

Title

stcox — Fit Cox proportional hazards model

Syntax

```
stcox [varlist] [if] [in] [, options]
```

<i>options</i>	description
Model	
<u>estimate</u>	fit model without covariates
<u>strata</u> (varnames)	strata ID variables
<u>shared</u> (varname)	shared-frailty ID variable
<u>offset</u> (varname)	include <i>varname</i> in model with coefficient constrained to 1
<u>breslow</u>	use Breslow method to handle tied failures; the default
<u>efron</u>	use Efron method to handle tied failures
<u>exactm</u>	use exact marginal-likelihood method to handle tied failures
<u>exactp</u>	use exact partial-likelihood method to handle tied failures
Time varying	
<u>tv</u> c(varlist)	time-varying covariates
<u>te</u> xp(<i>exp</i>)	multiplier for time-varying covariates; default is <code>te</code> xp(<code>_t</code>)
SE/Robust	
<u>v</u> ce(<i>vcetype</i>)	<i>vcetype</i> may be <code>oim</code> , <code>robust</code> , <code>cluster</code> <i>clustvar</i> , <code>bootstrap</code> , or <code>jackknife</code>
<u>n</u> oadjust	do not use standard degree-of-freedom adjustment
Reporting	
<u>l</u> evel(#)	set confidence level; default is <code>level(95)</code>
<u>n</u> ohr	report coefficients, not hazard ratios
<u>n</u> oshow	do not show st setting information
Reporting 2	
<u>m</u> gale(<i>newvar</i>)	create <i>newvar</i> containing partial martingale residuals
<u>b</u> asechazard(<i>newvar</i>)	create <i>newvar</i> containing cumulative baseline hazard
<u>b</u> asehc(<i>newvar</i>)	create <i>newvar</i> containing baseline hazard contributions
<u>b</u> asesurv(<i>newvar</i>)	create <i>newvar</i> containing baseline survivor function
<u>e</u> ffects(<i>newvar</i>)	create <i>newvar</i> containing estimated log-frailties
<u>e</u> sr(<i>newvars</i>)	create <i>newvars</i> containing partial efficient score residuals
<u>s</u> choenfeld(<i>newvars</i>)	create <i>newvars</i> containing Schoenfeld residuals
<u>s</u> caledsch(<i>newvars</i>)	create <i>newvars</i> containing scaled Schoenfeld residuals
Max options	
<u>m</u> aximize_options	control the maximization process; seldom used

You must `stset` your data before using `stcox`; see [ST] `stset`.

`bootstrap`, `by`, `jackknife`, `nestreg`, `statsby`, `stepwise`, `svy`, and `xi` are allowed; see [U] 11.1.10 **Prefix commands**.

`estimate`, `shared()`, `efron`, `exactm`, `exactp`, `vce()`, and `noadjust` are not allowed with the `svy` prefix.

`fweights`, `iweights`, and `pweights` may be specified using `stset`; see [ST] `stset`. However, weights may not be specified if you are using the `bootstrap` prefix with the `stcox` command.

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

Description

`stcox` fits, via maximum likelihood, proportional hazards models on st data. `stcox` can be used with single- or multiple-record or single- or multiple-failure st data.

Options for stcox

Model

`estimate` forces fitting of the null model. All Stata estimation commands redisplay results when the command name is typed without arguments. So does `stcox`. What if you wish to fit a Cox model on $x_j \mathbf{b}$, where $x_j \mathbf{b}$ is defined as 0? Logic says that you would type `stcox`. There are no explanatory variables, so there is nothing to type after the command. Unfortunately, `stcox` looks the same as `stcox` typed without arguments, which is a request to redisplay results.

To fit the null model, type `stcox`, `estimate`.

`strata(varnames)` specifies up to five strata variables. Observations with equal values of the strata variables are assumed to be in the same stratum. Stratified estimates (equal coefficients across strata but with baseline hazard unique to each stratum) are then obtained.

`shared(varname)` specifies that a Cox model with shared frailty be fitted. Observations with equal value of `varname` are assumed to have shared (the same) frailty. Across groups, the frailties are assumed to be gamma-distributed latent random effects that affect the hazard multiplicatively, or, equivalently, the logarithm of the frailty enters the linear predictor as a random offset. Think of a shared-frailty model as a Cox model for panel data. `varname` is a variable in the data that identifies the groups.

Shared-frailty models are discussed more in *Cox regression with shared frailty*.

`offset(varname)`; see [ST] **estimation options**.

`breslow`, `efron`, `exactm`, and `exactp` specify the method for handling tied failures in the calculation of the log partial likelihood (and residuals). `breslow` is the default. Each method is described in the *Methods and Formulas* section below. `efron` and the exact methods require substantially more computer time than the default `breslow` option. `exactm` and `exactp` may not be specified with `tvc()`, `vce(robust)`, or `vce(cluster clustvar)`.

Time varying

`tvc(varlist)` specifies those variables that vary continuously with respect to time, i.e., time-varying covariates. This is a convenience option used to speed up calculations and to avoid having to `stsplit` the data over many failure times.

`texp(exp)` is used in conjunction with `tvc(varlist)` to specify the function of analysis time that should be multiplied by the time-varying covariates. For example, specifying `texp(ln(_t))` would cause the time-varying covariates to be multiplied by the logarithm of analysis time. If `tvc(varlist)` is used without `texp(exp)`, Stata understands that you mean `texp(_t)` and thus multiplies the time-varying covariates by the analysis time.

Both `tvc(varlist)` and `texp(exp)` are explained more in the section on *Cox regression with continuously time-varying covariates* below.

SE/Robust

`vce(vcetype)` specifies the type of standard error reported, which includes types that are derived from asymptotic theory, that are robust to some kinds of misspecification, that allow for intragroup correlation, and that use bootstrap or jackknife methods; see [R] *vce_option*.

`noadjust` is for use with `vce(robust)` or `vce(cluster clustvar)`. `noadjust` prevents the estimated variance matrix from being multiplied by $N/(N - 1)$ or $g/(g - 1)$, where g is the number of clusters. The default adjustment is somewhat arbitrary because it is not always clear how to count observations or clusters. In such cases, however, the adjustment is likely to be biased toward 1, so we would still recommend making it.

Reporting

`level(#)`; see [ST] **estimation options**.

`nohr` specifies that coefficients be displayed rather than exponentiated 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 redisplaying previously estimated results (which you do by typing `stcox` without a variable list).

`noshow` prevents `stcox` from showing the key `st` variables. This option is seldom used since most people type `stset`, `show` or `stset`, `noshow` to set whether they want to see these variables mentioned at the top of the output of every `st` command; see [ST] **stset**.

Reporting 2

`mgale(newvar)` adds *newvar* containing the partial martingale residuals, which are fully described in *Methods and Formulas*. If each observation in your data represents a different subject (single-record-per-subject data), the partial martingale residuals are the martingale residuals.

If you have data with multiple records per subject, the value that `mgale()` stores in each observation is the observation's contribution to the martingale residual, and these partial residuals can be summed within `id()` to obtain the subject's martingale residual. Say that you specify `mgale(pmr)` and that you have previously `stset`, `id(patid)`. Then `egen mr = total(pmr), by(patid)` would create the martingale residuals.

`basechazard(newvar)` adds *newvar* to the data containing the estimated cumulative baseline hazard. If `strata()` is also specified, cumulative baseline estimates for each stratum are provided.

`basehc(newvar)` adds *newvar* to the data containing the estimated baseline hazard contributions as described by Kalbfleisch and Prentice (2002, 115). These are used to construct the product-limit type estimator for the baseline survivor function generated by `basesurv()`. If `strata()` is also specified, baseline estimates for each stratum are provided.

`basesurv(newvar)` adds *newvar* to the data containing the estimated baseline survivor function. In the null model, this is equivalent to the Kaplan–Meier product-limit estimate. If `strata()` is also specified, baseline estimates for each stratum are provided.

`effects(newvar)` is for use with `shared()` and creates *newvar* containing estimates of the log-frailty for each group. The log-frailties are random group-specific offsets to the linear predictor that measure the group effect on the log-relative hazard.

`esr(newvars)` adds *newvars* containing the partial efficient score residuals; see *Methods and Formulas*. If each observation in your data represents a different subject (single-record-per-subject data), the partial efficient score residuals are the efficient score residuals.

If you have data with multiple records per subject, the values `esr()` stores are each observation's contribution to the score residuals, and these partial residuals can be summed within `id()` to form the subject's efficient score residuals. This could be accomplished as noted under `mgale()` above.

One efficient score residual variable is created for each regressor in the model; the first new variable corresponds to the first regressor, the second to the second, and so on.

`schoenfeld(newvars)` adds *newvars* containing the Schoenfeld residuals; see *Methods and Formulas*. This option may not be used with the `exactm` and `exactp` options. Schoenfeld residuals are calculated and reported only at failure times.

One Schoenfeld residual variable is created for each regressor in the model; the first new variable corresponds to the first regressor, the second to the second, and so on.

`scaledsch(newvars)` adds *newvars* containing the scaled Schoenfeld residuals; see *Methods and Formulas*. This option may not be used with the `exactm` or `exactp` options. Scaled Schoenfeld residuals are calculated and reported only at failure times.

One scaled Schoenfeld residual variable is created for each regressor in the model; the first new variable corresponds to the first regressor, the second to the second, and so on.

Note: the easiest way to specify the preceding three options is, for example, `esr(stub*)`, where *stub* is a short name of your choosing. Stata then creates variables *stub1*, *stub2*, etc. You may also specify each variable explicitly, in which case there must be as many (and no more) variables specified in `esr()` as regressors in the model.

However, be aware that `stcox` will drop variables from the model because of collinearity. This is a desirable feature. A side effect is that the score residual variable, the Schoenfeld residual variable, and the scaled Schoenfeld residual variable may not align with the regressors in the way you expect. Say that you fit a model by typing

```
. stcox x1 x2 x3, esr(r1 r2 r3)
```

Usually, `r1` will contain the residual associated with `x1`, `r2` the residual associated with `x2`, etc.

Now assume that `x2` is dropped because of collinearity. Then `r1` will correspond to `x1`, `r2` to `x3`, and `r3` will contain 0. This happens because, after the collinear variables are omitted, there are only two variables in the model: `x1` and `x3`.

Max options

maximize_options: `iterate(#)`, `[no]log`, `trace`, `tolerance(#)`, `ltolerance(#)`, see [R] **maximize**. These options are seldom used.

(Continued on next page)

Remarks

Remarks are presented under the following headings:

Cox regression with uncensored data
Cox regression with censored data
Treatment of tied failure times
Cox regression with discrete time-varying covariates
Cox regression with continuous time-varying covariates
Robust estimate of variance
Cox regression with multiple-failure data
Stratified estimation
Cox regression with shared frailty
Obtaining baseline function estimates

What follows is a summary of what can be done with `stcox`. For a complete tutorial, see Cleves, Gould, and Gutierrez (2004), which devotes three chapters to this topic.

In the Cox proportional hazards model, the hazard is assumed to be

$$h(t) = h_0(t) \exp(\beta_1 x_1 + \cdots + \beta_k x_k)$$

The Cox model provides estimates of β_1, \dots, β_k but provides no direct estimate of $h_0(t)$ —the baseline hazard. Formally, the function $h_0(t)$ is not directly estimated, but it is possible to recover an estimate of the cumulative hazard $H_0(t)$ and, from that, an estimate of the baseline survivor function $S_0(t)$.

`stcox` fits the Cox proportional hazards model; that is, it provides estimates of β and its variance–covariance matrix.

`stcox, basehc(newvar)` fits the Cox proportional hazards model and calculates the set of baseline hazard contributions used to estimate the baseline survivor function $S_0(t)$.

`stcox, basechazard(newvar)` fits the Cox proportional hazards model and estimates the cumulative baseline hazard function $H_0(t)$.

`stcox, basesurv(newvar)` fits the Cox model and estimates the baseline survivor function $S_0(t)$.

The three baseline options may also be used in combination to concurrently produce estimates of their respective functions. Also, `stcox` with the `strata()` option will produce stratified Cox regression estimates. In the stratified estimator, the hazard at time t for a subject in group i is assumed to be

$$h_i(t) = h_{0i}(t) \exp(\beta_1 x_1 + \cdots + \beta_k x_k)$$

That is, the coefficients are assumed to be the same, regardless of group, but the baseline hazard can be group specific. If you specify the `strata()` option, the baseline options produce the group-specific estimates of the baseline functions.

Whether or not you specify `strata()`, `stcox` can produce either of two variance estimators for β . The default is to calculate the conventional, inverse matrix of negative second derivatives. The theoretical justification for this estimator is based on likelihood theory.

The `vce(robust)` option instead switches to the robust measure developed by Lin and Wei (1989). This variance estimator is a variant of the estimator discussed in [U] **20.15 Obtaining robust variance estimates**.

Finally, `stcox`, with the `shared()` option, fits a Cox model with shared frailty. A *frailty* is a group-specific latent random effect that multiplies into the hazard function. The distribution of the frailties is gamma with mean one and variance to be estimated by the data. Shared-frailty models are used to model within-group correlation (observations within a group are correlated because they share the same frailty).

We give examples below with uncensored, censored, time-varying, and recurring failure data, but it does not matter in terms of what you type. Once you have `stset` your data, to fit a model you type `stcox` followed by the names of the explanatory variables. You do this whether your dataset has single or multiple records, includes censored observations or delayed entry, or even has single or multiple failures. You use `stset` to describe the properties of the data, and then that information is available to `stcox`—and all the other `st` commands—so that you do not have to specify it again.

Cox regression with uncensored data

▷ Example 1

We wish to analyze an experiment testing the ability of emergency generators with a new-style bearing to withstand overloads. For this experiment, the overload protection circuit was disabled, and the generators were run overloaded until they burned up. Here are our data:

```
. use http://www.stata-press.com/data/r10/kva
(Generator experiment)
. list
```

	failtime	load	bearings
1.	100	15	0
2.	140	15	1
3.	97	20	0
4.	122	20	1
5.	84	25	0
6.	100	25	1
7.	54	30	0
8.	52	30	1
9.	40	35	0
10.	55	35	1
11.	22	40	0
12.	30	40	1

Twelve generators, half with the new-style bearings and half with the old, were allocated to this destructive test. The first observation reflects an old-style generator (`bearings = 0`) under a 15-kVA overload. It stopped functioning after 100 hours. The second generator had new-style bearings (`bearings = 1`) and, under the same overload condition, lasted 140 hours. Paired experiments were also performed under overloads of 20, 25, 30, 35, and 40 kVA.

We wish to fit a Cox proportional hazards model in which the failure rate depends on the amount of overload and the style of the bearings. That is, we assume that `bearings` and `load` do not affect the shape of the overall hazard function, but they do affect the relative risk of failure. To fit this model, we type

(Continued on next page)

```

. stset failtime
(output omitted)
. stcox load bearings
      failure _d: 1 (meaning all fail)
      analysis time _t: failtime
Iteration 0:  log likelihood = -20.274897
Iteration 1:  log likelihood = -10.515114
Iteration 2:  log likelihood = -8.8700259
Iteration 3:  log likelihood = -8.5915211
Iteration 4:  log likelihood = -8.5778991
Iteration 5:  log likelihood = -8.577853
Refining estimates:
Iteration 0:  log likelihood = -8.577853
Cox regression -- Breslow method for ties
No. of subjects =          12          Number of obs =          12
No. of failures =          12
Time at risk   =          896
Log likelihood = -8.577853          LR chi2(2)      =          23.39
                                          Prob > chi2    =          0.0000

```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
load	1.52647	.2188172	2.95	0.003	1.152576	2.021653
bearings	.0636433	.0746609	-2.35	0.019	.0063855	.6343223

We find that after controlling for overload, the new-style bearings result in a lower hazard and therefore a longer survivor time.

Once an `stcox` model has been fitted, typing `stcox` without arguments redisplay the previous results. Options that affect the display, such as `nohr`—which requests that coefficients rather than hazard ratios be displayed—can be specified upon estimation or when results are redisplayed:

```

. stcox, nohr
Cox regression -- Breslow method for ties
No. of subjects =          12          Number of obs =          12
No. of failures =          12
Time at risk   =          896
Log likelihood = -8.577853          LR chi2(2)      =          23.39
                                          Prob > chi2    =          0.0000

```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
load	.4229578	.1433485	2.95	0.003	.1419999	.7039157
bearings	-2.754461	1.173115	-2.35	0.019	-5.053723	-.4551981

◀

□ Technical Note

`stcox`'s iteration log looks like a standard Stata iteration log right up to the point where it says "Refining estimates". The Cox proportional hazards likelihood function is indeed a difficult function, both conceptually and numerically. Until Stata says "Refining estimates", it maximizes the Cox likelihood in the standard way using double-precision arithmetic. Then just to be sure that the answers are accurate, Stata switches to quad-precision routines (double double precision) and completes the maximization procedure from its current location on the likelihood.

□

Cox regression with censored data

▷ Example 2

We have data on 48 participants in a cancer drug trial. Of these 48, 28 receive treatment (`drug = 1`) and 20 receive a placebo (`drug = 0`). The participants range in age from 47 to 67 years. We wish to analyze time until death, measured in months. Our data include 1 observation for each patient. The variable `studytime` records either the month of their death or the last month that they were known to be alive. Some of the patients still live, so together with `studytime` is `died`, indicating their health status. Persons known to have died—“noncensored” in the jargon—have `died = 1`, whereas the patients who are still alive—“right-censored” in the jargon—have `died = 0`.

Here is an overview of our data:

```
. use http://www.stata-press.com/data/r10/drugtr
(Patient Survival in Drug Trial)
. st
-> stset studytime, failure(died)
      failure event:  died != 0 & died < .
obs. time interval:  (0, studytime]
exit on or before:   failure
. summarize
```

Variable	Obs	Mean	Std. Dev.	Min	Max
studytime	48	15.5	10.25629	1	39
died	48	.6458333	.4833211	0	1
drug	48	.5833333	.4982238	0	1
age	48	55.875	5.659205	47	67
_st	48	1	0	1	1
<hr/>					
_d	48	.6458333	.4833211	0	1
_t	48	15.5	10.25629	1	39
_t0	48	0	0	0	0

We typed `stset studytime, failure(died)` previously; that is how `st` knew about this dataset. To fit the Cox model, we type

```
. stcox drug age
      failure _d:  died
analysis time _t: studytime
Iteration 0:  log likelihood = -99.911448
Iteration 1:  log likelihood = -83.551879
Iteration 2:  log likelihood = -83.324009
Iteration 3:  log likelihood = -83.323546
Refining estimates:
Iteration 0:  log likelihood = -83.323546
Cox regression -- Breslow method for ties
No. of subjects =          48          Number of obs =          48
No. of failures =          31
Time at risk   =          744
Log likelihood = -83.323546          LR chi2(2)   =          33.18
                                          Prob > chi2 =          0.0000
```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
drug	.1048772	.0477017	-4.96	0.000	.0430057 .2557622
age	1.120325	.0417711	3.05	0.002	1.041375 1.20526

We find that the drug results in a lower hazard—and therefore a longer survivor time—controlling for age. Older patients are more likely to die. The model as a whole is statistically significant.

The hazard ratios reported correspond to a one-unit change in the corresponding variable. It is more typical to report relative risk for 5-year changes in age. To obtain such a hazard ratio, we create a new age variable such that a one-unit change indicates a 5-year change:

```
. replace age = age/5
age was int now float
(48 real changes made)

. stcox drug age, nolog
      failure _d:  died
      analysis time _t:  studytime

Cox regression -- Breslow method for ties

No. of subjects =          48          Number of obs =          48
No. of failures =          31
Time at risk   =          744

Log likelihood = -83.323544          LR chi2(2) =          33.18
                                          Prob > chi2 =          0.0000
```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
drug	.1048772	.0477017	-4.96	0.000	.0430057	.2557622
age	1.764898	.3290196	3.05	0.002	1.224715	2.543338

◀

Treatment of tied failure times

The proportional hazards model assumes that the hazard function is continuous and, thus, that there are no tied survival times. Because of the way that time is recorded, however, tied events do occur in survival data. In such cases, the partial likelihood must be modified. See *Methods and Formulas* for more details on the methods described below.

Stata provides four methods for handling tied failures in calculating the Cox partial likelihood through the options `breslow`, `efron`, `exactm`, and `exactp`. If there are no ties in the data, the results are identical, regardless of the method selected.

When there are tied failure times, we must decide how to calculate the risk pools for these tied observations. Assume that there are 2 observations that fail in succession. In the calculation involving the second observation, the first observation is not in the risk pool since failure has already occurred. If the 2 observations have the same failure time, we must decide how to calculate the risk pool for the second observation and in which order to calculate the two observations.

There are two views of time. In the first, time is continuous, so ties should not occur. If they have occurred, the likelihood reflects the marginal probability that the tied-failure events occurred before the nonfailure events in the risk pool (the order that they occurred is not important). This is called the exact marginal likelihood (option `exactm`).

In the second view, time is discrete, so ties are expected. The likelihood is changed to reflect this discreteness and calculates the conditional probability that the observed failures are those that fail in the risk pool given the observed number of failures. This is called the exact partial likelihood (option `exactp`).

Let's assume that there are five subjects— e_1, e_2, e_3, e_4, e_5 —in the risk pool and that subjects e_1 and e_2 fail. Had we been able to observe the events at a better resolution, we might have seen that e_1 failed from risk pool $e_1 + e_2 + e_3 + e_4 + e_5$ and then e_2 failed from risk pool $e_2 + e_3 + e_4 + e_5$. Alternatively, e_2 might have failed first from risk pool $e_1 + e_2 + e_3 + e_4 + e_5$, and then e_1 failed from risk pool $e_1 + e_3 + e_4 + e_5$.

The Breslow method (option `breslow`) for handling tied values simply says that since we do not know the order, we will use the largest risk pool for each tied failure event. This method assumes that both e_1 and e_2 failed from risk pool $e_1 + e_2 + e_3 + e_4 + e_5$. This approximation is fast and is the default method for handling ties. If there are many ties in the dataset, this approximation will not be accurate since the risk pools include too many observations. The Breslow method is an approximation of the exact marginal likelihood.

The Efron method (option `efron`) for handling tied values assumes that the first risk pool is $e_1 + e_2 + e_3 + e_4 + e_5$ and the second risk pool is either $e_2 + e_3 + e_4 + e_5$ or $e_1 + e_3 + e_4 + e_5$. From this, Efron noted that the e_1 and e_2 terms were in the second risk pool with probability 1/2 and so used for the second risk pool $.5(e_1 + e_2) + e_3 + e_4 + e_5$. Efron's approximation is a more accurate approximation of the exact marginal likelihood than Breslow's but takes longer to calculate.

The exact marginal method (option `exactm`) is a misnomer in that the calculation performed is also an *approximation* of the exact marginal likelihood. It is an approximation because it evaluates the likelihood (and derivatives) by using 15-point Gauss–Laguerre quadrature. For small-to-moderate samples, this is slower than the Efron approximation, but the difference in execution time diminishes when samples become larger. You may want to consider the quadrature when deciding to use this method. If the number of tied deaths is large (on average), the quadrature approximation of the function is not well behaved. A little empirical checking suggests that if the number of tied deaths is larger (on average) than 30, the quadrature does not approximate the function well.

When we view time as discrete, the exact partial method (option `exactp`) is the final method available. This approach is equivalent to computing conditional logistic regression where the groups are defined by the risk sets and the outcome is given by the death variable. This is the slowest method to use and can take a significant amount of time if the risk sets and the number of tied failures are large.

Cox regression with discrete time-varying covariates

In [ST] `stset`, we introduce the Stanford heart transplant data in which there are one or two records per patient depending on whether they received a new heart.

This dataset (Crowley and Hu 1977) consists of 103 patients admitted to the Stanford Heart Transplantation Program. Patients were admitted into the program after review by a committee and then waited for an available donor heart. While waiting, some patients died or were transferred out of the program, but 67% received a transplant. The dataset includes the year the patient was accepted into the program along with the patient's age, whether the patient had other heart surgery previously, and whether the patient received a transplant.

In the data, `posttran` becomes 1 when a patient receives a new heart, so it is a time-varying covariate. That does not affect what we type to fit the model:

```
. use http://www.stata-press.com/data/r10/stan3, clear
(Heart transplant data)
. stset t1, failure(died) id(id)
(output omitted)
```

```

. stcox age posttran surg year
      failure _d: died
      analysis time _t: t1
      id: id
Iteration 0:  log likelihood = -298.31514
Iteration 1:  log likelihood = -289.7344
Iteration 2:  log likelihood = -289.53498
Iteration 3:  log likelihood = -289.53378
Iteration 4:  log likelihood = -289.53378
Refining estimates:
Iteration 0:  log likelihood = -289.53378
Cox regression -- Breslow method for ties
No. of subjects =          103          Number of obs =          172
No. of failures =           75
Time at risk   =       31938.1
Log likelihood = -289.53378          LR chi2(4) =       17.56
                                          Prob > chi2 =       0.0015

```

	_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
age		1.030224	.0143201	2.14	0.032	1.002536 1.058677
posttran		.9787243	.3032597	-0.07	0.945	.5332291 1.796416
surgery		.3738278	.163204	-2.25	0.024	.1588759 .8796
year		.8873107	.059808	-1.77	0.076	.7775022 1.012628

We find that older patients have higher hazards, that patients tend to do better over time, and that patients with prior surgery do better. Whether a patient ultimately receives a transplant does not seem to make much difference.

Cox regression with continuous time-varying covariates

The basic proportional hazards regression assumes the relationship

$$h(t) = h_0(t) \exp(\beta_1 x_1 + \cdots + \beta_k x_k)$$

where $h_0(t)$ is the baseline hazard function. For most purposes this model is sufficient, but sometimes we may wish to introduce variables of the form $z_i(t) = z_i g(t)$, which vary continuously with time so that

$$h(t) = h_0(t) \exp\{\beta_1 x_1 + \cdots + \beta_k x_k + g(t)(\gamma_1 z_1 + \cdots + \gamma_m z_m)\}$$

where (z_1, \dots, z_m) are the time-varying covariates and where estimation has the net effect of estimating, say, a regression coefficient γ_i for a covariate $g(t)z_i$, which is a function of the current time.

The time-varying covariates (z_1, \dots, z_m) are specified using the `tvc(varlist)` option, and $g(t)$ is specified using the `texp(exp)` option, where t in $g(t)$ is analysis time. For example, if we want $g(t) = \ln(t)$, we would use `texp(ln(_t))` since `_t` stores the analysis time once the data are `stset`.

Since the calculations in Cox regression concern themselves only with the times at which failures occur, the above results could also be achieved by `stsplitting` the data at the observed failure times and manually generating the time-varying covariates. When this is feasible, `stsplit` merely represents a more convenient way to accomplish this. However, for large datasets with many distinct failure times, using `stsplit` may produce datasets that are too large to fit in memory, and even if this were not so, the estimation would take far longer to complete. For these reasons, the options described above were introduced.

▷ Example 3

Consider a dataset consisting of 45 observations on recovery time from walking pneumonia. Recovery time (in days) is recorded in the variable `time`, and there are measurements on the covariates `age`, `drug1`, and `drug2`, where `drug1` and `drug2` interact a choice of treatment with initial dosage level. The study was terminated after 30 days, so those who had not recovered by that time were censored (`cured = 0`).

```
. use http://www.stata-press.com/data/r10/drugtr2
. list age drug1 drug2 time cured in 1/12, sep(0)
```

	age	drug1	drug2	time	cured
1.	36	0	50	20.6	1
2.	14	0	50	6.8	1
3.	43	0	125	8.6	1
4.	25	100	0	10	1
5.	50	100	0	30	0
6.	26	0	100	13.6	1
7.	21	150	0	5.4	1
8.	25	0	100	15.4	1
9.	32	125	0	8.6	1
10.	28	150	0	8.5	1
11.	34	0	100	30	0
12.	40	0	50	30	0

Patient 1 took 50 mg of drug number 2 and was cured after 20.6 days, whereas patient 5 took 100 mg of drug number 1 and had yet to recover when the study ended and so was censored at 30 days.

We run a standard Cox regression after `stsetting` the data.

```
. stset time, failure(cured)
      failure event:  cured != 0 & cured < .
obs. time interval:  (0, time]
exit on or before:  failure
```

```
      45 total obs.
       0 exclusions
```

```
      45 obs. remaining, representing
      36 failures in single record/single failure data
677.9 total analysis time at risk, at risk from t =      0
           earliest observed entry t =      0
           last observed exit t =      30
```

(Continued on next page)

```

. stcox age drug1 drug2
      failure _d:  cured
      analysis time _t:  time
Iteration 0:  log likelihood = -116.54385
Iteration 1:  log likelihood = -102.77311
Iteration 2:  log likelihood = -101.92794
Iteration 3:  log likelihood = -101.92504
Iteration 4:  log likelihood = -101.92504
Refining estimates:
Iteration 0:  log likelihood = -101.92504
Cox regression -- Breslow method for ties
No. of subjects =          45                Number of obs =          45
No. of failures =          36
Time at risk   = 677.9000034
Log likelihood = -101.92504                LR chi2(3)    =          29.24
                                           Prob > chi2   =          0.0000

```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
age	.8759449	.0253259	-4.58	0.000	.8276873	.9270162
drug1	1.008482	.0043249	1.97	0.049	1.000041	1.016994
drug2	1.00189	.0047971	0.39	0.693	.9925323	1.011337

The output includes p -values for the tests of the null hypotheses that each regression coefficient is zero or, equivalently, that each hazard ratio is one. That all hazard ratios are apparently close to one is a matter of scale; however, we can see that drug number 1 significantly increases the risk of being cured and so is an effective drug, whereas drug number 2 is ineffective (given the presence of age and drug number 1 in the model).

Suppose now that we wish to fit a model in which we account for the effect that as time goes by, the actual level of the drug remaining in the body diminishes, say, at an exponential rate. If it is known that the half-life of both drugs is close to 2 days, we can say that the actual concentration level of the drug in the patient's blood is proportional to the initial dosage times $\exp(-0.35t)$, where t is analysis time. We now fit a model that reflects this change.

```

. stcox age, tvc(drug1 drug2) texp(exp(-0.35*_t)) nolog
      failure _d:  cured
      analysis time _t:  time
Cox regression -- Breslow method for ties
No. of subjects =          45                Number of obs =          45
No. of failures =          36
Time at risk   = 677.9000034
Log likelihood = -98.052763                LR chi2(3)    =          36.98
                                           Prob > chi2   =          0.0000

```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
rh						
age	.8614636	.028558	-4.50	0.000	.8072706	.9192948
t						
drug1	1.304744	.1135967	3.06	0.002	1.100059	1.547514
drug2	1.200613	.1113218	1.97	0.049	1.001103	1.439882

Note: second equation contains variables that continuously vary with respect to time; variables are interacted with current values of $\exp(-0.35*_t)$.

The first equation, `rh`, reports the results (hazard ratios) for the covariates that do not vary over time; the second equation, `t`, reports the results for the time-varying covariates.

As the level of drug in the blood system decreases, the drug's effectiveness diminishes. Accounting for this serves to unmask the effects of both drugs in that we now see increased effects on both. In fact, the effect on recovery time of drug number 2 now becomes significant.

□ Technical Note

The interpretation of hazard ratios requires careful consideration here. For the first model, the hazard ratio for, say, `drug1` is interpreted as the proportional change in hazard when the dosage level of `drug1` is increased by one unit. For the second model, the hazard ratio for `drug1` is the proportional change in hazard when the blood concentration level—i.e., $\text{drug1} \cdot \exp(-0.35t)$ —increases by one. □

Since the number of observations in our data is relatively small, for illustrative purposes we can `stsplit` the data at each recovery time, manually generate the blood concentration levels, and refit the second model.

```
. generate id=_n
. streset, id(id)
  (output omitted)
. stsplit, at(failures)
(31 failure times)
(812 observations (episodes) created)
. generate drug1emt = drug1*exp(-0.35*_t)
. generate drug2emt = drug2*exp(-0.35*_t)
. stcox age drug1emt drug2emt
      failure _d:  cured
  analysis time _t:  time
             id:  id

Iteration 0:  log likelihood = -116.54385
Iteration 1:  log likelihood = -99.321912
Iteration 2:  log likelihood = -98.07369
Iteration 3:  log likelihood = -98.05277
Iteration 4:  log likelihood = -98.052763
Refining estimates:
Iteration 0:  log likelihood = -98.052763

Cox regression -- Breslow method for ties
No. of subjects =          45          Number of obs   =          857
No. of failures =          36
Time at risk    = 677.9000034

                               LR chi2(3)    =          36.98
                               Prob > chi2    =          0.0000
Log likelihood    = -98.052763
```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
age	.8614636	.028558	-4.50	0.000	.8072706	.9192948
drug1emt	1.304744	.1135967	3.06	0.002	1.100059	1.547514
drug2emt	1.200613	.1113218	1.97	0.049	1.001103	1.439882

We get the same answer. However, this required more work both for Stata and for you.

The full functionality of `stcox` is available with time-varying covariates, including the ability to generate residuals and baseline functions. The only exception is when the `exactm` or `exactp` options is specified for handling ties, in which case the `tvc(varlist)` option is currently not supported. You must then use the `stsplit` approach outlined above.

□ Technical Note

Finally, specifying $g(t)$ via the `texp(exp)` option is intended for functions of analysis time, `_t` only, with the default being `texp(_t)` if left unspecified. However, specifying any other valid Stata expression will not produce a syntax error, yet usually will not yield the anticipated output. For example, specifying `texp(varname)` will not generate interaction terms. This has to do mainly with how the calculations are carried out—by careful summations over risk pools at each failure time. □

Robust estimate of variance

By default, `stcox` produces the conventional estimate for the variance–covariance matrix of the coefficients (and hence the reported standard errors). If, however, you specify the `vce(robust)` option, `stcox` switches to the robust variance estimator (Lin and Wei 1989).

The key to the robust calculation is using the efficient score residual for each subject in the data for the variance calculation. Even in simple single-record, single-failure survival data, the same subjects appear repeatedly in the risk pools, and the robust calculation tries to account for that.

▷ Example 4

Refitting the Stanford heart transplant data model with robust standard errors, we obtain

```
. use http://www.stata-press.com/data/r10/stan3, clear
(Heart transplant data)
. stset t1, failure(died) id(id)
      id: id
      failure event: died != 0 & died < .
obs. time interval: (t1[_n-1], t1]
exit on or before: failure
```

```
172 total obs.
  0 exclusions
```

```
172 obs. remaining, representing
103 subjects
 75 failures in single failure-per-subject data
31938.1 total analysis time at risk, at risk from t =      0
      earliest observed entry t =      0
      last observed exit t =      1799
```

```
. stcox age posttran surg year, vce(robust)
      failure _d: died
      analysis time _t: t1
      id: id
Iteration 0:  log pseudolikelihood = -298.31514
Iteration 1:  log pseudolikelihood = -289.7344
Iteration 2:  log pseudolikelihood = -289.53498
Iteration 3:  log pseudolikelihood = -289.53378
Iteration 4:  log pseudolikelihood = -289.53378
Refining estimates:
Iteration 0:  log pseudolikelihood = -289.53378
Cox regression -- Breslow method for ties
No. of subjects      =          103          Number of obs   =          172
No. of failures     =           75
Time at risk        =        31938.1
Log pseudolikelihood = -289.53378          Wald chi2(4)      =          19.68
                                                Prob > chi2      =          0.0006
                                                (Std. Err. adjusted for 103 clusters in id)
```

_t	Haz. Ratio	Robust		z	P> z	[95% Conf. Interval]	
		Std. Err.					
age	1.030224	.0148771	2.06	0.039	1.001474	1.059799	
posttran	.9787243	.2961736	-0.07	0.943	.5408498	1.771104	
surgery	.3738278	.1304912	-2.82	0.005	.1886013	.7409665	
year	.8873107	.0613176	-1.73	0.084	.7749139	1.01601	

Note the word **Robust** above **Std. Err.** in the table and the phrase “Std. Err. adjusted for 103 clusters in id” above the table.

The hazard ratio estimates are the same as before, but the standard errors are slightly different.



□ Technical Note

In the previous example, `stcox` knew to specify `vce(cluster id)` for us when we specified `vce(robust)`.

To see the importance of `vce(cluster id)`, consider simple single-record, single-failure survival data, a piece of which is

t0	t	died	x
0	5	1	1
0	9	0	1
0	8	0	0

and then consider the absolutely equivalent multiple-record survival data:

id	t0	t	died	x
1	0	3	0	1
1	3	5	1	1
2	0	6	0	1
2	6	9	0	1
3	0	3	0	0
3	3	8	0	0

Both datasets record the same underlying data, and so both should produce the same numerical results. This should be true whether or not `vce(robust)` is specified.

In the second dataset, were we to ignore `id`, it would appear that there are 6 observations on six subjects. The key ingredients in the robust calculation are the efficient score residuals, and viewing the data as 6 observations on six subjects produces different score residuals. Let us call the six score residuals s_1, s_2, \dots, s_6 and the three score residuals that would be generated by the first dataset $S_1, S_2,$ and S_3 . $S_1 = s_1 + s_2$, $S_2 = s_3 + s_4$, and $S_3 = s_5 + s_6$.

That residuals sum is the key to understanding the `vce(cluster clustvar)` option. When you specify `vce(cluster id)`, Stata makes the robust calculation based not on the overly detailed s_1, s_2, \dots, s_6 but on $s_1 + s_2, s_3 + s_4,$ and $s_5 + s_6$. That is, Stata sums residuals within clusters before entering them into subsequent calculations (where they are squared), so results estimated from the second dataset are equal to those estimated from the first. In more complicated datasets with time-varying regressors, delayed entry, and gaps, this action of summing within cluster, in effect, treats the cluster (which is typically a subject) as a unified whole.

Because we had `stset` an `id()` variable, `stcox` knew to specify `vce(cluster id)` for us when we specified `vce(robust)`. You may, however, override the default clustering by specifying `vce(cluster clustvar)` with a different variable from the one you used in `stset`, `id()`. This is useful in analyzing multiple-failure data, where you need to `stset` a pseudo-ID establishing the time from the last failure as the onset of risk.

□

Cox regression with multiple-failure data

In [ST] `stsum`, we introduce a multiple-failure dataset:

```
. use http://www.stata-press.com/data/r10/mfail
. stdescribe
```

Category	total	per subject			
		mean	min	median	max
no. of subjects	926				
no. of records	1734	1.87257	1	2	4
(first) entry time		0	0	0	0
(final) exit time		470.6857	1	477	960
subjects with gap	0				
time on gap if gap	0
time at risk	435855	470.6857	1	477	960
failures	808	.8725702	0	1	3

This dataset contains two variables—`x1` and `x2`—which we believe affect the hazard of failure.

If we simply want to analyze these multiple-failure data as if the baseline hazard remains unchanged as events occur (that is, the hazard may change with time, but time is measured from 0 and is independent of when the last failure occurred), we can type

```

. stcox x1 x2, vce(robust)
Iteration 0:  log pseudolikelihood = -5034.9569
Iteration 1:  log pseudolikelihood = -4978.4198
Iteration 2:  log pseudolikelihood = -4978.1915
Iteration 3:  log pseudolikelihood = -4978.1914
Refining estimates:
Iteration 0:  log pseudolikelihood = -4978.1914
Cox regression -- Breslow method for ties
No. of subjects      =          926          Number of obs   =       1734
No. of failures      =          808
Time at risk        =       435855
Log pseudolikelihood = -4978.1914          Wald chi2(2)    =       152.13
                                          Prob > chi2     =       0.0000
                                          (Std. Err. adjusted for 926 clusters in id)

```

		Robust				[95% Conf. Interval]	
_t	Haz. Ratio	Std. Err.	z	P> z			
x1	2.273456	.1868211	9.99	0.000	1.935259	2.670755	
x2	.329011	.0523425	-6.99	0.000	.2408754	.4493951	

We chose to fit this model with robust standard errors—we specified `vce(robust)`—but you can estimate conventional standard errors if you wish.

In [ST] `stsum`, we discuss analyzing this dataset as the time since last failure. We wished to assume that the hazard function remained unchanged with failure, except that one restarted the same hazard function. To that end, we made the following changes to our data:

```

. stgen nf = nfailures()
. egen newid = group(id nf)
. sort newid t
. by newid: replace t = t - t0[1]
(808 real changes made)
. by newid: gen newt0 = t0 - t0[1]
. stset t, id(newid) failure(d) time0(newt0) noshow
           id: newid
           failure event: d != 0 & d < .
obs. time interval: (newt0, t]
exit on or before: failure

```

```

1734 total obs.
   0 exclusions

```

```

1734 obs. remaining, representing
1734 subjects
  808 failures in single failure-per-subject data
435444 total analysis time at risk, at risk from t =           0
           earliest observed entry t =           0
           last observed exit t =           797

```

That is, we took each subject and made many `newid` subjects out of each, with each subject entering at time 0 (now meaning the time of the last failure). `id` still identifies a real subject, but Stata thinks the identifier variable is `newid` because we `stset, id(newid)`. If we were to fit a model with `vce(robust)`, we would get

```
. stcox x1 x2, vce(robust) nolog
Cox regression -- Breslow method for ties
No. of subjects      =          1734      Number of obs   =          1734
No. of failures      =           808
Time at risk         =        435444
Log pseudolikelihood = -5062.5815      Wald chi2(2)    =          88.51
                                          Prob > chi2     =          0.0000
                                          (Std. Err. adjusted for 1734 clusters in newid)
```

_t	Haz. Ratio	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
x1	2.002547	.1936906	7.18	0.000	1.656733	2.420542
x2	.2946263	.0569167	-6.33	0.000	.2017595	.4302382

Note carefully the message concerning the clustering: standard errors have been adjusted for clustering on `newid`. We, however, want the standard errors adjusted for clustering on `id`, so we must specify the `vce(cluster clustvar)` option:

```
. stcox x1 x2, vce(cluster id) nolog
Cox regression -- Breslow method for ties
No. of subjects      =          1734      Number of obs   =          1734
No. of failures      =           808
Time at risk         =        435444
Log pseudolikelihood = -5062.5815      Wald chi2(2)    =          93.66
                                          Prob > chi2     =          0.0000
                                          (Std. Err. adjusted for 926 clusters in id)
```

_t	Haz. Ratio	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
x1	2.002547	.1920151	7.24	0.000	1.659452	2.416576
x2	.2946263	.0544625	-6.61	0.000	.2050806	.4232709

That is, if you are using `vce(robust)`, you must remember to specify `vce(cluster clustvar)` for yourself when

1. you are analyzing multiple-failure data and
2. you have reset time to time since last failure, so what Stata considers the subjects are really subsubjects.

Stratified estimation

When you type

```
. stcox xvars, strata(svars)
```

you are allowing the baseline hazard functions to differ for the groups identified by `svars`. This is equivalent to fitting separate Cox proportional hazards models under the constraint that the coefficients are equal but not the baseline hazard functions.

► Example 5

Say that in the Stanford heart experiment data, there was a change in treatment for all patients, before and after transplant, in 1970 and then again in 1973. Further assume that the proportional-hazards assumption is not reasonable for these changes in treatment—perhaps the changes result in short-run benefit but little expected long-run benefit. Our interest in the data is not in the effect of these treatment changes but in the effect of transplantation, for which we still find the proportional-hazards assumption reasonable. We might fit our model to account for these fictional changes by typing

```
. use http://www.stata-press.com/data/r10/stan3, clear
(Heart transplant data)
. generate pgroup = year
. recode pgroup min/69=1 70/72=2 73/max=3
(pgroup: 172 changes made)
. stcox age posttran surg year, strata(pgroup) nolog
      failure _d: died
      analysis time _t: t1
                  id: id

Stratified Cox regr. -- Breslow method for ties
No. of subjects =          103                Number of obs   =          172
No. of failures =           75
Time at risk   =       31938.1
Log likelihood =    -213.35033                LR chi2(4)         =          20.67
                                                Prob > chi2       =          0.0004
```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
age	1.027406	.0150188	1.85	0.064	.9983874 1.057268
posttran	1.075476	.3354669	0.23	0.816	.583567 1.982034
surgery	.2222415	.1218386	-2.74	0.006	.0758882 .6508429
year	.5523966	.1132688	-2.89	0.004	.3695832 .825638

Stratified by pgroup

Of course, we could obtain the robust estimate of variance by also including the `vce(robust)` option.

◀

Cox regression with shared frailty

A shared-frailty model is the survival-data analog to regression models with random effects. A *frailty* is a latent random effect that enters multiplicatively on the hazard function. In a Cox model, the data are organized as $i = 1, \dots, n$ groups with $j = 1, \dots, n_i$ observations in group i . For the j th observation in the i th group, the hazard is

$$h_{ij}(t) = h_0(t)\alpha_i \exp(\mathbf{x}_{ij}\beta)$$

where α_i is the group-level frailty. The frailties are unobservable positive quantities and are assumed to have mean one and variance θ , to be estimated from the data. You can fit a Cox shared-frailty model by specifying `shared(varname)`, where `varname` defines the groups over which frailties are shared. `stcox`, `shared()` treats the frailties as being gamma distributed, but this is mainly an issue of computational convenience; see *Methods and Formulas*. Theoretically, any distribution with positive support, mean one, and finite variance may be used to model frailty.

Shared-frailty models are used to model within-group correlation; observations within a group are correlated because they share the same frailty. The estimate of θ is used to measure the degree of within-group correlation, and the shared-frailty model reduces to standard Cox when $\theta = 0$.

For $\nu_i = \log\alpha_i$, the hazard can also be expressed as

$$h_{ij}(t) = h_0(t) \exp(\mathbf{x}_{ij}\boldsymbol{\beta} + \nu_i)$$

and thus the log-frailties, ν_i , are analogous to random effects in standard linear models. In fact, you can retrieve estimates of the ν_i by specifying `effects(newvar)`.

▷ Example 6

Consider the data from a study of 38 kidney dialysis patients, as described in McGilchrist and Aisbett (1991). The study is concerned with the prevalence of infection at the catheter insertion point. Two recurrence times (in days) are measured for each patient, and each recorded time is the time from initial insertion (onset of risk) to infection or censoring.

```
. use http://www.stata-press.com/data/r10/catheter, clear
(Kidney data, McGilchrist and Aisbett, Biometrics, 1991)
. list patient time infect age female in 1/10
```

	patient	time	infect	age	female
1.	1	16	1	28	0
2.	1	8	1	28	0
3.	2	13	0	48	1
4.	2	23	1	48	1
5.	3	22	1	32	0
6.	3	28	1	32	0
7.	4	318	1	31.5	1
8.	4	447	1	31.5	1
9.	5	30	1	10	0
10.	5	12	1	10	0

Each patient (`patient`) has two recurrence times (`time`) recorded, with each catheter insertion resulting in either infection (`infect==1`) or right-censoring (`infect==0`). Among the covariates measured are `age` and sex (`female==1` if female, `female==0` if male).

One subtlety to note concerns the use of the generic term *subjects*. In this example, the subjects are taken to be the individual catheter insertions, not the patients themselves. This is a function of how the data were recorded—the onset of risk occurs at catheter insertion (of which there are two for each patient), and not (say) at the time of admission of the patient into the study. We therefore have two subjects (insertions) within each group (patient).

It is reasonable to assume independence of patients but unreasonable to assume that recurrence times within each patient are independent. One solution would be to fit a standard Cox model, adjusting the standard errors of the estimated hazard ratios to account for the possible correlation by specifying `vce(cluster patient)`.

We could instead model the correlation by assuming that the correlation is the result of a latent patient-level effect, or frailty. That is, rather than fitting a standard model and specifying `vce(cluster patient)`, we could fit a frailty model by specifying `shared(patient)`.

```
. stset time, fail(infect)
(output omitted)
```

```

. stcox age female, shared(patient)
      failure _d: infect
      analysis time _t: time
Fitting comparison Cox model:
Estimating frailty variance:
Iteration 0:  log profile likelihood = -182.06713
Iteration 1:  log profile likelihood = -181.9791
Iteration 2:  log profile likelihood = -181.97453
Iteration 3:  log profile likelihood = -181.97453
Fitting final Cox model:
Iteration 0:  log likelihood = -199.05599
Iteration 1:  log likelihood = -183.72296
Iteration 2:  log likelihood = -181.99509
Iteration 3:  log likelihood = -181.97455
Iteration 4:  log likelihood = -181.97453
Refining estimates:
Iteration 0:  log likelihood = -181.97453
Cox regression --
      Breslow method for ties          Number of obs   =       76
      Gamma shared frailty            Number of groups =       38
Group variable: patient
No. of subjects =           76          Obs per group: min =        2
No. of failures =           58          avg           =        2
Time at risk   =          7424         max           =        2
Wald chi2(2)   =              11.66
Log likelihood = -181.97453           Prob > chi2     =       0.0029

```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
age	1.006202	.0120965	0.51	0.607	.9827701	1.030192
female	.2068678	.095708	-3.41	0.001	.0835376	.5122756
theta	.4754497	.2673108				

Likelihood-ratio test of theta=0: $\chi^2(01) = 6.27$ Prob>= $\chi^2 = 0.006$

Note: standard errors of hazard ratios are conditional on theta.

From the output, we obtain $\hat{\theta} = 0.475$, and given the standard error of $\hat{\theta}$ and likelihood-ratio test of $H_0: \theta = 0$, we find a significant frailty effect, meaning that the correlation within patient cannot be ignored. Contrast this with the analysis of the same data in [ST] **streg**, which considered both Weibull and lognormal shared-frailty models. For Weibull, there was significant frailty; for lognormal, there was not.

The estimated ν_i are not displayed in the coefficient table but may be retrieved at estimation by specifying `effects(newvar)`.

```

. qui stcox age female, shared(patient) effects(nu)
. sort nu
. list patient nu in 1/2

```

	patient	nu
1.	21	-2.4487067
2.	21	-2.4487067

```
. list patient nu in 75/1
```

	patient	nu
75.	7	.51871587
76.	7	.51871587

From the results above, we estimate that the least frail patient is patient 21, with $\hat{\nu}_{21} = -2.45$, and that the frailest patient is patient 7, with $\hat{\nu}_7 = 0.52$.

◀

In shared-frailty Cox models, the estimation consists of two steps. In the first step, the optimization is in terms of θ only. For fixed θ , the second step consists of fitting a standard Cox model via penalized log likelihood, with the ν_i introduced as estimable coefficients of dummy variables identifying the groups. The penalty term in the penalized log likelihood is a function of θ ; see *Methods and Formulas*. The final estimate of θ is taken to be the one that maximizes the penalized log likelihood. Once the optimal θ is obtained, it is held fixed, and a final penalized Cox model is fitted. As a result, the standard errors of the main regression parameters (or hazard ratios, if displayed as such) are treated as conditional on θ fixed at its optimal value.

□ Technical Note

With gamma-distributed frailty, hazard ratios decay over time in favor of the *frailty effect* and thus the displayed “Haz. Ratio” in the above output is actually the hazard ratio only for $t = 0$. The degree of decay depends on θ . Should the estimated θ be close to zero, the hazard ratios do regain their usual interpretation; see Gutierrez (2002) for details.

□

□ Technical Note

The likelihood-ratio test of $\theta = 0$ is a boundary test and thus requires careful consideration concerning the calculation of its p -value. In particular, the null distribution of the likelihood-ratio test statistic is not the usual χ_1^2 but is rather a 50:50 mixture of a χ_0^2 (point mass at zero) and a χ_1^2 , denoted as $\bar{\chi}_{01}^2$. See Gutierrez, Carter, and Drukker (2001) for more details.

□

□ Technical Note

In [ST] **streg**, shared-frailty models are compared and contrasted with *unshared* frailty models. Unshared-frailty models are used to model heterogeneity, and the frailties are integrated out of the conditional survivor function to produce an unconditional survivor function, which serves as a basis for all likelihood calculations.

Given the nature of Cox regression (the baseline hazard remains unspecified), there is no Cox regression analog to the unshared parametric frailty model as fitted using **streg**. That is not to say that you cannot fit a shared-frailty model with 1 observation per group; you can as long as you do not fit a null model. However, there are subtle differences in singleton-group data between shared- and unshared-frailty models; see Gutierrez (2002).

□

Obtaining baseline function estimates

When you specify options `basechazard(newvar)` and `basesurv(newvar)`—which you may do together or separately—you obtain estimates of the baseline cumulative hazard and survivor functions. When you specify the option `basehc(newvar)`, you obtain estimates of the baseline hazard contribution at each failure time, which are factors used to develop both the product-limit estimator for the survivor function generated by `basesurv(newvar)` and the estimator of the cumulative hazard function generated by `basechazard(newvar)`.

Although in theory $S_0(t) = \exp\{-H_0(t)\}$, where $S_0(t)$ is the baseline survivor function and $H_0(t)$ is the baseline cumulative hazard, the estimates produced by `basechazard()` and `basesurv()` do not exactly correspond in this manner, although they closely do. The reason is that `stcox` uses different estimation schemes for each; the exact formulas are given in *Methods and Formulas*.

When the model is fitted with the `strata()` option, you obtain estimates of the baseline functions for each stratum.

Let us first understand how `stcox` stores the results.

Mathematically, the baseline hazard contribution $h_i = (1 - \alpha_i)$ (see Kalbfleisch and Prentice 2002, 115) is defined at every analytic time t_i at which a failure occurs and is undefined (or, if you prefer, 0) at other times. Stata stores h_i in observations where a failure occurred and missing values in the other observations. For instance, here are some data on which we have fitted a Cox model and specified the option `basehc(h)`:

```
. use http://www.stata-press.com/data/r10/stan3, clear
(Heart transplant data)
. generate age40 = age - 40
. generate year70 = year - 70
. stcox age40 posttran surg year70, basehc(h)
(output omitted)
. list id _t0 _t _d h in 1/10
```

	id	_t0	_t	_d	h
1.	1	0	50	1	.01503465
2.	2	0	6	1	.02035303
3.	3	0	1	0	.
4.	3	1	16	1	.03339642
5.	4	0	36	0	.
6.	4	36	39	1	.01365406
7.	5	0	18	1	.01167142
8.	6	0	3	1	.02875689
9.	7	0	51	0	.
10.	7	51	675	1	.06215003

Here is the interpretation: At time `_t = 50`, the hazard contribution h_1 is .0150. At time `_t = 6`, the hazard contribution h_2 is .0204.

In observation 3, no hazard contribution is stored. Observation 3 contains a missing because observation 3 did not fail at time 1.

We see that values of the hazard contributions are stored only in observations that are marked as failing.

The baseline survivor function $S_0(t)$ is defined at all values of t : its estimate changes its value when failures occur, and at times when no failures occur, the estimated $S_0(t)$ is equal to its value at the time of the last failure.

Here are some data on which we have fitted a Cox model and specified both `basehc(h)` and `basesurv(s)`:

```
. drop h
. stcox age40 posttran surg year70, basehc(h) bases(s)
  (output omitted)
. list id _t0 _t _d h s in 1/10
```

	id	_t0	_t	_d	h	s
1.	1	0	50	1	.01503465	.68100303
2.	2	0	6	1	.02035303	.89846438
3.	3	0	1	0	.	.99089681
4.	3	1	16	1	.03339642	.84087361
5.	4	0	36	0	.	.7527663
6.	4	36	39	1	.01365406	.73259264
7.	5	0	18	1	.01167142	.82144038
8.	6	0	3	1	.02875689	.93568733
9.	7	0	51	0	.	.6705895
10.	7	51	675	1	.06215003	.26115633

At time `_t = 50`, the baseline survivor function is `.6810`, or more precisely, $S_0(50 + 0) = .6810$. What we mean by $S_0(t)$ is $S(t + 0)$, the probability of surviving just beyond time t . This is done to clarify that the probability does not include failure at precisely time t .

Understanding what is stored is easier if we sort by `_t`:

```
. gen notd = !_d
. sort _t notd
. drop notd
. sort _t
. list id _t0 _t _d h s in 1/18
```

	id	_t0	_t	_d	h	s
1.	15	0	1	1	.00910319	.99089681
2.	3	0	1	0	.	.99089681
3.	45	0	1	0	.	.99089681
4.	20	0	1	0	.	.99089681
5.	61	0	2	1	.02775802	.96339147
6.	75	0	2	1	.02775802	.96339147
7.	43	0	2	1	.02775802	.96339147
8.	39	0	2	0	.	.96339147
9.	46	0	2	0	.	.96339147
10.	95	0	2	0	.	.96339147
11.	54	0	3	1	.02875689	.93568733
12.	42	0	3	1	.02875689	.93568733
13.	6	0	3	1	.02875689	.93568733
14.	23	0	3	0	.	.93568733
15.	68	0	3	0	.	.93568733
16.	60	0	3	0	.	.93568733
17.	72	0	4	0	.	.93568733
18.	94	0	4	0	.	.93568733

The baseline hazard contribution is stored on every failure record—if multiple failures occur at a time, the value of the hazard contribution is repeated—and the baseline survivor is stored on every record. (More correctly, baseline values are stored on records that meet the criterion and that were used in estimation. If some observations are explicitly or implicitly excluded from the estimation, their baseline values will be set to missing, no matter what.)

With this listing, we can better understand how the hazard contributions are used to calculate the survivor function. Since the patient with `id = 15` died at time $t_1 = 1$, the hazard contribution for that patient is $h_{15} = .00910319$. Since that was the only death at $t_1 = 1$, the estimated survivor function at this time is $S_0(1) = 1 - h_{15} = 1 - .00910319 = .99089681$. The next death occurs at time $t_1 = 2$, and the hazard contribution at this time for patient 61 is $h_{61} = .02775802$. Multiplying the previous survivor function value by $1 - h_{61}$ gives the new survivor function at $t_1 = 2$ as $S_0(2) = .96339147$. The other survivor function values are then calculated in succession, using this method at each failure time. At times when no failures occur, the survivor function remains unchanged.

If we had fitted a stratified model by using the `strata()` option, the recorded baseline hazard contribution and survivor on each record would be for the stratum of the record.

□ Technical Note

If you want to store the baseline hazard contribution on every record for which it is defined and not just on the failure records, you would do the following:

```
. sort _t _d
. by _t: replace h = h[_N]
```

The above assumes that you specified `basehc(h)` when fitting the Cox model. If you also specified the `strata()` option, say, `strata(group)`, the instructions would be

```
. sort group _t _d
. by group _t: replace h = h[_N]
```

In both examples, we placed the data in time order: we put the failures at the end of each time group and then copied the last value of `h` within each time to all the observations for that time.

Consider obtaining the estimate of $S_0(t)$ from the h_i s:

```
. sort _t _d
. by _t: keep if _d & _n==_N
. generate s = 1-h
. replace s = s[_n-1]*s if _n>1
```

If you had obtained stratified estimates, the equivalent code would be

```
. sort group _t _d
. by group _t: keep if _d & _n==_N
. generate s = 1-h
. by group: replace s = s[_n-1]*s if _n>1
```

□

▷ Example 7

Baseline functions refer to the values of the functions when all covariates are set to 0. Let's graph the survival curve for the heart transplant model that we have been fitting, and to make the baseline curve reasonable, let us do that at `age = 40` and `year = 70`. (We did so previously without explanation, but the following technical note provides important information on why baseline values should be chosen to be reasonable, meaning that they are in the range of the data.)

Thus we will begin by creating variables that, when 0, correspond to the baseline values we desire, and then we will refit our model, specifying the `basesurv()` option:

```
. use http://www.stata-press.com/data/r10/stan3, clear
(Heart transplant data)
. generate age40 = age-40
. generate year70 = year-70
. stcox age40 posttran surg year70, bases(s) nolog
      failure _d: died
      analysis time _t: t1
      id: id

Cox regression -- Breslow method for ties
No. of subjects =          103          Number of obs =          172
No. of failures =           75
Time at risk   =       31938.1
Log likelihood =    -289.53378          LR chi2(4) =          17.56
                                          Prob > chi2 =          0.0015
```

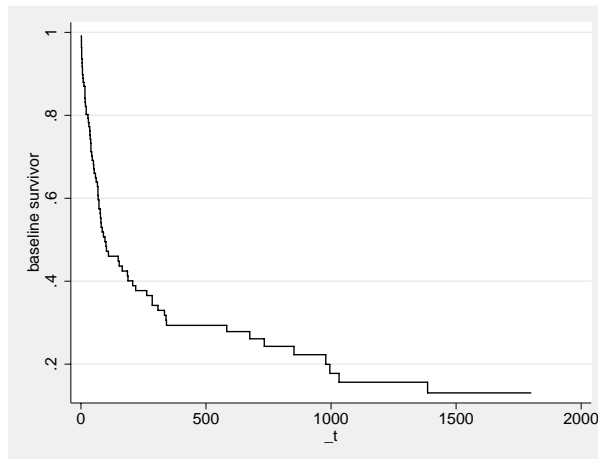
_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
age40	1.030224	.0143201	2.14	0.032	1.002536	1.058677
posttran	.9787243	.3032597	-0.07	0.945	.5332291	1.796416
surgery	.3738278	.163204	-2.25	0.024	.1588759	.8796
year70	.8873107	.059808	-1.77	0.076	.7775022	1.012628

```
. summarize s
      Variable |      Obs      Mean   Std. Dev.   Min      Max
-----+-----+-----+-----+-----+-----
      s       |      172   .629187   .2530009   .130666   .9908968
```

Our recentering of `age` and `year` did not affect the estimation. Also, the `s` variable we have just generated is the `s` variable we summarized above.

Here is a graph of the baseline survival curve:

```
. line s _t, sort c(J)
```



There is an easier way to graph estimated survivor, cumulative hazard, and hazard functions, both at baseline and at specified covariate values. Use `stcurve`, as we demonstrate in [ST] `stcurve`. ◀

□ Technical Note

If you specify the `basechazard()` or `basesurv()` option, for numerical accuracy reasons, the baseline functions must correspond to something reasonable in your data. Remember, the baseline functions correspond to all covariates equal to 0 in your Cox model.

Consider, for instance, a Cox model that includes the variable `calendar year` among the covariates. Say that `year` varies between 1980 and 1996. The baseline functions would correspond to year 0, almost 2,000 years in the past. Say that the estimated coefficient on `year` is $-.2$, meaning that the hazard ratio for one year to the next is a reasonable 0.82.

Think carefully about the contribution to the predicted log cumulative hazard: it would be approximately $-.2 \times 2,000 = -400$. Now $e^{-400} \approx 10^{-173}$, which on a digital computer is 1. There is simply no hope that $H_0(t)e^{-400}$ will produce an accurate estimate of $H(t)$.

Even with less extreme numbers, problems arise, even in the calculation of the baseline survivor function. Baseline hazard contributions near 1 produce baseline survivor functions, with the steps differing by 10s of orders of magnitudes because the calculation of the survivor function is cumulative. Producing a meaningful graph of such a survivor function is hopeless, and adjusting the survivor function (as opposed to the hazard contribution) to other values of the covariates is too much work.

For these reasons, covariate values of 0 must be meaningful if you are going to specify the `basechazard()` or `basesurv()` option. As the baseline values move to absurdity, the first problem you will encounter is a baseline survivor function that is too hard to interpret, even though the baseline hazard contributions are estimated accurately. Further out, the procedure Stata uses to estimate the baseline hazard contributions will break down—it will produce results that are exactly 1.

This, in fact, occurs with the Stanford heart transplant data:

```
. use http://www.stata-press.com/data/r10/stan3, clear
(Heart transplant data)
. stcox age posttran surg year, basec(ch) bases(s)
(output omitted)
. summarize ch s
```

Variable	Obs	Mean	Std. Dev.	Min	Max
ch	172	745.1134	682.8671	11.88239	2573.637
s	172	1.45e-07	9.43e-07	0	6.24e-06

The hint that there are problems is that the values of `ch` are huge and the values of `s` are close to zero. In this dataset, `age` (which ranges from 8 to 64 with a mean value of 45) and `year` (which ranges from 67 to 74) are the problems. The baseline functions correspond to a newborn at the turn of the century on the waiting list for a heart transplant!

To obtain accurate estimates of the baseline functions, type

```
. drop ch s
. generate age40 = age-40
. generate year70 = year-70
. stcox age40 posttran surg year70, basec(ch) bases(s)
(output omitted)
. summarize ch s
```

Variable	Obs	Mean	Std. Dev.	Min	Max
ch	172	.5685743	.521076	.0090671	1.963868
s	172	.629187	.2530009	.130666	.9908968

Adjusting the variables does not affect the coefficient (and hence hazard ratio) estimates, but it changes the values at which the baseline functions are estimated to be within the range of the data. □

□ Technical Note

When used with shared-frailty models, the options `basehc()`, `basesurv()`, and `basechazard()` produce estimates of baseline quantities that are based on the last-step penalized Cox model fit. Therefore, the term *baseline* means that not only are the covariates set to zero but ν_i also is, and thus *baseline* corresponds to $\alpha_i = 1$ (and zero covariates).

□

Saved Results

`stcox` saves the following in `e()`:

Scalars

<code>e(N)</code>	number of observations	<code>e(N_g)</code>	number of groups
<code>e(N_sub)</code>	number of subjects	<code>e(g_max)</code>	largest group size
<code>e(N_fail)</code>	number of failures	<code>e(g_min)</code>	smallest group size
<code>e(risk)</code>	total time at risk	<code>e(g_avg)</code>	average group size
<code>e(N_clust)</code>	number of clusters	<code>e(l1)</code>	log likelihood
<code>e(df_m)</code>	model degrees of freedom	<code>e(l1_0)</code>	log likelihood, constant-only model
<code>e(r2_p)</code>	pseudo- <i>R</i> -squared	<code>e(chi2)</code>	χ^2
<code>e(theta)</code>	frailty parameter	<code>e(se_theta)</code>	standard error of θ
<code>e(chi2_c)</code>	χ^2 , comparison model	<code>e(l1_c)</code>	log likelihood, comparison model
<code>e(p_c)</code>	significance, comparison model		

Macros

<code>e(cmd)</code>	<code>cox</code> or <code>stcox_fr</code>
<code>e(cmd2)</code>	<code>stcox</code>
<code>e(cmdline)</code>	command as typed
<code>e(depvar)</code>	<code>_t</code>
<code>e(t0)</code>	<code>_t0</code>
<code>e(texp)</code>	function used for time-varying covariates
<code>e(ties)</code>	method used for handling ties
<code>e(shared)</code>	frailty grouping variable
<code>e(offset)</code>	offset
<code>e(clustvar)</code>	name of cluster variable
<code>e(re_var)</code>	variable containing estimated random effects
<code>e(bases)</code>	variable containing baseline survivor function
<code>e(basec)</code>	variable containing baseline cumulative hazard function
<code>e(basehc)</code>	variable containing baseline hazard contributions
<code>e(mgale)</code>	variable containing partial martingale residuals
<code>e(vl_esr)</code>	variables containing partial efficient score residuals
<code>e(vl_sch)</code>	variables containing Schoenfeld residuals
<code>e(vl_ssc)</code>	variables containing scaled Schoenfeld residuals
<code>e(chi2type)</code>	Wald or LR; type of model χ^2 test
<code>e(vce)</code>	<i>vcetype</i> specified in <code>vce()</code>
<code>e(vcetype)</code>	title used to label Std. Err.
<code>e(method)</code>	requested estimation method
<code>e(crittype)</code>	optimization criterion
<code>e(properties)</code>	<code>b v</code>
<code>e(estat_cmd)</code>	program used to implement <code>estat</code>
<code>e(footnote)</code>	program used to implement the footnote display
<code>e(predict)</code>	program used to implement <code>predict</code>

Matrices		
<code>e(b)</code>	coefficient vector	<code>e(V)</code> variance–covariance matrix of the estimators
Functions		
<code>e(sample)</code>	marks estimation sample	

Methods and Formulas

`stcox` is implemented as an ado-file.

The proportional hazards model with time-dependent explanatory variables was first suggested by Cox (1972). For an introductory explanation, see Hosmer and Lemeshow (1999, chap. 3, 4, and 7), Kahn and Sempos (1989, 193–198), and Selvin (2004, 412–442). For an introduction for the social scientist, see Box-Steffensmeier and Jones (2004, chap. 4). For a comprehensive review of the methods in this entry, see Klein and Moeschberger (2003). For a detailed development of these methods, see Kalbfleisch and Prentice (2002). For more Stata-specific insight, see Cleves, Gould, and Gutierrez (2004); Dupont (2002); and Vittinghoff et al. (2005).

Let \mathbf{x}_i be the row vector of covariates for the time interval $(t_{0i}, t_i]$ for the i th observation in the dataset ($i = 1, \dots, N$). `stcox` obtains parameter estimates $\hat{\beta}$ by maximizing the partial log-likelihood function

$$\ln L = \sum_{j=1}^D \left[\sum_{k \in D_j} \mathbf{x}_k \beta - d_j \ln \left\{ \sum_{i \in R_j} \exp(\mathbf{x}_i \beta) \right\} \right]$$

where j indexes the ordered failure times $t_{(j)}$ ($j = 1, \dots, D$), D_j is the set of d_j observations that fail at $t_{(j)}$, d_j is the number of failures at $t_{(j)}$, and R_j is the set of observations k that are at risk at time $t_{(j)}$ (i.e., all k such that $t_{0k} < t_{(j)} \leq t_k$). This formula for $\ln L$ is for unweighted data and handles ties by using the Peto–Breslow approximation (Peto 1972; Breslow 1974), which is the default method of handling ties in `stcox`.

The variance of $\hat{\beta}$ is estimated by the conventional inverse matrix of (negative) second derivatives of $\ln L$, unless `vce(robust)` is specified, in which case the method of Lin and Wei (1989) is used. If `vce(cluster clustvar)` is specified, the efficient score residuals are summed within cluster before the sandwich (robust) estimator is applied.

See [R] **maximize** for a description of the maximization algorithm and for the translation of estimated results to hazard ratios when `hr` is specified.

`mgale()` stores in each observation the observation’s contribution to the martingale residual, which Stata’s terminology calls the “partial” martingale residual. The derivative of $\ln L$ can be written as

$$\frac{\partial \ln L}{\partial \beta} = \sum_{j=1}^D \sum_{i=1}^N \mathbf{x}_i dM_i(t_{(j)})$$

where

$$dM_i(t_{(j)}) = \delta_{ij} - I(t_{0i} < t_{(j)} \leq t_i) \frac{d_j \exp(\mathbf{x}_i \beta)}{\sum_{\ell \in R_j} \exp(\mathbf{x}_\ell \beta)}$$

with $\delta_{ij} = 1$ if observation i fails at $t_{(j)}$ and 0 otherwise. $I(\cdot)$ is the indicator function. $dM_i(t_{(j)})$ is the increment of the martingale residual for the i th observation owing the failures at time $t_{(j)}$. The `mga1e()` option saves the partial martingale residuals ΔM_i , which are the sum of the $dM_i(t_{(j)})$ over all failure times $t_{(j)}$, such that $t_{0i} < t_{(j)} \leq t_i$. For single-record data, the partial martingale residual is the martingale residual. For multiple-record data, the martingale residual for an individual can be obtained by summing the partial martingale residuals over all observations belonging to the individual. For a discussion of martingale residuals, see Fleming and Harrington (1991, 163–197).

The increments of the efficient score residuals are

$$d\mathbf{F}_{ij} = \mathbf{x}_{ij}^* dM_i(t_{(j)})$$

where

$$\mathbf{x}_{ij}^* = \mathbf{x}_i - \frac{\sum_{\ell \in R_j} \mathbf{x}_\ell \exp(\mathbf{x}_\ell \boldsymbol{\beta})}{\sum_{\ell \in R_j} \exp(\mathbf{x}_\ell \boldsymbol{\beta})}$$

When the `esr()` option is specified, `stcox` creates $p = \dim(\mathbf{x})$ new variables containing $\Delta \mathbf{F}_i$, which are the sum of $d\mathbf{F}_{ij}$ over all failure times $t_{(j)}$, such that $t_{0i} < t_{(j)} \leq t_i$. The efficient score residuals for an individual can be obtained by summing $\Delta \mathbf{F}_i$ over all observations i belonging to the individual.

The estimated baseline hazard contribution, if requested, is obtained as $h_j = 1 - \hat{\alpha}_j$, where $\hat{\alpha}_j$ is the solution of

$$\sum_{k \in D_j} \frac{\exp(\mathbf{x}_k \boldsymbol{\beta})}{1 - \hat{\alpha}_j^{\exp(\mathbf{x}_k \boldsymbol{\beta})}} = \sum_{l \in R_j} \exp(\mathbf{x}_l \boldsymbol{\beta})$$

(Kalbfleisch and Prentice 2002, equation (4.34), 115).

The estimated baseline survivor function, if requested, is obtained as

$$\hat{S}_0(t_{(j)}) = \prod_{h=0}^{j-1} \hat{\alpha}_h$$

where $\hat{\alpha}_0 = 1$.

The estimated baseline cumulative hazard function, if requested, is related to the baseline survivor function calculation, yet the values of $\hat{\alpha}_j$ are set at their starting values and are not iterated. Equivalently,

$$\hat{H}_0(t_{(j)}) = \sum_{h=0}^{j-1} \frac{d_j}{\sum_{l \in R_j} \exp(\mathbf{x}_l \boldsymbol{\beta})}$$

Tied values are handled using one of four approaches. The log likelihoods corresponding to the four approaches are given with weights (`exactp` does not allow weights) and offsets by

$$\ln L_{\text{breslow}} = \sum_{j=1}^D \sum_{i \in D_j} \left[w_i (\mathbf{x}_i \boldsymbol{\beta} + \text{offset}_i) - w_i \ln \left\{ \sum_{\ell \in R_j} w_\ell \exp(\mathbf{x}_\ell \boldsymbol{\beta} + \text{offset}_\ell) \right\} \right]$$

$$\begin{aligned} \ln L_{\text{efron}} = & \sum_{j=1}^D \sum_{k=1}^{d_j} \left\{ \frac{1}{d_j} \sum_{i \in D_j} w_i (\mathbf{x}_i \boldsymbol{\beta} + \text{offset}_i) - \right. \\ & \left. \left(\frac{1}{d_j} \sum_{i \in D_j} w_i \right) \ln \sum_{\ell \in R_j} w_\ell f_{kj\ell} \exp(\mathbf{x}_\ell \boldsymbol{\beta} + \text{offset}_\ell) \right\} \\ f_{kj\ell} = & \begin{cases} \frac{d_j - k + 1}{d_j} & \text{if } \delta_{\ell j} = 1 \\ 1 & \text{otherwise} \end{cases} \end{aligned}$$

$$\begin{aligned} \ln L_{\text{exactm}} = & \sum_{j=1}^D \ln \int_0^\infty \prod_{\ell \in D_j} \left\{ 1 - \exp\left(-\frac{e_\ell}{s} t\right) \right\}^{w_\ell} \exp(-t) dt \\ e_\ell = & \exp(\mathbf{x}_\ell \boldsymbol{\beta} + \text{offset}_\ell) \\ s = & \sum_{\substack{k \in R_j \\ k \notin D_j}} w_k \exp(\mathbf{x}_k \boldsymbol{\beta} + \text{offset}_k) = \text{sum of weighted nondeath risk scores} \end{aligned}$$

$$\begin{aligned} \ln L_{\text{exactp}} = & \sum_{j=1}^D \left\{ \sum_{i \in R_j} \delta_{ij} (\mathbf{x}_i \boldsymbol{\beta} + \text{offset}_i) - \ln f(r_j, d_j) \right\} \\ f(r, d) = & f(r-1, d) + f(r-1, d-1) \exp(\mathbf{x}_k \boldsymbol{\beta} + \text{offset}_k) \\ & k = r^{\text{th}} \text{ observation in the set } R_j \\ & r_j = \text{cardinality of the set } R_j \\ f(r, d) = & \begin{cases} 0 & \text{if } r < d \\ 1 & \text{if } d = 0 \end{cases} \end{aligned}$$

Calculations for the exact marginal log likelihood (and associated derivatives) are obtained with 15-point Gauss–Laguerre quadrature. The `breslow` and `efron` options both provide approximations of the exact marginal log likelihood. The `efron` approximation is a better (closer) approximation, but the `breslow` approximation is a much faster approximation. The choice of the approximation to use in a given situation should generally be driven by the proportion of ties in the data.

Weights are not allowed with the `exactp` method.

Cox–Snell residuals are calculated using

$$r_{C_i} = \delta_i - r_{M_i}$$

where r_{M_i} are the martingale residuals, and δ_i is a censoring indicator equal to 1 if the i th observation is a failure and 0 otherwise. The martingale residuals are nonnegative values following an exponential distribution with mean 1. Modified Cox–Snell residuals are defined similarly, except that the residual is augmented by some positive constant if the observation is censored. The most common modifications are to add either 1 or the Crowley–Hu adjustment of $\ln 2 \approx .693$ (which is the median of an exponential distribution with parameter 1).

Martingale residuals are calculated using

$$r_{M_i} = \delta_i - \sum_{j=1}^D I(t_{0i} < t_{(j)} \leq t_i) \frac{d_j \exp(\mathbf{x}_i \boldsymbol{\beta} + \text{offset}_i)}{\sum_{\ell \in R_j} \exp(\mathbf{x}_\ell \boldsymbol{\beta} + \text{offset}_\ell)}$$

These residuals are in $(-\infty, 1)$.

Using the Breslow–Peto approach, we have

$$r_{M_i} = 1 - \exp(\mathbf{x}_i \boldsymbol{\beta} + \text{offset}_i) \sum_{k: t_k \leq t_i} \frac{w_k \delta_k}{\sum_{\ell \in R_i} w_\ell \exp(\mathbf{x}_\ell \boldsymbol{\beta} + \text{offset}_\ell)}$$

Using the Efron approach, we have

$$r_{M_i} = \left\{ w_i - \frac{1}{d_i} \left(\sum_{j=1}^{d_i} w_j \right) \frac{\exp(\mathbf{x}_i \boldsymbol{\beta} + \text{offset}_i)}{\sum_{\ell \in R_i} w_\ell f_{ij\ell} \exp(\mathbf{x}_\ell \boldsymbol{\beta} + \text{offset}_\ell)} \right\} / w_i$$

Deviance residuals are calculated using

$$r_{D_i} = \text{sign}(r_{M_i}) \left[-2 \{r_{M_i} + \delta_i \cdot \log(\delta_i - r_{M_i})\} \right]^{1/2}$$

These residuals are expected to be symmetric about zero but do not necessarily sum to zero.

Schoenfeld residuals are calculated using the Breslow–Peto approach as

$$r_{uS_i} = \sum_{j=1}^D \delta_{ij} \left\{ x_{ui} - \frac{\sum_{\ell \in R_j} w_\ell x_{u\ell} \exp(\mathbf{x}_\ell \boldsymbol{\beta} + \text{offset}_\ell)}{\sum_{\ell \in R_j} w_\ell \exp(\mathbf{x}_\ell \boldsymbol{\beta} + \text{offset}_\ell)} \right\}$$

Using the Efron approach, we have

$$r_{uS_i} = \sum_{j=1}^D \delta_{ij} \left\{ x_{ui} - \frac{\sum_{\ell \in R_j} w_\ell x_{u\ell} f_{ij\ell} \exp(\mathbf{x}_\ell \boldsymbol{\beta} + \text{offset}_\ell)}{\sum_{\ell \in R_j} w_\ell f_{ij\ell} \exp(\mathbf{x}_\ell \boldsymbol{\beta} + \text{offset}_\ell)} \right\}$$

These are for each of the covariates, $u = 1, \dots, p$.

David Roxbee Cox (1924–) was born in Birmingham, England. He earned Ph.D. degrees in mathematics and statistics from the universities of Cambridge and Leeds and worked at the Royal Aircraft Establishment, the Wool Industries Research Association, and the universities of Cambridge, London (Birkbeck and Imperial Colleges), and Oxford. He was knighted in 1985. Sir David has worked on a wide range of theoretical and applied statistical problems, with outstanding contributions in areas such as experimental design, stochastic processes, binary data, survival analysis, asymptotic techniques, and multivariate dependencies.

Acknowledgment

We thank Peter Sasieni of Cancer Research UK for his statistical advice and guidance in implementing the robust variance estimator for this command.

References

- Box-Steffensmeier, J. M., and B. S. Jones. 2004. *Event History Modeling: A Guide for Social Scientists*. Cambridge: Cambridge University Press.
- Breslow, N. E. 1974. Covariance analysis of censored survival data. *Biometrics* 30: 89–99.
- Cleves, M. A. 1999. ssa13: Analysis of multiple failure-time data with Stata. *Stata Technical Bulletin* 49: 30–39. Reprinted in *Stata Technical Bulletin Reprints*, vol. 9, pp. 338–349.
- Cleves, M. A., W. W. Gould, and R. G. Gutierrez. 2004. *An Introduction to Survival Analysis Using Stata*. Rev. ed. College Station, TX: Stata Press.
- Cox, D. R. 1972. Regression models and life-tables (with discussion). *Journal of the Royal Statistical Society, Series B* 34: 187–220.
- . 1975. Partial likelihood. *Biometrika* 62: 269–276.
- Cox, D. R., and D. Oakes. 1984. *Analysis of Survival Data*. London: Chapman & Hall/CRC.
- Cox, D. R., and E. J. Snell. 1968. A general definition of residuals (with discussion). *Journal of the Royal Statistical Society, Series B* 30: 248–275.
- Crowley, J., and M. Hu. 1977. Covariance analysis of heart transplant survival data. *Journal of the American Statistical Association* 72: 27–36.
- Cui, J. 2005. Buckley–James method for analyzing censored data, with an application to a cardiovascular disease and an HIV/AIDS study. *Stata Journal* 5: 517–526.
- Dupont, W. D. 2002. *Statistical Modeling for Biomedical Researchers: A Simple Introduction to the Analysis of Complex Data*. Cambridge: Cambridge University Press.
- Fleming, T. R., and D. P. Harrington. 1991. *Counting Processes and Survival Analysis*. New York: Wiley.
- Gutierrez, R. G. 2002. Parametric frailty and shared frailty survival models. *Stata Journal* 2: 22–44.
- Gutierrez, R. G., S. L. Carter, and D. M. Drukker. 2001. On boundary-value likelihood ratio tests. *Stata Technical Bulletin* 60: 15–18. Reprinted in *Stata Technical Bulletin Reprints*, vol. 10, pp. 269–273.
- Hills, M., and B. L. De Stavola. 2006. *A Short Introduction to Stata for Biostatistics*. London: Timberlake.
- Hosmer, D. W., Jr., and S. Lemeshow. 1999. *Applied Survival Analysis: Regression Modeling of Time to Event Data*. New York: Wiley.
- Jenkins, S. P. 1997. sbe17: Discrete time proportional hazards regression. *Stata Technical Bulletin* 39: 22–32. Reprinted in *Stata Technical Bulletin Reprints*, vol. 7, pp. 109–121.
- Kahn, H. A., and C. T. Sempos. 1989. *Statistical Methods in Epidemiology*. New York: Oxford University Press.
- Kalbfleisch, J. D., and R. L. Prentice. 2002. *The Statistical Analysis of Failure Time Data*. 2nd ed. New York: Wiley.
- Klein, J. P., and M. L. Moeschberger. 2003. *Survival Analysis: Techniques for Censored and Truncated Data*. 2nd ed. New York: Springer.
- Lin, D. Y., and L. J. Wei. 1989. The robust inference for the Cox proportional hazards model. *Journal of the American Statistical Association* 84: 1074–1078.
- McGilchrist, C. A., and C. W. Aisbett. 1991. Regression with frailty in survival analysis. *Biometrics* 47: 461–466.
- Newman, S. C. 2001. *Biostatistical Methods in Epidemiology*. New York: Wiley.
- Peto, R. 1972. Contribution to the discussion of paper by D. R. Cox. *Journal of the Royal Statistical Society, Series B* 34: 205–207.
- Reid, N. 1994. A conversation with Sir David Cox. *Statistical Science* 9: 439–455.
- Royston, J. P. 2001. Flexible parametric alternatives to the Cox model, and more. *Stata Journal* 1: 1–28.
- . 2006. Explained variation for survival models. *Stata Journal* 6: 83–96.

- Schoenfeld, D. 1982. Partial residuals for the proportional hazards regression model. *Biometrika* 69: 239–241.
- Selvin, S. 2004. *Statistical Analysis of Epidemiologic Data*. 3rd ed. New York: Oxford University Press.
- Sterne, J. A. C., and K. Tilling. 2002. G-estimation of causal effects, allowing for time-varying confounding. *Stata Journal* 2: 164–182.
- Therneau, T. M., and P. M. Grambsch. 2000. *Modeling Survival Data: Extending the Cox Model*. New York: Springer.
- Vittinghoff, E., D. V. Glidden, S. C. Shiboski, and C. E. McCulloch. 2005. *Regression Methods in Biostatistics: Linear, Logistic, Survival, and Repeated Measures Models*. New York: Springer.

Also See

- [ST] **stcox postestimation** — Postestimation tools for stcox
- [ST] **stcurve** — Plot survivor, hazard, or cumulative hazard function
- [ST] **stcox diagnostics** — stcox diagnostic plots
- [ST] **sts** — Generate, graph, list, and test the survivor and cumulative hazard functions
- [ST] **stset** — Declare data to be survival-time data
- [ST] **streg** — Fit parametric survival models
- [SVY] **svy estimation** — Estimation commands for survey data
- [U] **20 Estimation and postestimation commands**