Title

stata.com

stintcox — Cox proportional hazards model for interval-censored survival-time data

Description

`stintcox` fits semiparametric Cox proportional hazards models to interval-censored survival-time data or, more precisely, event-time data, which may contain right-censored, left-censored, and interval-censored observations. With interval-censored data, the event-time variables are specified with the `stintcox` command instead of using `stset`. All `st` settings are ignored by `stintcox`.

Quick start

Cox proportional hazards model with covariates `x1` and `x2` fit to interval-censored data with lower and upper interval endpoints `t1` and `t2`

```
stintcox x1 x2, interval(t1 t2)
```

As above, but estimate the baseline hazard function based on all observed intervals instead of the default reduced set

```
stintcox x1 x2, interval(t1 t2) full
```

Use less stringent convergence criteria to explore initial results more quickly

```
stintcox x1 x2, interval(t1 t2) favorspeed
```

Fit a stratified Cox model with strata defined by levels of `svar`

```
stintcox x1 x2, interval(t1 t2) strata(svar)
```

Report the log-likelihood model test instead of the default Wald model test, and report regression coefficients instead of hazard ratios

```
stintcox x1 x2, interval(t1 t2) lrmodel nohr
```

Report OIM standard-error estimates instead of the default OPG estimates

```
stintcox x1 x2, interval(t1 t2) vce(oim)
```

After estimation, report regression coefficients instead of hazard ratios

```
stintcox, nohr
```

After estimation, report OPG standard-error estimates using fixed step size instead of the default adaptive step size

```
stintcox, vce(opg, stepsize(fixed))
```

After estimation, save estimated baseline hazard contributions to `basehc.dta`, and store estimation results as `intcox`

```
stintcox, saving(basehc)
estimates store intcox
```

Menu

Statistics > Survival analysis > Regression models > Interval-censored Cox PH model

Syntax

```
stintcox [indepvars] [if] [in], interval(t_l t_u) [options]
```

options	Description
Model	
* interval(t_l t_u)	specify lower and upper endpoints for the event-time interval
reduced	estimate baseline hazard function using a reduced set of time intervals; the default
full	estimate baseline hazard function using all time intervals
strata(*varlist*)	specify strata variables
favoraccuracy	favor accuracy of results over speed; the default
favorspeed	favor speed possibly over accuracy of results
SE	
vce(*vcetype*)	*vcetype* may be opg or oim; may be specified on replay of results; default is opg
Reporting	
level(#)	set confidence level; default is level(95)
lrmodel	perform the likelihood-ratio model test instead of the default Wald model test
saving(*filename*[, replace])	save estimates of baseline hazard contributions to *filename*; use replace to overwrite existing *filename*
nohr	report regression coefficients, not hazard ratios
noheader	suppress header from coefficient table
[no]log	display or suppress EM and VCE iteration logs; default is log
[no]dots	display all EM and VCE iterations as dots or suppress both
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
EM options	
emiterate(#)	maximum number of EM iterations; default is emiterate(5000)
emtolerance(#)	tolerance for the coefficient vector; default is emtolerance(1e-6)
emltolerance(#)	tolerance for the log likelihood; default is emltolerance(1e-7)
emhsgtolerance(#)	tolerance for the scaled gradient; default is emhsgtolerance(1e-5)
noemhsgtolerance	do not perform the scale-gradient convergence check
from(*init_specs*)	initial values for the regression coefficients

[no]emlog[(#)]	display or suppress EM iteration log; default is emlog(100), which displays the log-likelihood value every 100 iterations
[no]emdots[(#)]	display or suppress EM iteration dots; default is noemdots
coeflegend	display legend instead of statistics

*interval(t_l t_u) is required.

varlist may contain factor variables; see [U] 11.4.3 Factor variables.

by, collect, and statsby are allowed; see [U] 11.1.10 Prefix commands.

coeflegend does not appear in the dialog box.

See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.

vcetype	Description
SE	
opg[, *vce_options*]	outer product of the gradient (OPG) vectors; the default
oim[, *vce_options*]	observed information matrix (OIM)

vce_options	Description
SE	
stepsize(adaptive|fixed [#])	use adaptive or fixed step size to compute VCE; default is adaptive step size
derivopts	options to control computation of numerical derivatives when adaptive step size is used
iterate(#)	maximum number of iterations to compute VCE; default is iterate(5000)
tolerance(#)	profile log-likelihood tolerance to compute VCE; default is tolerance(1e-6)
[no]vcedots[(#)]	display or suppress VCE iteration dots; default is to display a dot every iteration, meaning vcedots or vcedots(1)
[no]dots[(#)]	synonym for vcedots, vcedots(), and novcedots
post	replace the current e(V) with the specified VCE type; can be used only on replay with opg or oim

dots, dots(), nodots, and post do not appear in the dialog box.

Options

Model

interval(t_l t_u) specifies two time variables that contain the endpoints of the event-time interval. t_l represents the lower endpoint, and t_u represents the upper endpoint. interval() is required.

The interval time variables t_l and t_u should have the following form:

Type of observations		t_l	t_u
interval-censored	$(a,b]$	a	b
left-censored	$(0,b]$	.	b
left-censored	$(0,b]$	0	b
right-censored	$(a,+\infty)$	a	.
missing		.	.
missing		0	.

Note that $t_l = t_u$ is not allowed with left-censored or interval-censored observations.

`reduced`, the default, specifies that the baseline hazard function be estimated using a reduced (innermost) set of time intervals. This allows the estimator of the cumulative baseline hazard function to change its values only at the endpoints of the innermost time intervals, which were originally used by Turnbull (1976) to estimate the survivor function in the one-sample case. This option may not be combined with `full`.

`full` specifies that the baseline hazard function be estimated using all observed time intervals. In this case, the estimator of the cumulative baseline hazard function can potentially change its values at the endpoints of all the observed time intervals. This is the approach used by Zeng, Mao, and Lin (2016). It is more time consuming, but it may provide more accurate results. `full` may not be combined with `reduced`.

`strata(`*varlist*`)` specifies the stratification variables. Observations with equal values of the strata variables are assumed to be in the same stratum. Stratified estimates (equal regression coefficients across strata but with a baseline hazard unique to each stratum) are then obtained.

`favoraccuracy`, the default, and `favorspeed` control the tradeoff between accuracy of the results and the execution speed. `favoraccuracy` specifies that the command run longer to obtain more accurate results. `favorspeed` specifies that the command run faster at the possible expense of reduced accuracy of the results. You can use `favorspeed` for a quick initial exploration of the results and `favoraccuracy` for final reporting of the results.

When you specify `favorspeed`, `stintcox` uses less stringent convergence criteria to obtain the results. Specifically, it assumes lower EM coefficient, likelihood, and VCE tolerances of 0.0001 and implies option `noemhsgtolerance`. In addition, it uses a fixed step size with a multiplier of 5 instead of an adaptive step size when computing VCE. That is, specifying `favorspeed` is equivalent to specifying `emtolerance(0.0001)`, `emltolerance(0.0001)`, `noemhsgtolerance`, and `vce(, tolerance(0.0001) stepsize(fixed))`.

SE

`vce(`*vcetype*`)` specifies the type of standard error estimate reported. `vce()` may be specified at the time of estimation or when replaying results. *vcetype* may be one of the following:

`vce(opg[ , `*vce_options*`])` uses the sum of the outer product of the gradient (OPG) vectors based on the profile log likelihood; see *Methods and formulas*. `vce(opg)` is the default.

`vce(oim[ , `*vce_options*`])` uses the sum of the observed information matrix (OIM) vectors based on the profile log likelihood; see *Methods and formulas*.

vce_options include `stepsize()`, *derivopts* with adaptive step size, `iterate(#)`, `tolerance(#)`, `dots`, `dots()`, `nodots`, and `post`. These options are available only with `vce(opg)` and `vce(oim)`.

`stepsize(adaptive | fixed [#])` specifies the step size for computing numerical derivatives with methods `opg` and `oim`. The default is `stepsize(adaptive)`, which uses adaptive step size in computations; see [M-5] **deriv()**. `stepsize(fixed)` uses a fixed step size equal to $\delta_n = 5n^{-1/2}$. `stepsize(fixed #)` uses a fixed step size equal to $\# \times n^{-1/2}$.

derivopts are allowed only with `stepsize(adaptive)` and include `search()`, `h()`, `scale()`, and `bounds()`.

`search(`*search_type*`)` specifies the approach used to search for an optimal step size for computing the numerical derivatives; three approaches are offered: `bracket`, `interpolate`, and `off`; see `deriv_init_search()` in [M-5] **deriv()**. The default is `search(interpolate)`. In some cases, such as when factor variables have highly unbalanced levels, the search may lead to the step size that is too small or too large, which may lead to the error message that the estimates of baseline hazard contributions cannot be computed because the VCE matrix is close to being singular. Trying `search(bracket)` may be helpful in this case.

`h(#` | *matname*`)` specifies the h values, which are multipliers for step size used to compute numerical derivatives; see `deriv_init_h()` in [M-5] **deriv()**. You can specify the same h value, #, for all parameters or parameter-specific h values as a Stata matrix (vector) *matname*.

`scale(#` | *matname*`)` specifies the starting scale values used to compute numerical derivatives; see `deriv_init_scale()` in [M-5] **deriv()**. You can specify the same initial scale value, #, for all parameters or parameter-specific initial scale values as a Stata matrix (vector) *matname*.

`bounds(`$\#_1$ $\#_2$`)` specifies the minimum and maximum values used to search for optimal scale values; see `deriv_init_bounds()` in [M-5] **deriv()**. The default is `bounds(1e-6 1e-5)`.

`iterate(#)` specifies the maximum number of iterations to compute the VCE based on the profile log likelihood. The default is `iterate(5000)`.

`tolerance(#)` specifies the tolerance for the profile log likelihood used to compute the VCE. The default is `tolerance(1e-6)`.

`vcedots`, `vcedots(#)`, and `novcedots` display or suppress iteration dots showing the progress of the variance estimation. The dots are displayed by default, but you can use `novcedots` to suppress them. By default, the dot is displayed every iteration, but you can change this by specifying `vcedots(#)`.

When a fixed step size is used, an iteration corresponds to one derivative computation with respect to a regression coefficient. When an adaptive step size is used, an iteration corresponds to one call of the Mata `deriv()` function, which may be called multiple times to compute one derivative with respect to one regression coefficient. Thus, you will typically see more iteration dots with VCE estimation using an adaptive step size than using a fixed step size.

`dots`, `dots(#)`, and `nodots` are synonyms for `vcedots`, `vcedots(#)`, and `novcedots`, respectively. These options do not appear in the dialog box.

`post` can be used only on replay with `vce(opg)` or `vce(oim)`. It replaces the current `e(V)` with the specified *vcetype*. When `vce(opg)` or `vce(oim)` is used on replay without `post`, the coefficient table will display the standard error of the specified *vcetype*, but `e(V)` will remain unchanged. This option does not appear in the dialog box.

Reporting

`level(#)`, `lrmodel`; see [R] **Estimation options**.

`saving(`*filename*`[ , replace ])` saves the estimated baseline hazard contributions in *filename*`.dta`. The `replace` option specifies to overwrite *filename*`.dta` if it exists. If option `saving()` is not specified, `stintcox` saves estimation results in a temporary file for later access by postestimation commands. This temporary file will be overridden every time `stintcox` is run and will also be erased if the current estimation results are cleared. `saving()` may be specified during estimation or on replay.

Because the file containing the baseline hazard contributions is considered to be part of estimation results, you must use option `saving()` before storing or saving your estimation results using `estimates store` or `estimates save`.

`nohr` specifies that regression coefficients be displayed rather than exponentiated regression coefficients or hazard ratios. This option affects only how results are displayed and not how they are estimated. `nohr` may be specified at estimation time or when replaying results.

`noheader` suppresses the output header, either at estimation or upon replay.

`log` and `nolog` are synonyms for `emlog` and `vcedots` and for `noemlog` and `novcedots`, respectively. The default is `log`. If `log` or `nolog` is specified, any other option that controls an iteration log is ignored. `log` and `nolog` may not be combined with `dots` and `nodots`.

`nodots` and `dots` are synonyms for `noemdots` and `novcedots` and for `emdots` and `vcedots`, respectively. The default is `emlog` and `vcedots`. If `dots` or `nodots` is specified, any other option that controls an iteration log is ignored. `dots` and `nodots` may not be combined with `log` and `nolog`.

display_options: `noci`, `nopvalues`, `noomitted`, `vsquish`, `noemptycells`, `baselevels`, `allbaselevels`, `nofvlabel`, `fvwrap(#)`, `fvwrapon(`*style*`)`, `cformat(%`*fmt*`)`, `pformat(%`*fmt*`)`, `sformat(%`*fmt*`)`, and `nolstretch`; see [R] **Estimation options**.

EM options

`emiterate(#)`, `emtolerance(#)`, `emltolerance(#)`, `emhsgtolerance(#)`, `noemhsgtolerance`, and `from()`; see `iterate()`, `tolerance()`, `ltolerance()`, `nrtolerance()`, `nonrtolerance`, and `from()` in [R] **Maximize**. These options control the EM optimization process. The defaults are `emiterate(5000)`, `emtolerance(1e-6)`, `emltolerance(1e-7)`, and `emhsgtolerance(1e-5)`.

`emlog`, `emlog(#)`, and `noemlog` display or suppress an iteration log showing the progress of the EM algorithm. The log is displayed by default, and `noemlog` suppresses it; see `set iterlog` in [R] ***set iter***. `emlog`, the default, displays the log-likelihood value every 100 iterations and is equivalent to `emlog(100)`. `emlog(#)` displays the log-likelihood value every #th iterations.

`noemdots`, `emdots(#)`, and `emdots` control the display of the EM iteration log as dots. By default, the EM iteration log displays the log-likelihood value every 100 iterations; that is, `noemdots` is implied. Instead, you can specify `emdots` to display every 100 iterations as a dot or `emdots(#)` to display every # iterations as a dot. This is a useful alternative for long EM iteration logs.

The following option is available with `stintcox` but is not shown in the dialog box:

`coeflegend`; see [R] **Estimation options**.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Introduction

`stintcox` fits the Cox proportional hazards model to interval-censored survival-time data. In the context of interval-censored data, the term "failure-time data" or, more generally, "event-time data" is more appropriate, so we will use it in that context.

Interval-censoring occurs when the failure time or, more generally, the event time of interest is not exactly observed but is known only to lie within some interval. See *Introduction* in [ST] **stintreg** for details about interval-censored data. If you have right-censored data, see [ST] **stcox**. See [ST] **stintreg** for fitting parametric models to interval-censored data.

The Cox proportional hazards model was first introduced by Cox (1972) for right-censored survival data. For an introduction to interval-censored data, see Finkelstein and Wolfe (1985), Odell, Anderson, and D'Agostino (1992), Rabinowitz, Tsiatis, and Aragon (1995), Huang and Wellner (1997), Lindsey (1998), Lindsey and Ryan (1998), Sun (2006), and Sun and Li (2014).

The Cox proportional hazards model specifies that the hazard function of the event time conditional on covariates takes the form

$$h(t;\mathbf{x}) = h_0(t)\exp(\beta_1 x_1 + \cdots + \beta_p x_p)$$

where $\beta_1,\ldots,\beta_p$ are unknown regression coefficients, $\mathbf{x}$ are p-vector time-independent covariates, and $h_0(t)$ is an arbitrary baseline hazard function. Under the proportional-hazards assumption, the hazard ratios, or exponentiated regression coefficients $\exp(\beta_1),\ldots,\exp(\beta_p)$, are constant over time. As with right-censored data, the Cox proportional hazards model is appealing for interval-censored data because it does not require parameterization of the baseline hazard function and, for low event rates, the exponentiated regression parameters approximate the log relative risks.

The partial-likelihood approach (Cox 1972, 1975) is used to estimate parameters of the Cox model with right-censored data, in which some of the event times are observed exactly while others are known to be longer than the duration of follow-up. Under interval-censoring, however, none of the event times are observed exactly. Thus, it is much more challenging to deal with interval-censored data than right-censored data, both theoretically and computationally. In particular, the traditional partial-likelihood approach is not applicable.

Several authors (Cai and Betensky 2003; Zhang, Hua, and Huang 2010; Wang et al. 2016) have proposed spline methods to fit the Cox proportional hazards model to interval-censored data. Spline methods have limitations, however. First, the choices for the type of spline and the number and positions of knots are arbitrary, and different choices may yield conflicting results. Second, the analysis will be biased if the event-time distribution is not well approximated by the chosen spline function. Finally, the variance estimation is difficult given the data-dependent choices of spline functions (Zhang, Hua, and Huang 2010).

Direct maximum-likelihood optimization for the Cox model with interval-censored data using, for instance, the Newton–Raphson algorithm is highly unstable (Sun 2006; Finkelstein 1986).

Zeng, Mao, and Lin (2016) developed a novel EM algorithm for efficient nonparametric maximum-likelihood estimation (NPMLE) of the Cox proportional hazards model with interval-censored data. It allows a completely arbitrary event-time distribution and results in consistent, asymptotically normal, and asymptotically efficient estimators of the regression parameters. And it reduces to the classical maximum partial-likelihood estimation in the special case of right-censored data. For more details about this method, see *Methods and formulas*.

Unlike with right-censored data, the estimation of regression coefficients must be performed jointly with estimation of the baseline cumulative hazard function for interval-censored data. Stata provides two ways to estimate the baseline cumulative hazard function. One is to use all distinct lower and upper interval endpoints as time points for estimating the baseline cumulative hazard function. This is available by specifying the `full` option.

For large datasets with many distinct time points, this approach may become time consuming. An alternative is to estimate the baseline cumulative hazard at fewer time points. Turnbull (1976) proposed a method for estimating the one-sample survivor function at a subset of time intervals, known as Turnbull's intervals, or innermost intervals, or regions of the maximal cliques. Thus, one can allow the baseline cumulative hazard function to change its values only at the endpoints of those time intervals and set baseline hazard contributions to zero for the other times. This is available via the `reduced` option and, for computational reasons, is the default in `stintcox`.

As mentioned above, NPMLE is a computationally intensive approach, so `stintcox` may take some time to run, especially for large datasets. The speed of the command depends on the desired accuracy of the computations, among other things. The higher the accuracy, the more iterations are needed to achieve that accuracy, and thus the longer the command runs. It is important to have high accuracy for the final reporting of the results, but the speed may become an issue during the exploratory stage of the project. Thus, you may consider using the `favorspeed` option to expedite the command execution. When you specify this option, `stintcox` uses less stringent convergence criteria to produce the results more quickly; see description of option `favorspeed` for details.

Just like `stintreg`, `stintcox` requires the outcome to be stored in the dataset as interval data. That is, two time variables, t_l and t_u, that contain the endpoints of the event-time interval must be specified in the `interval()` option. If the data are left-censored, the lower endpoint is zero and may be represented in t_l by either a missing value (`.`) or zero. If the data are right-censored, the upper endpoint is $+\infty$ and is represented in t_u by a missing value. Uncensored data are represented by the two endpoints that are equal. If $0 < t_l < t_u < \infty$, the data are interval-censored. Truly missing values must be represented by missing values in both t_l and t_u or by a 0 in t_l and a missing value in t_u. Typing `stset` ([ST] **stset**) is unnecessary, and `stintcox` will ignore any settings of `stset` for the usual trivariate response variable (t_0, t, d). `stintcox` does not support data exhibiting delayed entry, gaps, time-varying covariates, and multiple failures.

Case II interval-censored data

Case II interval-censored data arise when there are potentially two or more examination times for each study subject. In this case, the interval that brackets the event time of interest, the event-time interval, is recorded for each subject. The event of interest may occur before the first examination time, resulting in a left-censored observation; after the last examination time, resulting in a right-censored observation; or between two examination times, resulting in a truly interval-censored observation.

▷ Example 1: Case II interval-censoring

Zeng, Mao, and Lin (2016) considered a cohort study of injecting drug users in Thailand. Subjects were initially seronegative for the HIV-1 virus. They were followed and assessed for HIV-1

seropositivity through blood tests approximately every four months. The event of interest was time to HIV-1 seropositivity. Because the subjects were tested approximately every four months, the exact time of HIV-1 seropositivity was not observed and was known to fall only in the interval between blood tests.

The data used in this example are the data provided in supplementary materials of Zeng, Mao, and Lin (2016), which are based on the study described above. The dataset contains 1,124 subjects: 76 are females and 1,048 are males. We wish to identify the factors that influence HIV-1 infection. The covariates that we are interested in are age at recruitment (`age`), sex (`male`), history of needle sharing (`needle`), history of drug injection before recruitment (`inject`), and whether a subject has been in jail at the time of recruitment (`jail`). The dataset also contains two variables, `ltime` and `rtime`, that record the last time of blood test when the HIV-1 was seronegative and the first time of blood test when the HIV-1 was seropositive.

```
. use https://www.stata-press.com/data/r17/idu
(Modified Bangkok IDU Preparatory Study)
. describe
Contains data from https://www.stata-press.com/data/r17/idu.dta
 Observations:         1,124                  Modified Bangkok IDU Preparatory
                                                Study
    Variables:             8                  15 Dec 2020 13:34
                                              (_dta has notes)
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
age             byte    %8.0g                 Age (in years)
male            byte    %8.0g      yesno      Male
needle          byte    %8.0g      yesno      Shared needles
jail            byte    %8.0g      yesno      Imprisoned
inject          byte    %8.0g      yesno      Injected drugs before recruitment
ltime           double  %10.0g                Last time seronegative for HIV-1
rtime           double  %10.0g                First time seropositive for HIV-1
age_mean        double  %10.0g                Centered age (in years)
-------------------------------------------------------------------------------
Sorted by:
```

We want to use `stintcox` to fit a Cox proportional hazards model in which the time to HIV-1 infection depends on `age`, `male`, `needle`, `inject`, and `jail`. To make the interpretation of the baseline hazard function more reasonable, we will use the centered age variable, `age_mean`. (Remember that a baseline hazard function corresponds to all covariates equal to zero, and `age` of zero would not make sense for our sample of subjects. See *Making baseline reasonable* in [ST] **stcox postestimation** for more details.)

Unlike `stcox`'s specification, in which the survival variables are set using `stset` and do not appear in the command, the interval time variables `ltime` and `rtime` must be specified in the `stintcox`'s option `interval()`. Also, recall that the command implements an NPMLE method, which is computationally intensive, so it may take a little longer to run on our dataset of 1,124 observations:

```
. stintcox age_mean i.male i.needle i.inject i.jail, interval(ltime rtime)
note: using adaptive step size to compute derivatives.
Performing EM optimization (showing every 100 iterations):
Iteration 0:   log likelihood = -1086.2564
Iteration 100:  log likelihood = -597.65634
Iteration 200:  log likelihood = -597.57555
Iteration 295:  log likelihood = -597.56443
Computing standard errors: ......................... done
```

```
Interval-censored Cox regression                  Number of obs     =  1,124
Baseline hazard: Reduced intervals                       Uncensored =      0
                                                     Left-censored =     41
                                                    Right-censored =    991
                                                    Interval-cens. =     92
                                                  Wald chi2(5)      =  17.10
Log likelihood = -597.56443                       Prob > chi2       = 0.0043
```

	Haz. ratio	OPG std. err.	z	P>|z|	[95% conf.	interval]
age_mean	.9684341	.0126552	-2.45	0.014	.9439452	.9935582
male						
Yes	.6846949	.1855907	-1.40	0.162	.4025073	1.164717
needle						
Yes	1.275912	.2279038	1.36	0.173	.8990401	1.810768
inject						
Yes	1.250154	.2414221	1.16	0.248	.8562184	1.825334
jail						
Yes	1.567244	.3473972	2.03	0.043	1.014982	2.419998

```
Note: Standard-error estimates may be more variable for small datasets and
      datasets with low proportions of interval-censored observations.
```

The header above the coefficient table summarizes censored observations. There are 991 subjects who did not test positive for HIV-1 by the last visit, resulting in right-censored observations. There are 41 subjects who tested positive for HIV-1 at their first follow-up, resulting in left-censored observations. The remaining 92 subjects are interval-censored.

The coefficient table shows that age is significantly associated with lower risk of HIV-1 infection and there is some evidence that being in jail at enrollment is associated with higher risk of HIV-1 infection. Being a female, needle sharing, and history of drug injection are associated with the higher risk of HIV-1 infection, although without statistical significant evidence. Our findings are consistent with those in Zeng, Mao, and Lin (2016), although they used a time-varying covariate for imprisonment.

The command displays a note following the command that an adaptive step size is used to compute derivatives during VCE computation. The command also displays a note following the output table about potential variability of the standard error estimates. We address all of this in more detail in *Standard-error estimation with interval-censored data*.

`stintcox` uses the EM algorithm to estimate parameters. EM algorithms are known to require many iterations. Thus, by default, `stintcox` displays log-likelihood values every 100 iterations. You can change how often to display the iteration log by specifying the `emlog(#)` option. For example, `emlog(1)` will display every iteration. You can also use the `noemlog` option to suppress the iteration log.

Similarly, the progress of the VCE computation is displayed with a dot for each iteration. With more regression coefficients and with larger datasets, you may see many more dots. In this case, you may consider displaying a dot every # iterations by specifying `vcedots(#)`. To suppress the dots, use `novcedots`.

◁

▷ Example 2: Speed versus accuracy

By default, `stintcox` uses Stata's standard convergence rules for estimation of parameters. For instance, the parameter tolerance of 1e–6 is used as a stopping rule, and the Hessian scale-gradient tolerance of 1e–5 is used to check for convergence. More stringent criteria typically require more iterations and thus lead to longer execution times.

Although high accuracy of the results is important for final reporting, it may be reasonable to consider less stringent criteria during exploratory analysis in favor of speed. `stintcox` provides the `favorspeed` option for this.

```
. stintcox age_mean i.male i.needle i.inject i.jail, interval(ltime rtime)
> favorspeed
note: using fixed step size with a multiplier of 5 to compute derivatives.
note: using EM and VCE tolerances of 0.0001.
note: option noemhsgtolerance assumed.

Performing EM optimization (showing every 100 iterations):
Iteration 0:    log likelihood = -1086.2564
Iteration 32:   log likelihood = -598.60293

Computing standard errors: ..... done

Interval-censored Cox regression                  Number of obs     =  1,124
Baseline hazard: Reduced intervals                       Uncensored =      0
                                                    Left-censored =     41
                                                   Right-censored =    991
                                                   Interval-cens. =     92

                                                  Wald chi2(5)      =  16.95
Log likelihood = -598.60293                       Prob > chi2       = 0.0046
```

	Haz. ratio	OPG std. err.	z	P>|z|	[95% conf.	interval]
age_mean	.9684228	.0126481	-2.46	0.014	.9439476	.9935326
male						
Yes	.6853044	.1873617	-1.38	0.167	.4010197	1.17112
needle						
Yes	1.275045	.2272609	1.36	0.173	.899103	1.80818
inject						
Yes	1.251637	.2414583	1.16	0.245	.8575707	1.826784
jail						
Yes	1.566873	.3479667	2.02	0.043	1.013914	2.421398

```
Note: Standard-error estimates may be more variable for small datasets and
      datasets with low proportions of interval-censored observations.
```

The above takes only a few seconds to run. And the results happen to be very similar to the results from the previous two examples.

Following the command specification, we now see additional notes about the updated convergence criteria. The EM and VCE tolerances are now 0.0001 compared with the defaults of 1e–6. The tolerance check for the Hessian scaled gradient is suppressed. And a fixed step size with the multiplier of 5 is used to compute derivatives during the VCE computation.

◁

▷ Example 3: Full versus reduced sets of time intervals

By default, `stintcox` uses a reduced set of intervals (option `reduced`) to estimate the baseline hazard function. You can specify the `full` option to estimate the baseline hazard using all time intervals. This approach may be much more time consuming, especially for large datasets. For demonstration purposes, we will also use the `favorspeed` option from example 2 to speed up execution.

```
. stintcox age_mean i.male i.needle i.inject i.jail, interval(ltime rtime) full
> favorspeed
note: using fixed step size with a multiplier of 5 to compute derivatives.
note: using EM and VCE tolerances of 0.0001.
note: option noemhsgtolerance assumed.

Performing EM optimization (showing every 100 iterations):
Iteration 0:   log likelihood = -951.11659
Iteration 42:  log likelihood = -599.05659

Computing standard errors: ..... done

Interval-censored Cox regression                  Number of obs     =  1,124
Baseline hazard: All intervals                           Uncensored =      0
                                                     Left-censored =     41
                                                    Right-censored =    991
                                                    Interval-cens. =     92
                                                  Wald chi2(5)      =  16.94
Log likelihood = -599.05659                       Prob > chi2       = 0.0046
```

	Haz. ratio	OPG std. err.	z	P>|z|	[95% conf.	interval]
age_mean	.9685754	.0126454	-2.45	0.014	.9441053	.9936797
male						
Yes	.6845339	.1871941	-1.39	0.166	.4005194	1.169947
needle						
Yes	1.276357	.2276023	1.37	0.171	.8998792	1.810339
inject						
Yes	1.252501	.2416295	1.17	0.243	.8581566	1.828058
jail						
Yes	1.566614	.3479196	2.02	0.043	1.013734	2.42103

```
Note: Standard-error estimates may be more variable for small datasets and
      datasets with low proportions of interval-censored observations.
```

The results are very similar between the two ways of estimating the baseline hazard in this example. ◁

Standard-error estimation with interval-censored data

With interval-censored data, the NPMLE approach estimates regression coefficients jointly with the contributions to the baseline cumulative hazard function, which is infinite dimensional. The inverse of the entire information matrix for all the parameters, which is typically used to estimate standard errors, does not provide a valid estimator of the variance–covariance matrix in this case.

Murphy and van der Vaart (2000) and Zeng, Mao, and Lin (2016) propose to estimate VCE for regression coefficients using the profile log likelihood, which is obtained by maximizing the likelihood by holding the regression coefficients fixed. Two different methods based on this profile log likelihood

are offered to calculate the VCE for regression coefficients. The `vce(opg)` method, the default, uses the first-order numerical derivatives, whereas the `vce(oim)` method uses the second-order numerical derivatives. When there are sufficient data to estimate the second-order derivatives reliably, the OIM method will generally provide more accurate results. With small samples, however, Zeng, Gao, and Lin (2017) found that the OIM method may lead to a negative definite matrix of second-order derivatives, which is not invertible.

For some datasets, the choice of the step size used to compute numerical derivatives may also affect the estimates. By default, `stintcox` uses an adaptive step size, `stepsize(adaptive)`, which is calculated from the data and updated during the computation of numerical derivatives. Zeng, Mao, and Lin (2016) and Zeng, Gao, and Lin (2017) found a fixed ad hoc step size of $\delta_n = 5n^{-1/2}$ to work well in the examples they considered. You can specify the `stepsize(fixed)` option to use that step size. And you can also provide your own multiplier instead of the above 5 by specifying `stepsize(fixed #)`.

When an adaptive step size is used, `stintcox` searches for an optimal step size to use to compute the numerical derivatives. In some cases, such as in the presence of covariates of different magnitudes, the search may lead to step sizes that are too large or too small such that the VCE matrix becomes close to being singular. In that case, you may consider trying a different search method, for example, `vce(, search(bracket))`, or using a fixed step size, `vce(, stepsize(fixed))`.

For small datasets or datasets with low proportions of interval-censored observations, the standard error estimates may be more variable between different estimation methods. In that case, you may want to compare several VCE estimation methods. `stintcox` provides the OPG and OIM methods on replay so that you do not need to rerun the estimation command.

▷ Example 4: Variance estimation

Continuing with example 1, we note the dataset contains only 92 interval-censored observations out of a total of 1,124 observations. Let's compare the OPG and OIM methods using both an adaptive and a fixed step size.

Let's refit our model using the default settings and save the estimation results for later comparison.

```
. stintcox age_mean i.male i.needle i.inject i.jail, interval(ltime rtime)
  (output omitted)
```

To save estimation results after `stintcox`, we must save the estimated baseline hazard contributions in a dataset before using `estimates store` because they are an integral part of the estimation results. This can be done after estimation, on replay, or we could have specified the `saving()` option during estimation above.

```
. stintcox, saving(basehc, replace)
note: file basehc.dta not found; file saved.
. estimates store opg_adapt
```

We do not need to refit the model to compute OPG and OIM VCEs. To compute OIM estimates using an adaptive step size, we simply type

```
. stintcox, vce(oim, post)
note: using adaptive step size to compute derivatives.
Computing standard errors: ..... done
Interval-censored Cox regression                  Number of obs     =  1,124
Baseline hazard: Reduced intervals                      Uncensored  =      0
                                                  Left-censored     =     41
                                                  Right-censored    =    991
                                                  Interval-cens.    =     92
                                                  Wald chi2(5)      =  17.10
Log likelihood = -597.56443                       Prob > chi2       = 0.0043
```

	Haz. ratio	OIM std. err.	z	P>|z|	[95% conf.	interval]
age_mean	.9684341	.0142529	-2.18	0.029	.9408979	.9967761
male						
Yes	.6846949	.2175916	-1.19	0.233	.3672745	1.276449
needle						
Yes	1.275912	.2688544	1.16	0.248	.8442278	1.928333
inject						
Yes	1.250154	.3363599	0.83	0.407	.7378083	2.11828
jail						
Yes	1.567244	.4679689	1.50	0.132	.8729178	2.813843

```
Note: Standard-error estimates may be more variable for small datasets and
      datasets with low proportions of interval-censored observations.
. estimates store oim_adapt
```

We also specified the `post` suboption with `vce(oim)` to store the `oim` estimates in `e(V)` and stored the updated estimation results.

We repeat the same steps using a fixed step size for `oim`,

```
. stintcox, vce(oim, stepsize(fixed) post)
  (output omitted)
. estimates store oim_fixed
```

and for `opg`,

```
. stintcox, vce(opg, stepsize(fixed) post)
  (output omitted)
. estimates store opg_fixed
```

We compare the results using `estimates table`:

```
. estimates table opg* oim*, b(%9.4f) se(%9.4f) t p
```

Variable	opg_adapt	opg_fixed	oim_adapt	oim_fixed
age_mean	-0.0321	-0.0321	-0.0321	-0.0321
	0.0131	0.0131	0.0147	0.0120
	-2.45	-2.46	-2.18	-2.67
	0.0141	0.0141	0.0293	0.0077
male				
Yes	-0.3788	-0.3788	-0.3788	-0.3788
	0.2711	0.2736	0.3178	0.2994
	-1.40	-1.38	-1.19	-1.27
	0.1623	0.1662	0.2333	0.2058
needle				
Yes	0.2437	0.2437	0.2437	0.2437
	0.1786	0.1784	0.2107	0.1824
	1.36	1.37	1.16	1.34
	0.1725	0.1719	0.2475	0.1817
inject				
Yes	0.2233	0.2233	0.2233	0.2233
	0.1931	0.1933	0.2691	0.1961
	1.16	1.16	0.83	1.14
	0.2476	0.2480	0.4066	0.2548
jail				
Yes	0.4493	0.4493	0.4493	0.4493
	0.2217	0.2221	0.2986	0.2379
	2.03	2.02	1.50	1.89
	0.0427	0.0431	0.1324	0.0589

Legend: b/se/t/p

As expected, the regression coefficient estimates are the same for all VCE methods. The standard error estimates are fairly similar across all methods but a little more variable for `oim_adapt`. This is not surprising because the OIM method is based on the second-order derivatives, which are estimated numerically and thus require higher tolerances to produce more accurate estimates. For instance, we can specify a slightly lower tolerance (1e–7 instead of the default 1e–6) for the `oim` method using an adaptive step size,

```
. stintcox, vce(oim, tolerance(1e-7)) nohr
note: using adaptive step size to compute derivatives.

Computing standard errors: ..................................................
> ..... done

Interval-censored Cox regression                  Number of obs     =  1,124
Baseline hazard: Reduced intervals                       Uncensored =      0
                                                   Left-censored =     41
                                                  Right-censored =    991
                                                  Interval-cens. =     92

                                                  Wald chi2(5)      =  17.10
Log likelihood = -597.56443                       Prob > chi2       = 0.0043
```

	Coefficient	OIM std. err.	z	P>|z|	[95% conf.	interval]
age_mean	-.0320749	.012257	-2.62	0.009	-.0560982	-.0080516
male						
Yes	-.378782	.288278	-1.31	0.189	-.9437965	.1862326
needle						
Yes	.2436616	.1848383	1.32	0.187	-.1186149	.605938
inject						
Yes	.2232666	.1981754	1.13	0.260	-.1651501	.6116832
jail						
Yes	.4493186	.2299995	1.95	0.051	-.0014721	.9001094

```
Note: Standard-error estimates may be more variable for small datasets and
      datasets with low proportions of interval-censored observations.
```

and the standard errors are now closer to those of the other methods.

Current status or case I interval-censored data

Current status data or case I interval-censored data arise when each study subject is examined only once, such that the event of interest is known to occur before or after the examination time, resulting in a left- or right-censored observation. These data can be viewed as a special case of interval-censored data without truly interval-censored observations. The `stintcox` command requires that the current status data be recorded in the same format as the general interval-censored data, that is, by two time variables, which contain the lower and upper endpoints of an event-time interval and identify which observations are left-censored and which observations are right-censored.

▷ Example 5: Case I interval-censoring

Sun (2006) investigated a dataset about calcification of the hydrogel intraocular lenses (IOL), a rarely reported complication of cataract treatment. The dataset contains 379 patients who had IOL implantation and were examined by an ophthalmologist. The `status` variable indicates the degree of severity of IOL calcification: 0 means no or little calcification, and 1 means mild or serious calcification. Also, the study contains 237 females and 142 males. We want to test whether the IOL calcification is different between males and females.

Current status data usually contain two variables: one that records the examination time and one that records the status of the event of interest. `stintcox` requires that data be recorded in the interval-censored format. In this example, we will use the version of the dataset that was already converted to this format. See example 3 of [ST] **stintreg** on how to convert the current status data to the interval-censored format.

Let us fit a Cox proportional hazards model on `gender`:

```
. use https://www.stata-press.com/data/r17/iol
(Hydrogel Intraocular Lenses (IOL) Study)
. stintcox i.gender, interval(ltime rtime) nohr
note: using adaptive step size to compute derivatives.
Performing EM optimization (showing every 100 iterations):
Iteration 0:    log likelihood =  -191.3847
Iteration 100:  log likelihood = -137.70653
Iteration 200:  log likelihood = -137.65739
Iteration 300:  log likelihood = -137.64799
Iteration 400:  log likelihood = -137.64509
Iteration 425:  log likelihood = -137.64472
Computing standard errors: ..... done
Interval-censored Cox regression                  Number of obs     =        379
Baseline hazard: Reduced intervals                       Uncensored =          0
                                                   Left-censored =         48
                                                  Right-censored =        331
                                                  Interval-cens. =          0
                                                  Wald chi2(1)      =       0.52
Log likelihood = -137.64472                       Prob > chi2       =     0.4719
```

	OPG Coefficient	std. err.	z	P>|z|	[95% conf.	interval]
gender Male	-.2242553	.3117387	-0.72	0.472	-.835252	.3867415

```
Note: Standard-error estimates may be more variable for small datasets and
      datasets with low proportions of interval-censored observations.
```

When the `nohr` option is specified, the coefficient table displays the regression coefficient estimates instead of the hazard ratios for the independent variables. The above table reports $\widehat{\beta} = -0.2243$ and its estimated standard error of 0.312. This yields a test of $\beta = 0$ with a p-value of 0.47, which suggests that there is no difference between males and females in terms of the time to IOL calcification.

Sun (2006) reports $\widehat{\beta} = -0.2241$ with its standard error of 0.295, which are obtained by direct maximum-likelihood optimization. Our EM-based estimates are comparable.

◁

Stored results

stintcox stores the following in e():

Scalars

e(N)	number of observations
e(N_unc)	number of uncensored observations
e(N_lc)	number of left-censored observations
e(N_rc)	number of right-censored observations
e(N_int)	number of interval-censored observations
e(k)	number of parameters
e(k_eq)	number of equations in e(b)
e(k_eq_model)	number of equations in overall model test
e(k_dv)	number of dependent variables
e(df_m)	model degrees of freedom
e(ll)	log likelihood
e(ll_0)	log likelihood, constant-only model
e(chi2)	χ^2
e(p)	p-value for model test
e(rank)	rank of e(V)
e(ic)	number of iterations
e(converged)	1 if converged, 0 otherwise
e(delta)	multiplier used with fixed step size
e(emiterate)	maximum EM iterations
e(emtolerance)	EM coefficient tolerance
e(emltolerance)	EM log-likelihood tolerance
e(emhsgtolerance)	EM scaled-gradient tolerance
e(noemhsgtolerance)	1 if noemhsgtolerance, 0, otherwise
e(vceiterate)	maximum VCE iterations
e(vcetolerance)	VCE log-likelihood tolerance

Macros

e(cmd)	stintcox
e(cmdline)	command as typed
e(depvar)	names of time interval variables specified in interval()
e(title)	title in estimation output
e(title2)	secondary title in estimation output
e(chi2type)	Wald or LR; type of model χ^2 test
e(vce)	*vcetype* specified in vce()
e(vcetype)	title used to label Std. err.
e(properties)	b V
e(predict)	program used to implement predict
e(marginsok)	predictions allowed by margins
e(marginsnotok)	predictions disallowed by margins
e(strata)	strata variables
e(intervals)	reduced or full
e(filename)	name of the file with estimated baseline hazard contributions
e(stepsize)	adaptive or fixed

Matrices

e(b)	coefficient vector
e(V)	variance–covariance matrix of the estimators

Functions

e(sample)	marks estimation sample

In addition to the above, the following is stored in r():

Matrices

r(table)	matrix containing the coefficients with their standard errors, test statistics, p-values, and confidence intervals

Note that results stored in r() are updated when the command is replayed and will be replaced when any r-class command is run after the estimation command.

Methods and formulas

Methods and formulas are presented under the following headings:

Data and model

For a comprehensive review of the methods in this entry, see Zeng, Mao, and Lin (2016).

Let T denote the event or failure time, and let $\mathbf{x}$ denote a $1 \times p$ vector of covariates. Under the Cox proportional hazards model, the hazard function of T conditional on $\mathbf{x}$ is

$$h(t;\mathbf{x}) = h_0(t)\exp(\mathbf{x}\boldsymbol{\beta})$$

where $\boldsymbol{\beta}$ is a $p \times 1$ vector of unknown regression parameters and $h_0(t)$ is an arbitrary baseline hazard function. Let $H_0(t) = \int_0^t h_0(s)ds$ be the baseline cumulative hazard function.

The occurrence of an asymptomatic event can be detected only through periodic examinations. Let $(T_l, T_u]$ denote the shortest time interval that brackets T, with $T_l < T_u$. Left-censoring is indicated by $T_l = 0$, and right-censoring by $T_u = \infty$.

For a study with n subjects, the observed data consist of $(t_{li}, t_{ui}, \mathbf{x}_i)$ for $i = 1, \ldots, n$, where t_{li} and t_{ui} define the observed time interval and $\mathbf{x}_i$ records covariate values for a subject i.

The observed-data likelihood function for $\boldsymbol{\beta}$ and $H_0(t)$ is

$$L_n(\boldsymbol{\beta}, H_0) = \prod_{i=1}^{n}\Big[\exp\Big\{-\int_0^{t_{li}}\exp(\mathbf{x}_i\boldsymbol{\beta})dH_0(s)\Big\} - \exp\Big\{-\int_0^{t_{ui}}\exp(\mathbf{x}_i\boldsymbol{\beta})dH_0(s)\Big\}\Big]$$

where the integral is ∞ if $t_{ui} = \infty$. Under the NPMLE approach, H_0 is regarded as a step function with nonnegative jumps $h_1, \ldots, h_m$ at $t_1, \ldots, t_m$, respectively, where $t_1 < \cdots < t_m$ are the distinct time points for all $t_{li} > 0$ and $t_{ui} < \infty$ for $i = 1, \ldots, n$. Thus, we need to maximize the function

$$L_n(\boldsymbol{\beta}, \{h_k\}) = \prod_{i=1}^{n}\exp\Big\{-\sum_{t_k \le t_{li}} h_k\exp(\mathbf{x}_i\boldsymbol{\beta})\Big\}\Big[1 - \exp\Big\{-\sum_{t_{li} < t_k \le t_{ui}} h_k\exp(\mathbf{x}_i\boldsymbol{\beta})\Big\}\Big]^{I(t_{ui}<\infty)} \quad (1)$$

Direct maximization of (1) is difficult because of the lack of an analytic expression for the parameters h_k $(k = 1, \ldots, m)$. And an even greater challenge is that not all t_{li} and t_{ui} are informative about the event times, so many h_k's are zeros and thus lie on the boundary of the parameter space.

To address these challenges, Zeng, Mao, and Lin (2016) construct some latent Poisson variables that yield the same observed-data likelihood as (1). They propose the EM algorithm, in which the E-step involves simple calculations, and the M-step amounts to the maximization of a weighted sum of Poisson log-likelihood functions that is strictly concave and has a closed-form solution for h_k's.

Turnbull (1976) showed that the NPMLE of the survival distribution is unique only up to a set of intervals, which is called Turnbull's intervals (the innermost intervals or the regions of the maximal cliques). Therefore, for computational reasons, by default or if the `reduced` option is specified, we estimate the baseline cumulative hazard function at the endpoints of Turnbull's reduced set of intervals. Alternatively, you can specify the `full` option to estimate the baseline cumulative hazard function at the endpoints of all observed time intervals, which corresponds to the approach of Zeng, Mao, and Lin (2016).

EM algorithm for computing parameter estimates

Let W_{ik} $(i = 1, \ldots, n; k = 1, \ldots, m)$ be independent latent Poisson random variables with means $h_k \exp(\mathbf{x}_i\boldsymbol{\beta})$. Define $A_i = \sum_{t_k \le t_{li}} W_{ik}$ and $B_i = I(t_{ui} < \infty) \sum_{t_{li} < t_k \le t_{ui}} W_{ik}$. The likelihood for the observed data $(t_{li}, t_{ui}, \mathbf{x}_i, A_i = 0, B_i > 0)$ is

$$\prod_{i=1}^{n} \prod_{t_k \le t_{li}} \Pr(W_{ik} = 0) \Big\{ 1 - \Pr\Big(\sum_{t_{li} < t_k \le t_{ui}} W_{ik} = 0 \Big) \Big\}^{I(t_{ui} < \infty)} \tag{2}$$

which is exactly equal to (1). Thus, we can maximize (2) through an EM algorithm treating W_{ik} as missing data.

The complete-data log likelihood is

$$\sum_{i=1}^{n} \sum_{k=1}^{m} I(t_k \le t^*_{ui}) \Big[W_{ik} \log\big\{ h_k \exp(\mathbf{x}_i\boldsymbol{\beta}) \big\} - h_k \exp(\mathbf{x}_i\boldsymbol{\beta}) - \log W_{ik}! \Big]$$

where $t^*_{ui} = I(t_{ui} < \infty) t_{ui} + I(t_{ui} = \infty) t_{li}$. In the E-step, we evaluate the posterior means of W_{ik} as

$$w_{ik} = \begin{cases} \dfrac{h_k \exp(\mathbf{x}_i\boldsymbol{\beta})}{1 - \exp\big(- \sum_{t_{li} < t_j \le t_{ui}} h_j \exp(\mathbf{x}_i\boldsymbol{\beta}) \big)} & \text{if } t_{li} < t_k \le t_{ui} < \infty \\ 0 & \text{otherwise} \end{cases}$$

In the M-step, we update $\boldsymbol{\beta}$ by solving the following equation via the one-step Newton–Raphson method,

$$\sum_{i=1}^{n} \sum_{k=1}^{m} I(t_k \le t^*_{ui})\, w_{ik} \Big\{ \mathbf{x}_i - \frac{\sum_{j=1}^{n} I(t_k \le t^*_{uj}) \exp(\mathbf{x}_j\boldsymbol{\beta}) \mathbf{x}_j}{\sum_{j=1}^{n} I(t_k \le t^*_{uj}) \exp(\mathbf{x}_j\boldsymbol{\beta})} \Big\} = 0$$

and then update h_k $(k = 1, \ldots, m)$ using

$$h_k = \frac{\sum_{i=1}^{n} I(t_k \le t^*_{ui}) w_i}{\sum_{i=1}^{n} I(t_k \le t^*_{ui}) \exp(\mathbf{x}_i\boldsymbol{\beta})} \tag{3}$$

Setting the initial values of $\boldsymbol{\beta}$ to zeros and the initial values of h_k's to $1/m$, we iterate between the E- and M-steps until the desired convergence criteria are achieved. The convergence tolerances are described in detail in [R] **Maximize**, where `emtolerance()` is analogous to `tolerance()`, `emltolerance()` to `ltolerance()`, `hsgtolerance()` to `nrtolerance()`, and `nonrtolerance` to `nohsgtolerance`.

The observed-data log-likelihood function is calculated as

$$\log L = \sum_{i=1}^{n} \log \Big\{ I(t_{li} < t_{ui})(S_{li} - S_{ui}) + I(t_{li} = t_{ui}) \sum_{k=1}^{m} I(t_{li} = t_k)\, h_k \exp(\mathbf{x}_i\boldsymbol{\beta}) S_{li} \Big\}$$

where $S_{li} = \exp\big\{ - \sum_{t_k \le t_{li}} h_k \exp(\mathbf{x}_i\boldsymbol{\beta}) \big\}$ and $S_{ui} = \exp\big\{ - \sum_{t_k \le t_{ui}} h_k \exp(\mathbf{x}_i\boldsymbol{\beta}) \big\} I(t_{ui} < \infty)$.

When there are no covariates, the above algorithm becomes iteratively updating h_k as

$$h_k = \frac{\sum_{i=1}^{n} \big\{ h_k I(t_{li} < t_k \le t_{ui} < \infty) S_{li} / (S_{li} - S_{ui}) + I(t_{li} = t_{ui} = t_k) \big\}}{\sum_{i=1}^{n} I(t_k \le t^*_{ui})}$$

where $S_{li} = \exp\big\{ - \sum_{t_k \le t_{li}} h_k \big\}$ and $S_{ui} = \exp\big\{ - \sum_{t_k \le t_{ui}} h_k \big\} I(t_{ui} < \infty)$.

Variance estimation using profile log-likelihood function

Denote the estimators of $\boldsymbol{\beta}$ and h_k's by $\widehat{\boldsymbol{\beta}}$ and $\widehat{h}_k$'s, respectively, for $k = 1, \ldots, m$. The NPMLEs $\widehat{\boldsymbol{\beta}}$ and $\widehat{H}_0(t) = \sum_{t_k \le t} \widehat{h}_k$ are strongly consistent, and $\widehat{\boldsymbol{\beta}}$ is asymptotically normal and asymptotically efficient (Zeng, Mao, and Lin 2016). The covariance matrix of $\widehat{\boldsymbol{\beta}}$ can be estimated using profile likelihood (Murphy and van der Vaart 2000; Zeng, Mao, and Lin 2016; Zeng, Gao, and Lin 2017).

The profile log-likelihood function for $\boldsymbol{\beta}$ takes the form

$$pl_n(\boldsymbol{\beta}) = \sum_{i=1}^{n} \log\Big[\exp\Big\{ -\sum_{t_k \le t_{li}} \widetilde{h}_k \exp(\mathbf{x}_i\boldsymbol{\beta}) \Big\} - I(t_{iu} < \infty) \exp\Big\{ -\sum_{t_k \le t_{ui}} \widetilde{h}_k \exp(\mathbf{x}_i\boldsymbol{\beta}) \Big\} \Big]$$

where $\widetilde{h}_k$ $(k = 1, \ldots, m)$ are the maximizers of (1) for the given $\boldsymbol{\beta}$. These maximizers are obtained from the EM algorithm described in the previous section with $\widehat{h}_k$ $(k = 1, \ldots, m)$ as the initial values and with $\boldsymbol{\beta}$ fixed over the iterations. Specifically, we apply the same EM algorithm but hold $\boldsymbol{\beta}$ fixed during the iterations. Thus, the only steps in the EM algorithm are to explicitly evaluate w_{ik} and to update h_k using (3).

Two likelihood-based methods, `vce(opg)` and `vce(oim)`, are available to estimate the covariance matrix of $\widehat{\boldsymbol{\beta}}$.

The `oim` method estimates the covariance matrix of $\widehat{\boldsymbol{\beta}}$ by the negative inverse of the Hessian matrix whose (j, k)th element is

$$\frac{pl_n(\widehat{\boldsymbol{\beta}}) - pl_n(\widehat{\boldsymbol{\beta}} + \delta_n\mathbf{e}_k) - pl_n(\widehat{\boldsymbol{\beta}} + \delta_n\mathbf{e}_j) + pl_n(\widehat{\boldsymbol{\beta}} + \delta_n\mathbf{e}_k + \delta_n\mathbf{e}_j)}{\delta_n^2}$$

where $\mathbf{e}_j$ is the jth canonical vector in the space of $\boldsymbol{\beta}$ and δ_n is the step size chosen for numerical difference. Zeng, Gao, and Lin (2017) found that the above estimated matrix may be negative definite, especially in small samples. Therefore, they proposed the `opg` method, which estimates the covariance matrix using the inverse of the sum of the OPG vectors whose (j, k)th element is

$$\sum_{i=1}^{n} \frac{\{pl_{ni}(\widehat{\boldsymbol{\beta}} + \delta_n\mathbf{e}_j) - pl_{ni}(\widehat{\boldsymbol{\beta}})\}\{pl_{ni}(\widehat{\boldsymbol{\beta}} + \delta_n\mathbf{e}_k) - pl_{ni}(\widehat{\boldsymbol{\beta}})\}}{\delta_n^2}$$

where $pl_{ni}(\boldsymbol{\beta})$ is the subject i contribution to $pl_n(\boldsymbol{\beta})$. The resulting covariance matrix estimator is guaranteed to be positive semidefinite and more robust with respect to the choice of the step size than the estimator based on the second-order numerical difference.

The `stepsize()` suboption of the `vce()` option offers different ways for choosing the step size with the `opg` and `oim` methods when computing numerical derivatives. `stepsize(adaptive)`, the default, uses an adaptive step size; see [M-5] **deriv( )**. `stepsize(fixed)` uses a fixed step size equal to $\delta_n = 5n^{-1/2}$ (Zeng, Mao, and Lin 2016; Zeng, Gao, and Lin 2017). And `stepsize(fixed #)` uses a fixed step size equal to $\# \times n^{-1/2}$.

Stratified estimation

`stintcox` with the `strata()` option will produce a stratified proportional hazards model for interval-censored data. Assume that within stratum q ($q = 1, \ldots, Q$), the hazard function of T is

$$h_q(t; \mathbf{x}) = h_{q0}(t) \exp(\mathbf{x}\boldsymbol{\beta})$$

where $h_{q0}(t)$ is an unknown baseline hazard function for stratum q.

Let S_i be the stratum for subject i, which can take value from $\{1, \ldots, Q\}$. Analogously to (1), we maximize the function

$$L_n(\boldsymbol{\beta}, \{h_{qk}\}) = \prod_{i=1}^{n} \prod_{q=1}^{Q} \exp\Big\{ - \sum_{t_{qk} \le t_{li}} h_{qk} \exp(\mathbf{x}_i\boldsymbol{\beta}) \Big\} \Big[1 - \exp\Big\{ - \sum_{t_{li} < t_{qk} \le t_{ui}} h_{qk} \exp(\mathbf{x}_i\boldsymbol{\beta}) \Big\} \Big]^{I(t_{ui} < \infty)}$$

where, within the NPMLE framework, a stratum-specific baseline cumulative hazard function $H_{q0}(t) = \int_0^t h_{q0}(s)\, ds$ for $q = 1, \ldots, Q$ is regarded as a step function, $\sum_{t_{qk} \le t} h_{qk}$, with nonnegative jumps h_{qk} at the distinct values t_{qk}, ordered from smallest to largest, of the observed interval endpoints from all subjects in stratum q.

Acknowledgments

The development of `stintcox` was partially supported by SBIR awards 1R43CA233159-01 and 2R44CA233159-02. We also thank our consultants Danyu Lin and Donglin Zeng of the University of North Carolina at Chapel Hill for their contributions to the command.

References

Cai, T., and R. A. Betensky. 2003. Hazard regression for interval-censored data with penalized spline. *Biometrics* 59: 570–579. https://www.jstor.org/stable/3695433.

Cox, D. R. 1972. Regression models and life-tables (with discussion). *Journal of the Royal Statistical Society, Series B* 34: 187–220. https://doi.org/10.1111/j.2517-6161.1972.tb00899.x.

——. 1975. Partial likelihood. *Biometrika* 62: 269–276. https://doi.org/10.1093/biomet/62.2.269.

Finkelstein, D. M. 1986. A proportional hazards model for interval-censored failure time data. *Biometrics* 42: 845–854. https://doi.org/10.2307/2530698.

Finkelstein, D. M., and R. A. Wolfe. 1985. A semiparametric model for regression analysis of interval-censored failure time data. *Biometrics* 41: 933–945. https://doi.org/10.2307/2530965.

Huang, J., and J. A. Wellner. 1997. Interval censored survival data: A review of recent Progress. In *Proceedings of the First Seattle Symposium in Biostatistics: Survival Analysis*, ed. D. Y. Lin and T. R. Fleming, 123–169. New York: Springer.

Lindsey, J. C., and L. M. Ryan. 1998. Methods for interval-censored data. *Statistics in Medicine* 17: 219–238. https://doi.org/10.1002/(SICI)1097-0258(19980130)17:2⟨219::AID-SIM735⟩3.0.CO;2-O.

Lindsey, J. K. 1998. A study of interval censoring in parametric regression models. *Lifetime Data Analysis* 4: 329–354. https://doi.org/10.1023/A:1009681919084.

Murphy, S. A., and A. W. van der Vaart. 2000. On profile likelihood. *Journal of the American Statistical Association* 95: 449–465. https://doi.org/10.2307/2669386.

Odell, P. M., K. M. Anderson, and R. B. D'Agostino. 1992. Maximum likelihood estimation for interval-censored data using a Weibull-based accelerated failure time model. *Biometrics* 48: 951–959. https://doi.org/10.2307/2532360.

Rabinowitz, D., A. A. Tsiatis, and J. Aragon. 1995. Regression with interval-censored data. *Biometrika* 82: 501–513. https://doi.org/10.2307/2337529.

Sun, J. 2006. *The Statistical Analysis of Interval-Censored Failure Time Data*. New York: Springer.

Sun, J., and J. Li. 2014. Interval censoring. In *Handbook of Survival Analysis*, ed. J. P. Klein, H. C. van Houwelingen, J. G. Ibrahim, and T. H. Scheike, 369–390. Boca Raton, FL: CRC Press.

Turnbull, B. W. 1976. The empirical distribution function with arbitrarily grouped, censored and truncated data. *Journal of the Royal Statistical Society, Series B* 38: 290–295. https://doi.org/10.1111/j.2517-6161.1976.tb01597.x.

Wang, L., C. S. McMahan, M. G. Hudgens, and Z. P. Qureshi. 2016. A flexible, computationally efficient method for fitting the proportional hazards model to interval-censored data. *Biometrics* 72: 222–231. https://doi.org/10.1111/biom.12389.

Zeng, D., F. Gao, and D. Y. Lin. 2017. Maximum likelihood estimation for semiparametric regression models with multivariate interval-censored data. *Biometrika* 104: 505–525. https://doi.org/10.1093/biomet/asx029.

Zeng, D., L. Mao, and D. Y. Lin. 2016. Maximum likelihood estimation for semiparametric transformation models with interval-censored data. *Biometrika* 103: 253–271. https://doi.org/10.1093/biomet/asw013.

Zhang, Y., L. Hua, and J. Huang. 2010. A splinebased semiparametric maximum likelihood estimation method for the Cox model with intervalcensored data. *Scandinavian Journal of Statistics* 37: 338–354. https://doi.org/10.1111/j.1467-9469.2009.00680.x.

Also see

[ST] **stintcox postestimation** — Postestimation tools for stintcox

[ST] **stintcox PH-assumption plots** — Plots of proportional-hazards assumption after stintcox

[ST] **stcurve** — Plot the survivor or related function after streg, stcox, and others

[ST] **stcox** — Cox proportional hazards model

[ST] **stintreg** — Parametric models for interval-censored survival-time data

[U] **20 Estimation and postestimation commands**