$$

which cannot be re-arranged to directly solve for t.

To solve we can apply iterative root finding techniques, such as Newton-Raphson iterations, or Brent's univariate root finder. I favour the latter, using `mm_root()` from Ben Jann's `moremata` (Jann, 2005)

survsim

survsim *newvarname1* [*newvarname2*] [, *options*]

- mixture
- distribution(exp|gomp|weib)
- lambdas(#), gammas(#)
- covariates(*varname* # [*varname* #] ...)
- maxtime(#)

```
. survsim stime event, mixture dist(weib)
> lambdas(0.1 0.2) gammas(1.2 0.5) p(0.3)
```

Simulating survival times when the cumulative hazard doesn't have a closed form expression - joint model data

$$h(t) = h_0(t) \exp\left[X\beta + \alpha m(t)\right]$$

where

$$m(t) = \beta_{0i} + \beta_{1i} t$$

- To obtain the cumulative hazard function we require numerical integration
- We then require root finding techniques to solve for the simulated survival time, t

Numerical integration

$$\int_{-1}^{1} g(x)\mathrm{d}x = \int_{-1}^{1} W(x)g(x)\mathrm{d}x \approx \sum_{i=1}^{m} w_i g(x_i)$$

where $W(x)$ is a known weighting function and $g(x)$ can be approximated by a polynomial function.

$$\begin{aligned}\int_{t_{0i}}^{t_i} h(x)\mathrm{d}x &= \frac{t_i - t_{0i}}{2} \int_{-1}^{1} h\left(\frac{t_i - t_{0i}}{2}x + \frac{t_{0i} + t_i}{2}\right)\mathrm{d}x \\ &\approx \frac{t_i - t_{0i}}{2} \sum_{i=1}^{m} w_i h\left(\frac{t_i - t_{0i}}{2}x_i + \frac{t_{0i} + t_i}{2}\right)\end{aligned}$$

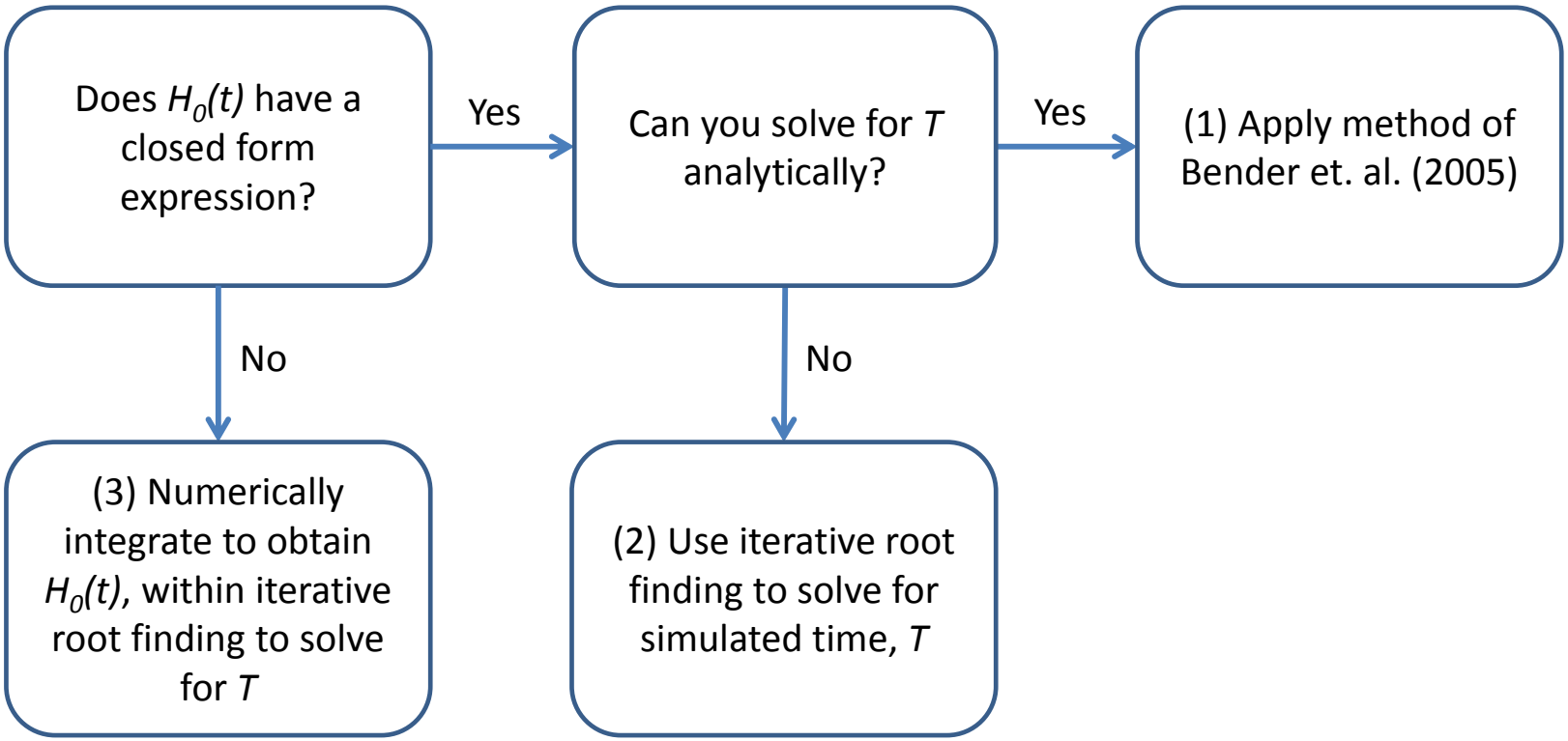
survsim

survsim *newvarname1* [*newvarname2*] [, *options*]

- [log]hazard()
- [log]cumhazard()
- nodes(#)
- covariates(*varname* # [*varname* #] ...)
- tde(*varname* # [*varname* #] ...)
- tdefunction()
- centol(#)
- maxtime(#)

```
. survsim stime event, hazard(0.1:*1.2:*t:^(1.1:-1))
```

Simulating survival data - recap



General survival simulation

Given a well-defined hazard function, $h(t)$, this two-stage algorithm involving

1. Numerical integration
2. Root-finding

provides a framework for general survival simulation which can incorporate:

- Practically *any* user-defined baseline hazard function
- Time-varying covariates
- Time-dependent effects
- Delayed entry
- Extends to competing risks, frailty etc.

Examples

- Fractional polynomial baseline

```
survsim stime event, logh(-18 :+
7.3:*log(#t):-11.5:*#t:^(0.5):*log(#t))
```

Examples

- Fractional polynomial baseline

```
survsim stime event, logh(-18 :+
7.3:*log(#t):-11.5:*#t:^(0.5):*log(#t))
```

- Non-proportional hazards

```
survsim stime event, logh(-18 :+
7.3:*log(#t):-11.5:*#t:^(0.5):*log(#t)) cov(trt -0.7)
tde(trt 1) tdefunc(0.01:*t :+ 0.4:*log(t))
```

Examples

- Joint model data (time-varying covariate)

```
. //Simulate 1000 survival times
. set obs 1000
. //Define the association between the biomarker and survival
. local alpha = 0.25
. //Generate the random intercept and random slopes
. gen b0 = rnormal(0,1)
. gen b1 = rnormal(1,0.5)
. survsim stime event, loghazard(-2.3:+2:*#t:-#t:^(2):+0.12:*#t:^3
> :+ `alpha´ :* (b0 :+ b1 :* #t)) maxt(5)
. //Generate observed biomarker values at times 0, 1, 2, 3 , 4 years
. gen id = _n
. expand 5
. bys id: gen meastime = _n-1
. //Remove observations after event or censoring time
. bys id: drop if meastime>=stime
. //Generate observed biomarker values incorporating measurement error
. gen response = b0 + b1*meastime + rnormal(0,0.5)
```

Practical advice

- Although computation time is often minimal, it may be of use to simulate your 1000 datasets, say, before applying any model fits

Practical advice

- Although computation time is often minimal, it may be of use to simulate your 1000 datasets, say, before applying any model fits
- With the numerical integration, it is important to assess the approximation by setting a seed and using an increasing number of quadrature points

Discussion

- We have described a general framework for the generation of survival data, incorporating any combination of complex hazard functions, time-dependent effects, time-varying covariates, delayed entry, random effects and covariates measured with error (Crowther and Lambert, 2013a)

Discussion

- We have described a general framework for the generation of survival data, incorporating any combination of complex hazard functions, time-dependent effects, time-varying covariates, delayed entry, random effects and covariates measured with error (Crowther and Lambert, 2013a)
- As the procedure relies on numerical integration, it is important to establish the consistency of the simulated survival times by setting a seed and using an increasing number of quadrature nodes

Discussion

- We have described a general framework for the generation of survival data, incorporating any combination of complex hazard functions, time-dependent effects, time-varying covariates, delayed entry, random effects and covariates measured with error (Crowther and Lambert, 2013a)
- As the procedure relies on numerical integration, it is important to establish the consistency of the simulated survival times by setting a seed and using an increasing number of quadrature nodes
- You can also specify a user-defined [log] cumulative hazard function (Royston, 2012) (`stsurvsim`)

Discussion

- We have described a general framework for the generation of survival data, incorporating any combination of complex hazard functions, time-dependent effects, time-varying covariates, delayed entry, random effects and covariates measured with error (Crowther and Lambert, 2013a)
- As the procedure relies on numerical integration, it is important to establish the consistency of the simulated survival times by setting a seed and using an increasing number of quadrature nodes
- You can also specify a user-defined [log] cumulative hazard function (Royston, 2012) (`stsurvsim`)
- Simulating from a fitted model (or observed censoring distribution) can be particularly useful (Royston, 2012)

References I

R. Bender, T. Augustin, and M. Blettner. Generating survival times to simulate Cox proportional hazards models. *Stat Med*, 24(11):1713–1723, 2005.

A. Burton, D. G. Altman, P. Royston, and R. L. Holder. The design of simulation studies in medical statistics. *Stat Med*, 25(24):4279–4292, 2006.

M. J. Crowther and P. C. Lambert. Simulating complex survival data. *Stata J*, 12(4): 674–687, 2012.

M. J. Crowther and P. C. Lambert. Simulating biologically plausible complex survival data. *Stat Med*, 32(23):4118–4134, 2013a.

M. J. Crowther and P. C. Lambert. stgenreg: A Stata package for the general parametric analysis of survival data. *J Stat Softw*, 53(12), 2013b.

M. J. Crowther and P. C. Lambert. A general framework for parametric survival analysis. *Stat Med*, In Press, 2014.

B. Jann. MOREMATA: Stata module (Mata) to provide various functions. Statistical Software Components, Boston College Department of Economics, 2005.

P. Royston. Tools to simulate realistic censored survival-time distributions. *Stata J*, 12 (4):639–654, 2012.

P. Royston and P. C. Lambert. *Flexible Parametric Survival Analysis Using Stata: Beyond the Cox Model*. Stata Press, 2011.