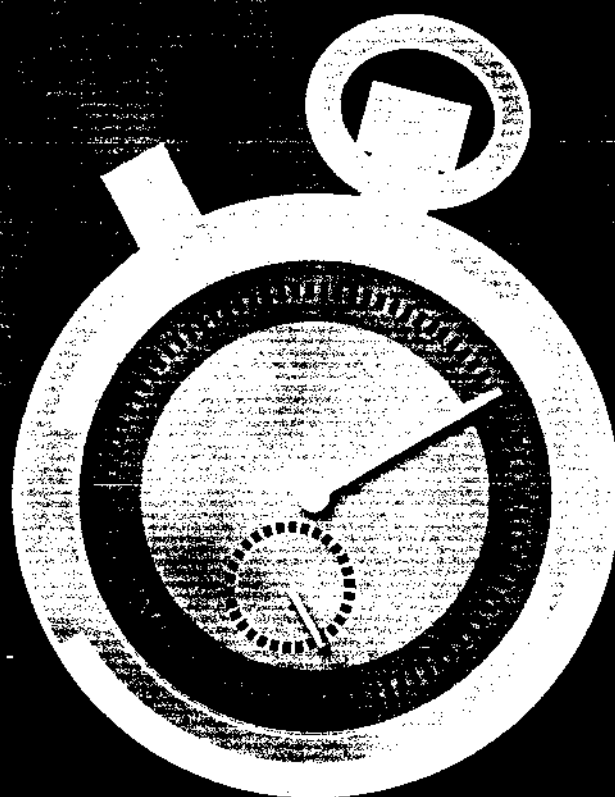




AN INTRODUCTION TO SURVIVAL ANALYSIS USING STATA®

REVISED EDITION



**MARIO A. CLEVES
WILLIAM W. GOULD
ROBERTO G. GUTIERREZ**

Stata
Press

An Introduction to Survival Analysis Using Stata

Revised Edition

An Introduction to Survival Analysis Using Stata

Revised Edition

MARIO A. CLEVES

*University of Arkansas Medical Sciences
Department of Pediatrics*

WILLIAM W. GOULD

Stata Corporation

ROBERTO G. GUTIERREZ

Stata Corporation



A Stata Press Publication
STATA CORPORATION
College Station, Texas

Stata Press, 4905 Lakeway Drive, College Station, Texas 77845

Copyright© 2004 by Stata Corporation
All rights reserved
Typeset using L^AT_EX2_ε
Printed in the United States of America
10 9 8 7 6 5 4 3 2 1

ISBN 1-881228-84-3

This book is protected by copyright. All rights are reserved. No part of this book may be reproduced, stored in a retrieval system, or transcribed, in any form or by any means—electronic, mechanical, photocopying, recording, or otherwise—without the prior written permission of Stata Corporation (StataCorp).

Stata is a registered trademark of Stata Corporation. L^AT_EX is a trademark of the American Mathematical Society.

Contents

Preface to the revised edition	xv
Preface	xvii
Notation and Typography	xxi
1 The problem of survival analysis	1
1.1 Parametric modeling	2
1.2 Semiparametric modeling	3
1.3 Nonparametric analysis	5
1.4 Linking the three approaches	5
2 Describing the distribution of failure times	7
2.1 The survivor and hazard functions	7
2.2 The quantile function	11
2.3 Interpreting the hazard and cumulative hazard	13
2.3.1 Interpreting the cumulative hazard	14
2.3.2 Interpreting the hazard rate	16
2.4 Means and medians	16
3 Hazard models	19
3.1 Parametric models	20
3.2 Semiparametric models	21
3.3 Analysis time (time at risk)	24
4 Censoring and truncation	29
4.1 Censoring	29

4.1.1	Right censoring	30
4.1.2	Interval censoring	32
4.1.3	Left censoring	34
4.2	Truncation	34
4.2.1	Left truncation (delayed entry)	35
4.2.2	Interval truncation (gaps)	36
4.2.3	Right truncation	36
5	Recording survival data	37
5.1	The desired format	37
5.2	Other formats	41
5.3	Example	45
6	Using stset	47
6.1	A short lesson on dates	48
6.2	The purpose of the stset command	50
6.3	The syntax of the stset command	51
6.3.1	Specifying analysis time	52
6.3.2	Variables defined by stset	55
6.3.3	Specifying what constitutes failure	57
6.3.4	Specifying when subjects exit from the analysis	59
6.3.5	Specifying when subjects enter the analysis	62
6.3.6	Specifying the subject-id variable	65
6.3.7	Specifying the begin-of-span variable	66
6.3.8	Convenience options	69
7	After stset	73
7.1	Look at stset's output	73
7.2	List some of your data	76
7.3	Use stdes	77
7.4	Use stvary	78

7.5	Perhaps use stfill	80
7.6	Example: Hip fracture data	82
8	Nonparametric analysis	91
8.1	Inadequacies of standard univariate methods	91
8.2	The Kaplan–Meier estimator	93
8.2.1	Calculation	93
8.2.2	Censoring	96
8.2.3	Left truncation (delayed entry)	97
8.2.4	Interval truncation (gaps)	99
8.2.5	Relationship to the empirical distribution function	99
8.2.6	Other uses of sts list	101
8.2.7	Graphing the Kaplan–Meier estimate	102
8.3	The Nelson–Aalen estimator	105
8.4	Estimating the hazard function	111
8.5	Tests of hypothesis	113
8.5.1	The log-rank test	113
8.5.2	The Wilcoxon test	116
8.5.3	Other tests	116
8.5.4	Stratified tests	117
9	The Cox proportional hazards model	121
9.1	Using stcox	122
9.1.1	The Cox model has no intercept	123
9.1.2	Interpreting coefficients	123
9.1.3	The effect of units on coefficients	125
9.1.4	Estimating the baseline cumulative hazard and survivor functions	127
9.1.5	Estimating the baseline hazard function	131
9.1.6	The effect of units on the baseline functions	134
9.2	Likelihood calculations	136
9.2.1	No tied failures	137

9.2.2	Tied failures	140
	The marginal calculation	140
	The partial calculation	141
	The Breslow approximation	142
	The Efron approximation	143
9.2.3	Summary	143
9.3	Stratified analysis	144
9.3.1	Obtaining coefficient estimates	144
9.3.2	Obtaining estimates of baseline functions	146
9.4	Cox models with shared frailty	147
9.4.1	Parameter estimation	148
9.4.2	Obtaining estimates of baseline functions	152
10	Model building using stcox	157
10.1	Indicator variables	157
10.2	Categorical variables	158
10.3	Continuous variables	160
10.4	Interactions	162
10.5	Time-varying variables	166
10.5.1	Using stcox, tvco(), texp()	167
10.5.2	Using stsplot	169
11	The Cox model: Diagnostics	175
11.1	Testing the proportional hazards assumption	175
11.1.1	Tests based on re-estimation	175
11.1.2	Test based on Schoenfeld residuals	178
11.1.3	Graphical methods	181
11.2	Residuals	184
	Reye's syndrome data	185
11.2.1	Determining functional form	186
11.2.2	Goodness of fit	190
11.2.3	Outliers and influential points	193

12 Parametric models	197
12.1 Motivation	197
12.2 Classes of parametric models	200
12.2.1 Parametric proportional hazards models	201
12.2.2 Accelerated failure-time models	207
12.2.3 Comparing the two parameterizations	209
 13 A survey of parametric regression models in Stata	 213
13.1 The exponential model	215
13.1.1 Exponential regression in the PH metric	215
13.1.2 Exponential regression in the AFT metric	222
13.2 Weibull regression	224
13.2.1 Weibull regression in the PH metric	224
Fitting null models	229
13.2.2 Weibull regression in the AFT metric	233
13.3 Gompertz regression (PH metric)	234
13.4 Log-normal regression (AFT metric)	237
13.5 Log-logistic regression (AFT metric)	240
13.6 Generalized gamma regression (AFT metric)	244
13.7 Choosing among parametric models	246
13.7.1 Nested models	246
13.7.2 Non-nested models	249
 14 Post-estimation commands for parametric models	 251
14.1 Use of predict after streg	251
14.1.1 Predicting the time of failure	253
14.1.2 Predicting the hazard and related functions	259
14.1.3 Calculating residuals	262
14.2 Using stcurve	264

15 Generalizing the parametric regression model	269
15.1 Using the ancillary() option	269
15.2 Stratified models	275
15.3 Frailty models	278
15.3.1 Unshared frailty models	279
15.3.2 Kidney data	280
15.3.3 Testing for heterogeneity	285
15.3.4 Shared frailty models	292
References	299
Author index	303
Subject index	305

List of Tables

8.1	$r \times 2$ contingency table for time t_j	114
9.1	Comparison of methods for handling ties	143
9.2	Comparison of various models for hip-fracture data	146
13.1	Comparison of streg models	249
13.2	Comparison of AIC values for streg models	250
14.1	Options for predict after streg	251
14.2	Use of predict , surv and predict , csurv	261

List of Figures

2.1	Examples of hazard functions obtained from various parametric survival models	10
8.1	Kaplan–Meier estimate for hip-fracture data	103
8.2	Kaplan–Meier estimates for treatment vs. control	104
8.3	Kaplan–Meier with the number of censored observations	105
8.4	Nelson–Aalen curves for treatment vs. control	108
8.5	Estimated survivor functions	109
8.6	Estimated cumulative hazard functions	110
8.7	Smoothed hazard functions	112
8.8	Smoothed hazard functions, log scale	113
9.1	Estimated baseline cumulative hazard	128
9.2	Estimated cumulative hazard: treatment vs. controls	129
9.3	Estimated baseline survivor function	129
9.4	Estimated survivor: treatment vs. controls	130
9.5	Estimated baseline hazard function	132
9.6	Estimated hazard functions, treatment vs. control	133
9.7	Log likelihood for the Cox model	139
9.8	Estimated baseline cumulative hazard for males vs. females	147
9.9	Comparison of survival curves for various frailty values	154
9.10	Comparison of hazards for various frailty values	155
11.1	Test of the proportional hazards assumption for age	181
11.2	Test of proportional hazards assumption for protect	182
11.3	Test of proportional hazards assumption for protect , given age	183
11.4	Comparison of Kaplan–Meier and Cox survivor functions	184

11.5	Finding the functional form for <code>ammonia</code>	188
11.6	Using the log transformation	188
11.7	Cumulative hazard of Cox-Snell residuals (<code>ammonia</code>)	192
11.8	Cumulative hazard of Cox-Snell residuals (<code>lamm</code>)	193
11.9	<code>Dfbeta(sgot)</code> for Reye's syndrome data	195
13.1	Estimated baseline hazard function	218
13.2	Estimated baseline hazard function using $\ln(\text{myt} + 1)$	219
13.3	Estimated baseline hazard function using a step function	220
13.4	Estimated baseline hazard function using a better step function	221
13.5	Comparison of estimated baseline hazards	222
13.6	Weibull hazard function for various p	225
13.7	Estimated baseline hazard function for Weibull model	228
13.8	Comparison of exponential (step) and Weibull hazards	229
13.9	Estimated Weibull hazard functions over values of <code>protect</code>	231
13.10	Gompertz hazard functions	235
13.11	Estimated baseline hazard for the Gompertz model	237
13.12	Examples of log-normal hazard functions ($\beta_0 = 0$)	238
13.13	Comparison of hazards for a log-normal model	240
13.14	Examples of log-logistic hazard functions ($\beta_0 = 0$)	242
14.1	Cumulative Cox-Snell residuals for a Weibull model	263
14.2	Hazard curves for the gamma model (<code>protect==0</code>)	265
14.3	Hazard curves for the gamma model (<code>protect==1</code>)	266
14.4	Cumulative survival probability as calculated by <code>predict</code>	267
14.5	Survival function as calculated by <code>stcurve</code>	268
15.1	Comparison of baseline hazards for males and females	276
15.2	Comparison of log-normal hazards for males and females	286
15.3	Comparison of Weibull/gamma population hazards	291
15.4	Comparison of Weibull/gamma individual ($\alpha_j = 1$) hazards	292
15.5	Comparison of piecewise-constant individual ($\alpha = 1$) hazards	298

Preface to the revised edition

This revised edition updates the original text (written to support Stata 7) to reflect Stata 8, which was released in January 2003. Most of the changes are minor and include new graphics, including the appearance of the graphics and the syntax used to create them, and updated datasets.

New sections describe Stata's ability to graph nonparametric and semiparametric estimates of hazard functions. Stata now calculates estimated hazards as weighted kernel-density estimates of the times at which failures occur, where weights are the increments of the estimated cumulative hazard function. These new capabilities are described in terms of nonparametric estimation in Chapter 8 and in terms of Cox regression (Chapter 9).

Another added section in Chapter 9 discusses Stata's ability to apply shared frailty to the Cox model. This section complements the discussion of parametric shared and unshared frailty models in Chapter 8. Because the frailty is best understood by beginning with a parametric model, this new section is relatively brief and focuses only on practical issues of estimation and interpretation.

Mario A. Cleves
William W. Gould
Roberto G. Gutierrez

College Station, Texas
August 2003

Preface

We have written this book for professional researchers outside the field of mathematics, people who do not spend their time wondering about the intricacies of generalizing a result from discrete space to \mathbb{R}_1 but who, nonetheless, understand statistics. Our readers may sometimes be sloppy when they say that a probability density is a probability, but when pressed, they know there is a difference and remember that a probability density can indeed even be greater than one. However, our readers are never sloppy when it comes to their science. Our readers use statistics as a tool, just as they use mathematics, and just as they sometimes use computer software.

This is a book about survival analysis for the professional data analyst, whether a health scientist, an economist, a political scientist, or any of a wide range of scientists who have found that survival analysis is applicable to their problems. This is a book for researchers who want to understand what they are doing and to understand the underpinnings and assumptions of the tools they use; in other words, this is a book for all researchers.

This book grew out of software, but nonetheless, it is not a manual. That genesis, however, gives this book an applied outlook that is sometimes missing from other works. We, the authors of this book, are also the authors of Stata's survival analysis commands, which have had something more than modest success. Writing application software places a discipline on authors not unlike that of building of scientific machines by engineers. Problems that might be swept under the rug as mere details cannot be ignored, in the construction of software, and the authors are often reminded that the devil is in the details. It is those details that cause users such grief, such confusion, and sometimes, such pleasure.

In addition to having written the software, we have all been involved in supporting it, which is to say, interacting with users (real professionals). We have seen the software used in ways that we would never have imagined, and we have seen the problems that arise in such uses. Those problems are often not simply programming issues but involve statistical issues that have given us pause. To the statisticians in the audience, we mention that there is nothing like embedding yourself in the problems of real researchers to teach you that problems you thought unimportant are of great importance, and vice versa. There is nothing like "straightforwardly generalizing" some procedure to teach you that there are subtle issues worth lots of thought.

In this book, we illustrate the concepts of using Stata. Readers should expect a certain bias on our part, but the concepts go beyond our implementation of them. We

will often discuss substantive issues right in the midst of issues of computer use, and we do that because, in real life, that is where they arise.

This book also grew out of a course we taught several times over the web, and the many researchers who took that course will find in this book the companion text they lamented not having for that course.

We do not wish to promise more than we can deliver, but the reader of this book should come away not just with an understanding of the formulas, but an intuition of how the various survival analysis estimators work and exactly what information they exploit.

We would like to thank all the people who over the years have contributed to our understanding of survival analysis and the improvement of Stata's survival capabilities be it through programs, comments, or suggestions. We are particularly grateful to

David Clayton of the Cambridge Institute for Medical Research
Joanne M. Garrett of the University of North Carolina
Michael Hills retired from the London School of Hygiene and Tropical Medicine
David Hosmer, Jr. of the University of Massachusetts, Amherst
Stephen P. Jenkins of the University of Essex
Stanley Lemeshow of Ohio State University
Adrian Mander of the MRC Biostatistics Unit
William H. Rogers of The Health Institute at New England Medical Center
Patrick Royston of the MRC Clinical Trials Unit
Peter Sasieni of Cancer Research UK
Jeroen Weesie of Utrecht University

By no means is this list complete; we would like to express our thanks as well to all those who should have been listed.

Mario A. Cleves
William W. Gould
Roberto G. Gutierrez

College Station, Texas
May 2002

Notation and Typography

This book is an introduction to the analysis of survival data using Stata, and we are going to assume that you are already familiar, more or less, with Stata.

For instance, if you had some raw data on outcomes after surgery, and we tell you to (1) enter it into Stata, (2) sort the data on patient's age, (3) save the data, (4) list the age and outcomes for the 10 youngest and 10 oldest patients in the data, (5) tell us the overall fraction of observed deaths, and (6) tell us the median time to death among those who died, you could do that. To wit,

```
. infile ...  
. sort age  
. save mydata  
. list age outcome in 1/10  
. list age outcome in -10/1  
. summarize died  
. summarize time if died, detail
```

This text was written using Stata 8, and in order to ensure that you can fully replicate what we have done, you need an up-to-date Stata version 8 or later. Type

```
. update query
```

from a web-aware Stata and follow the instructions to ensure that you are up to date.

The developments in this text are largely applied, and it is our intention that you read this text while sitting at a computer so that you can try for yourself the sequences of commands contained in the text to replicate our results. In this way, you may generalize these sequences to suit your own data analysis needs.

We use the typewriter font `command` to refer to Stata commands, syntax, and variables. When a “dot” prompt is displayed followed by a command (such as in the above sequence), it means you can type verbatim what is displayed after the dot (in context) to replicate the results in the book.

With the exception of some very small expository datasets we use, all the data we use in this text are freely available for you to download (via a web-aware Stata) from the Stata Press website, <http://www.stata-press.com>. In fact, when we introduce new datasets, we merely load them into Stata the same way that you would. For example,

```
. use http://www.stata-press.com/data/cgg/hip, clear /* hip-fracture data */
```

Feel free to try this for yourself. The `cgg` part of the pathname, in case you are curious, is comprised of the last initial of each of the three authors.

This text serves as a complement to the material in the Stata manuals, not as a substitute, and thus we often make reference to the material in the Stata manuals using the [R], [P], etc. notation. For example, [R] xi refers to the *Stata Base Reference Manual* entry for xi, and [P] **syntax** refers to the entry for **syntax** in the *Stata Programming Reference Manual*.

Survival analysis, as with most substantive fields, is a field full of jargon: left truncation, right censoring, hazard rates, cumulative hazard, survivor function, etc. Jargon arises so that researchers do not have to explain the same concepts over and over again. Those of you who practice survival analysis know that researchers tend to be a little sloppy in their use of language, saying truncation when they mean censoring or hazard when they mean cumulative hazard, and if we are going to communicate by the written word, we have to agree on what these terms mean. Moreover, these words form a wall around the field that is nearly impenetrable if you are not already a member of the cognoscenti.

If you are new to survival analysis, let us reassure you: Survival analysis is statistics. Master the jargon and think carefully, and you can do this.

1 The problem of survival analysis

Survival analysis is concerned with analyzing the time to the occurrence of an event. For instance, we have a dataset in which the times are 1, 5, 9, 20, and 22. Perhaps those measurements are made in seconds, perhaps in days, but that does not matter. Perhaps the event is the time until a generator's bearings seize, the time until a cancer patient dies, or the time until a person finds employment, but that does not matter either.

For now, we will just abstract the underlying data-generating process and say that we have some times until an event occurs, and that those times are 1, 5, 9, 20, and 22. In addition, perhaps we have some covariates (additional variables) that we wish to use to “explain” these times. So, pretend that we have the following (completely made up) dataset:

time	x
1	3
5	2
9	4
20	9
22	10

Now, what is to keep us from simply analyzing these data using ordinary least-squares (OLS) linear regression? Why not simply fit the model

$$\text{time}_j = \beta_0 + \beta_1 x_j + \epsilon_j, \quad \epsilon_j \sim N(0, \sigma^2)$$

for $j = 1, \dots, 5$, or, alternatively,

$$\ln(\text{time}_j) = \beta_0 + \beta_1 x_j + \epsilon_j, \quad \epsilon_j \sim N(0, \sigma^2)$$

That is easy enough to do in Stata by typing

```
. regress time x
```

or

```
. generate lntime = ln(time)
. regress lntime x
```

These days, researchers would seldom analyze survival times in this manner, but why not? Before you answer too dismissively, we warn you that we, the authors, can think of problems for which this would be a perfectly reasonable model to use.

1.1 Parametric modeling

The problem with using OLS to analyze survival data lies with the assumed distribution of the residuals, ϵ_j . In linear regression, the residuals are assumed to be distributed normally, which is to say, time conditional on x_j is assumed to follow a normal distribution:

$$\text{time}_j \sim N(\beta_0 + \beta_1 x_j, \sigma^2), \quad j = 1, \dots, 5$$

The simple fact is that the assumed normality of time to an event is unreasonable for many events. It is unreasonable, for instance, if we are thinking about an event that has an instantaneous risk of occurring that is constant over time. In that case, the distribution of time would follow an exponential distribution. It is also unreasonable if we are analyzing survival times following a particularly serious surgical procedure. In that case, the distribution might have two modes: many patients die shortly after the surgery, but if they survive, the disease might be expected to return. One other problem is that a time to failure is always positive, while theoretically, the normal distribution is supported on the entire real line. Realistically, however, this fact alone is not enough to render the normal distribution useless in this context, since σ^2 may be chosen (or estimated) to make the probability of a negative failure time virtually zero.

At its core, survival analysis concerns nothing more than making a substitution for the normality assumption characterized by OLS with something more appropriate for the problem at hand.

Perhaps, if you were already familiar with survival analysis, when we asked “why not linear regression?”, you offered the excuse of right censoring—that in real data we often do not observe subjects long enough for all of them to fail. In our data, however, there was no censoring, and really, censoring is just a nuisance. We can fix linear regression easily enough to deal with right censoring. It goes under the name censored normal regression, and Stata’s `cureg` command can fit such models; see [R] `tobit`. The real problem with linear regression in survival applications is with the assumed normality.

Not being already familiar with survival analysis, you might be tempted to use linear regression in the face of non-normality. Linear regression is known, after all, to be remarkably robust to deviations from normality, so why not just use it anyway? The problem is that the distributions for time to an event might be quite dissimilar from the normal—they are almost certainly nonsymmetric, they might be bimodal, and linear regression is not robust to these violations.

Substituting a more reasonable distributional assumption for ϵ_j leads to parametric survival analysis.

1.2 Semiparametric modeling

That results of analyses are being determined by the assumptions and not the data is always a source of concern, and this leads to a search for methods that do not require assumptions about the distribution of failure times. That, at first blush, seems hopeless. With survival data, the key insight into removing the distributional assumption is that, because events occur at given times, these events may be ordered and the analysis may be performed using the ordering of the survival times exclusively. Consider our dataset:

time	x
1	3
5	2
9	4
20	9
22	10

Examine the failure that occurred at time 1. Let's ask, "What is the probability of failure after exposure to the risk of failure for 1 unit of time?" At this point, observation 1 had failed, and the others had not. This reduces the problem to a problem of binary-outcome analysis,

time	x	outcome
1	3	1
5	2	0
9	4	0
20	9	0
22	10	0

and it would be perfectly reasonable for us to analyze failure at `time = 1` using, say, logistic regression

$$\begin{aligned}
 &= \Pr(\text{failure after exposure for 1 unit of time}) \\
 &= \Pr(\text{outcome}_j = 1) \\
 &= \frac{1}{1 + \exp(-\beta_0 - x_j\beta_x)}
 \end{aligned}$$

for $j = 1, \dots, 5$. This is easy enough to do:

```
. logistic outcome`x`
```

Do not make too much out of our choice of logistic regression—choose the analysis method you like. Use probit. Make a table. The point is that whatever particular technique you choose, you could do all your survival analysis using this analyze-the-first-failure method. It would be a mightily inefficient use of your data, but it would have the advantage that you would be making no assumptions about the distribution of failure times. Of course, you would have to give up on the idea of being able to make predictions conditional on x , but perhaps being able to predict whether failure occurs at `time = 1` would be sufficient.

There is nothing magical about the first death time; we could instead choose to analyze the second death time, which, it turns out in these data, is `time = 5`. We could

ask about the probability of failure, given exposure of 5 units of time, in which case we would exclude the first observation (which failed too early) and fit our logistic regression model using the second and subsequent observations:

```
. drop outcome
. generate outcome = cond(time==5,1,0) if time>=5
. logistic outcome x if time>=5
```

In fact, we could use this same procedure on each of the death times, separately.

Which analysis should we use? Well, there is slightly less information in the second analysis than in the first (because we have one less observation), and in the third than in the first two (for the same reason), and so on, so we should choose the first. It is, however, unfortunate, that we have to choose at all. Could we somehow combine all of these analyses and constrain the appropriate regression coefficients (say the coefficient on x) to be the same? The answer is yes, we could, and after some math, that leads to semiparametric survival analysis and, in particular, to Cox (1972) regression if a conditional logistic model is fit for each analysis. Conditional logistic models differ from ordinary logistic models for this example in that for the former we condition on the fact that we know that `outcome==1` for one and only one observation within each separate analysis.

However, for now we don't want to get lost in all the mathematical detail. What is important is that we could have done each of the analyses using whatever binary analysis method seemed appropriate. By doing so, we could combine them all if we are sufficiently clever in doing the math, and since each of the separate analyses made no assumption about the distribution of failure times, the combined analysis also makes no such assumption.

That last statement is rather slippery, so it does not hurt to verify its truth. We have been considering the data,

time	x
1	3
5	2
9	4
20	9
22	10

but now consider two variations on the data:

time	x
1.1	3
1.2	2
1.3	4
50.0	9
50.1	10

and

time	x
1	3
500	2
1000	4
10000	9
100000	10

These two alternatives have dramatically different distributions for time yet have the same temporal ordering and the same values of x . Think about performing the individual analyses on each of these datasets, and you will realize that the results you get will be exactly the same. Time plays no role other than ordering the observations.

The methods described above go under the name semiparametric analysis because, as far as time is concerned, they are nonparametric, but since we are still parameterizing the effect of x , there exists a parametric component to the analysis.

1.3 Nonparametric analysis

Semiparametric models are parametric in the sense that the effect of the covariates is still assumed to take a certain form. In the previous section, by performing a separate analysis at each failure time and concerning ourselves only with the order in which the failures occurred, we made no assumption about the distribution of time to failure. We did, however, make an assumption about how each subject's observed x value determined the probability that that subject would fail; for example, a probability determined by the logistic function.

An entirely nonparametric approach would be to do away with this assumption also and follow the philosophy of "letting the dataset speak for itself". There exists a vast literature on performing nonparametric regression using methods such as lowess or local polynomial regression; however, such methods do not adequately deal with censoring and other issues unique to survival data.

When no covariates exist, or when the covariates are qualitative in nature (gender, for instance), we can use nonparametric methods such as Kaplan and Meier (1958) or the method of Nelson (1972) and Aalen (1978) to estimate the probability of survival past a certain point in time, or to compare the survival experiences for each gender. These methods take into account censoring and other characteristics of survival data. There also exist methods such as the two-sample log-rank test, which can compare the survival experience across gender by using only the temporal ordering of the failure times. To wit, nonparametric methods make assumptions about neither (a) the distribution of the failure times nor (b) how covariates serve to change or shift the survival experience.

1.4 Linking the three approaches

Going back to our original data, consider the individual analyses we performed in order to obtain the semiparametric (combined) results. The individual analyses were

Pr(failure after exposure for (exactly) 1 unit of time)
 Pr(failure after exposure for (exactly) 5 units of time)
 Pr(failure after exposure for (exactly) 9 units of time)
 Pr(failure after exposure for (exactly) 20 units of time)
 Pr(failure after exposure for (exactly) 22 units of time)

We could omit any of the individual analyses above, and doing so would only affect the efficiency of our estimators. It is better, though, to include them all, so why not add the following to this list:

Pr(failure after exposure for (exactly) 1.1 units of time)
 Pr(failure after exposure for (exactly) 1.2 units of time)

...

That is, why not add individual analyses for all other times between the observed failure times? That would be a good idea because the more analyses we can combine, the more efficient our final results will be, which is to say that the standard errors of our estimated regression parameters will be smaller. The only reason we do not do this is that we do not know how to say anything about these intervening times—we do not know how to perform these analyses—unless we make an assumption about the distribution of failure time. If we made that assumption, we could perform the intervening analyses (the infinite number of them), and then we could combine them all to get super-efficient estimates. We could perform the individual analyses themselves a little differently, too, by taking into account the distributional assumptions, but that would only make our final analysis even more efficient.

That is the link between semiparametric and parametric analysis. Semiparametric analysis is nothing more than a combination of separate binary-outcome analyses, one per failure time, while parametric analysis is a combination of several analyses at *all* possible failure times. In parametric analysis, if no failures occur over a particular interval of time, that is informative. In semiparametric analysis, such periods are not informative. On the one hand, semiparametric analysis is advantageous in that it does not concern itself with the intervening analyses, yet parametric analysis will be more efficient if the proper distributional assumptions are made concerning those times when no failures are observed.

When no covariates are present, we hope that semiparametric methods such as Cox regression will produce estimates of relevant quantities (such as the probability of survival past a certain time) that are identical to the nonparametric estimates, and in fact, they do. When the covariates are qualitative in nature, parametric and semiparametric methods should yield more efficient tests and comparisons of the groups determined by the covariates than nonparametric methods, and these tests should agree. Should the tests disagree, this would serve as a signal that some of the assumptions made by the parametric or semiparametric models are incorrect.

2 Describing the distribution of failure times

The key to mastering survival analysis lies in grasping the jargon, and in this chapter and the next, we describe the statistical terms unique to the analysis of survival data.

2.1 The survivor and hazard functions

These days, survival analysis is cast in a language all its own. Let T be a non-negative random variable denoting the time to a failure event. Rather than referring to T 's probability density function $f(t)$ —or, if you prefer, its cumulative distribution function $F(t) = \Pr(T \leq t)$ —survival analysts instead talk about T 's survivor function $S(t)$ or its hazard function $h(t)$. There is good reason for this: it really is more convenient to think in terms of $S(t)$ and $h(t)$ rather than $F(t)$ or $f(t)$, although all forms describe exactly the same probability distribution for T . Translating between these four forms is quite simple.

The survivor function, also called the survivorship function or the survival function, is nothing more than the reverse cumulative distribution function of T :

$$S(t) = 1 - F(t) = \Pr(T > t)$$

The survivor function reports the probability of surviving beyond time t . Said differently, it is the probability that there is no failure event prior to t . The function is equal to one at $t = 0$ and decreases toward zero as t goes to infinity. (The survivor function is a monotone, nonincreasing function of time.)

The density function $f(t)$ can be obtained as easily from $S(t)$ as it can from $F(t)$,

$$f(t) = \frac{dF(t)}{dt} = \frac{d}{dt}\{1 - S(t)\} = -S'(t)$$

The hazard function $h(t)$ —also known as the conditional failure rate, the intensity function, the age-specific failure rate, the inverse of the Mills' ratio, and the force of mortality—is the instantaneous rate of failure, with the emphasis on the word *rate*, meaning that it has units $1/t$. It is the (limiting) probability that the failure event occurs in a given interval, conditional upon the subject having survived to the beginning of that interval, divided by the width of the interval:

$$h(t) = \lim_{\Delta t \rightarrow 0} \frac{\Pr(t + \Delta t > T > t | T > t)}{\Delta t} = \frac{f(t)}{S(t)} \quad (2.1)$$

The hazard rate (or function) can vary from zero (meaning no risk at all) to infinity (meaning the certainty of failure at that instant). Over time, the hazard rate can increase, decrease, remain constant, or even take on more serpentine shapes. There is a one-to-one relationship between the probability of survival past a certain time and the amount of risk that has been accumulated up to that time, and the hazard rate measures the rate at which risk is accumulated. The hazard function is at the heart of modern survival analysis, and it is well worth the effort to become familiar with this function.

It is, of course, the underlying process (disease, manufacturing a product, etc.) that determines the shape of the hazard function:

1. When the risk of something is zero, its hazard is zero.
2. We have all heard of risks that do not vary over time. That does not mean that, as I view my future prospects, my chances of having succumbed to the risk do not increase with time. Indeed, it is certain that I will succumb eventually (provided that the constant risk or hazard is nonzero), but my chances of succumbing at this instant or that are all the same.
3. If the risk is rising with time, so is the hazard. In such a case, the future is indeed bleak.
4. If the risk is falling with time, so is the hazard. In this case, the future looks better (if only we can make it through the present).
5. The human mortality pattern related to aging generates a falling hazard for a while after birth, then a long, flat plateau, and thereafter constantly rising and eventually reaching, one supposes, values near infinity at about 100 years. This is often called the "bathtub hazard" by biometricians.
6. The risk of post-operative wound infection falls as time from surgery increases, so the hazard function decreases with time.

Given one of the four functions that describe the probability distribution of failure times, the other three are completely determined. In particular, one may derive from a hazard function the probability density function, the cumulative distribution function, and the survivor function. To show this, it is first convenient to define yet another function, the cumulative hazard function,

$$H(t) = \int_0^t h(u) du$$

and thus

$$H(t) = \int_0^t \frac{f(u)}{S(u)} du = - \int_0^t \frac{1}{S(u)} \left\{ \frac{d}{du} S(u) \right\} du = - \ln\{S(t)\} \quad (2.2)$$

Note that the cumulative hazard function has an interpretation all its own—it measures the total amount of risk that has been accumulated up to time t , and from (2.2) we can see the relationship between accumulated risk and the probability of survival.

We can now conveniently write

$$\begin{aligned} S(t) &= \exp\{-H(t)\} \\ F(t) &= 1 - \exp\{-H(t)\} \\ f(t) &= h(t) \exp\{-H(t)\} \end{aligned}$$

As an example, consider the Weibull hazard function, often used by engineers,

$$h(t) = pt^{p-1}$$

where p is a shape parameter estimated from the data. Given this form of the hazard, we can determine the survivor, cumulative distribution, and probability density functions to be

$$\begin{aligned} S(t) &= \exp(-t^p) \\ F(t) &= 1 - \exp(-t^p) \\ f(t) &= pt^{p-1} \exp(-t^p) \end{aligned} \quad (2.3)$$

(Continued on next page)

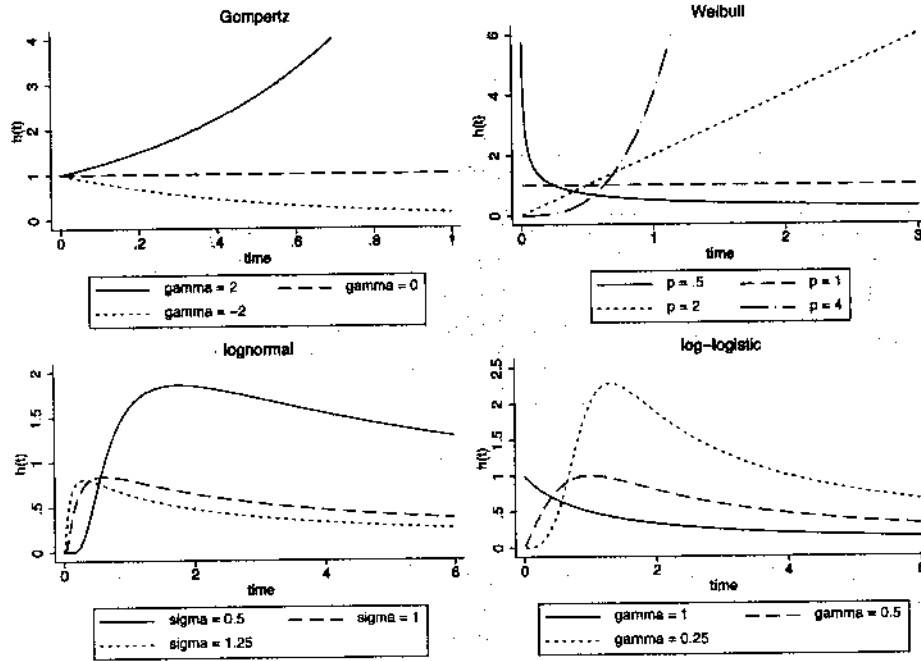


Figure 2.1: Examples of hazard functions obtained from various parametric survival models

In real datasets, we often do not observe subjects from the onset of risk. That is, rather than observing subjects from $t = 0$ until failure, we observe them from $t = t_0$ until failure with $t_0 > 0$. When the failure event is death or some other absorbing event after which continued observation is impossible or pointless, we will instead want to deal with the conditional variants of $S()$, $F()$, $H()$, $f()$, and $h()$. The important feature here is that those who failed (died) during the period 0 to t_0 will never be observed in our datasets. The conditional forms of the above functions are

$$\begin{aligned}
 h(t|T > t_0) &= h(t) \\
 H(t|T > t_0) &= H(t) - H(t_0) \\
 F(t|T > t_0) &= \frac{F(t) - F(t_0)}{S(t_0)} \\
 f(t|T > t_0) &= \frac{f(t)}{S(t_0)} \\
 S(t|T > t_0) &= \frac{S(t)}{S(t_0)}
 \end{aligned}$$

Conditioning on $T > t_0$ is common, and thus in what follows we suppress the notation

so that $S(t|t_0)$ is understood to mean $S(t|T > t_0)$, for instance. Note that $h(t)$ is unaffected by the conditioning; it is an instantaneous rate and so is not a function of the past.

The conditional functions may also be used to describe the second and subsequent failure times for events when failing more than once is possible. For example, the survivor function describing the probability of a second heart attack would naturally have to condition on the second heart attack taking place after the first, and so one could use $S(t|t_f)$, where t_f was the time of the first heart attack.

Some hazard functions for often-used distributions are depicted in Figure 2.1. Although determining a hazard from a density or distribution function is easy using (2.1), this is really turning the problem on its head. You want to think in terms of hazard functions.

2.2 The quantile function

In addition to $f()$, $F()$, $S()$, $h()$, and $H()$ —all different ways of summarizing the same information—a sixth way, $Q()$, is not often mentioned but is of use for those who wish to create artificial survival-time datasets. This is a book about analyzing data, not manufacturing artificial data, but sometimes the best way to understand a particular distribution is to look at the datasets it would imply.

The quantile function $Q(u)$ is defined (for continuous distributions) to be the inverse of the cumulative distribution function; that is,

$$Q(u) = F^{-1}(u)$$

so that $Q(u) = t$ only if $F(t) = u$. Among other things, the quantile function can be used to calculate percentiles of the time to failure distribution. The 40th percentile, for example, is given by $Q(0.40)$.

Our interest, however, is in using $Q()$ to produce an artificial dataset. What would a survival dataset look like that was, say, produced by a Weibull distribution with $p = 3$? Finding the quantile function associated with the Weibull lets us answer that question because the Probability Integral Transform states that if U is a uniformly distributed random variable of the unit interval, then $Q(U)$ is a random variable with cumulative distribution $F()$.

Stata has a uniform random-number generator, and so, with the addition of a little math by which we derive $Q()$ corresponding to $F()$, we can create artificial datasets. For the Weibull, from (2.3) we obtain $Q(u) = \{-\ln(1 - u)\}^{1/p}$, and so, if we want 5 random deviates from a Weibull distribution with shape parameter $p = 3$, we can type

```
. set obs 5
obs was 0, now 5
. set seed 12345
. gen t = (-ln(1-uniform()))^(1/3)
```

```
. list t
```

	t
1.	.7177559
2.	1.04948
3.	.5151563
4.	.9378536
5.	.7218155

We can also generate survival times that are conditional (or given) to be greater than some value t_0 ; that is, those that would represent times to failure for subjects who come into observation at some time $t_0 > 0$. The conditional distribution function $F(t|t_0) = \Pr(T \leq t|T > t_0) = 1 - S(t|t_0)$, and thus the conditional quantile function $Q(u|t_0) = F^{-1}(t|t_0)$. For the case of the Weibull,

$$Q(u|t_0) = \{t_0^p - \ln(1 - u)\}^{1/p}$$

and so we can use the following to generate 5 observations from a Weibull distribution with $p = 3$, given to be greater than, say, $t_0 = 2$. The generated observations would thus represent observed failure times for subjects whose times-to-failure follow a Weibull distribution, yet these subjects are only observed past time $t_0 = 2$. If failure should occur before this time, the subjects remain unobserved.

```
. gen t2 = (2^3 - ln(1-uniform()))^(1/3)
. list t2
```

	t2
1.	2.05692
2.	2.101795
3.	2.044083
4.	2.090031
5.	2.037246

Notice that these failure times do not extend much past time $t = 2$. The Weibull hazard function is an increasing function when $p > 0$, and one may gather from this small experiment that the hazard has already become quite large by time $t = 2$. Thus, those who are lucky enough to survive to the beginning of our study do not survive much longer. In other words, subjects begin accumulating risk at time zero even though they are not initially observed until time $t = 2$, and by that time they have already accumulated much risk and will continue to accumulate risk at an ever-increasing rate.

An alternate way to generate Weibull survival times conditional to be greater than 2 would be to generate several unconditional Weibull survival times and then drop those that are less than or equal to 2. Although a direct application of the rules of conditional probability, this approach is wasteful since you would be generating many survival times, only to end up dropping most of them. Also, the number of observations greater than 2

that you end up with is not guaranteed to be any certain number but is instead subject to chance.

As mentioned previously, the conditional survivor function not only describes independent individuals with late entry into a study but may also be used to describe repeated failures for a single individual, where each failure time is known to be greater than the previous one. We can simulate this process by conditioning each failure as being greater than the previous one. Using our Weibull with $p = 3$, we start at $t_0 = 0$ and generate the first failure time, and then generate four subsequent failures, each conditional on being greater than the previous one:

```
. gen u = uniform()
. gen t3 = (-ln(1-u))^(1/3) in 1
(4 missing values generated)
. replace t3 = ((t3[_n-1])^3 - ln(1-u))^(1/3) in 2/1
(4 real changes made)
. list t3
```

	t3
1.	1.075581
2.	1.417904
3.	1.43258
4.	1.459457
5.	1.696586

Now we can gather that the probability of survival past time $t = 2$ is really low, since if this hypothetical subject were allowed to fail repeatedly, he would have already failed at least 5 times by time $t = 2$. Of course, we would probably want to simulate various such individuals rather than only consider the single one depicted above, but already we can see that survival past $t = 2$ is highly unlikely. In fact, $S(2) = \exp(-2^3) \approx 0.03\%$.

Simulation is an excellent tool, not only for software developers who use it to test model estimation programs, but also to illustrate and to observe in practice how theoretical functions such as the hazard and the survivor functions realize themselves into observed failure times. Simulation serves to illustrate how these functions may be interpreted in more than one way, as seen above and as revisited later.

2.3 Interpreting the hazard and cumulative hazard

As previously mentioned, learning to think in terms of the hazard and the cumulative hazard functions, rather than the traditional density and cumulative density functions, has several advantages. Hazard functions give a more natural way to interpret the process that generates failures, and regression models for survival data are more easily grasped by observing how covariates affect the hazard.

2.3.1 Interpreting the cumulative hazard

The cumulative hazard function, $H(t)$, has much more to offer than merely an intermediate calculation to derive a survivor function from a hazard function. Hazards are rates, and in that they are not unlike the RPMs—revolutions per minute—of an automobile engine, because RPMs are also rates.

Cumulative hazards are the integral from zero to t of the hazard rates. Since an integral is really just a sum, a cumulative hazard is not unlike the total number of revolutions an automobile's engine makes over a given period of time. We could form the cumulative-revolution function by integrating RPM over time. If we let a car engine run at a constant 2,000 RPMs over a period of two minutes, then the cumulative RPMs at time two minutes would be 4,000, meaning the engine would have revolved 4,000 times over that period. Similarly, if a person faced a constant hazard rate of 2,000/minute (a big risk) for a period of 2 minutes, he would face a total hazard of 4,000. Going back to the car engine, if we raced the engine at 3,000 RPM for one minute and then let it idle at 1,000 for another, the total number of revolutions (cumulative RPMs) would still be 4,000. Going back to our fictional risk taker, if he faced a hazard of 3,000/minute for one minute and then a hazard of 1,000/minute for another, his total risk would still be 4,000.

Now let's stick with our fictional friend. Whatever the profile of risk, if over a 2-minute period the cumulative hazard is the same, then the probability of the event (presumably, death) occurring during that 2-minute period is the same.

Let's understand the units of this measurement of risk. In this, cumulative hazards are more easily understood than the hazard rates themselves. Remember that $S(t) = \exp\{-H(t)\}$, so our fictional friend has a probability of surviving the two-minute interval of $\exp(-4000)$, which is to say, our friend is going to die. One may similarly calculate the probability of survival given other values for the cumulative hazard.

Probabilities, however, are not the best way to think about cumulative hazards. An alternate interpretation of the cumulative hazard is that it records the number of times we would expect (in the mathematical sense) to observe failures over a given time period, if only the failure event were repeatable. Taking our fictional friend again, the cumulative hazard of 4,000 over the two-minute period means that we would expect him to die 4,000 times if, as in a video game, each time he died we could instantly resurrect him and let him continue on his risky path.

This is called the count-data interpretation of the cumulative hazard and learning to think this way has its advantages.

► Example

To see the count-data interpretation in action using Stata, let's consider an example using, as previously, the Weibull distribution with shape parameter $p = 3$, which has a cumulative hazard $H(t) = t^3$. For the time interval $(0, 4)$, since $H(4) = 64$, we can interpret this to mean that, if failure were a repeatable process, we would expect 64 failures over this time period.

This fact may be verified via simulation. We proceed, as we did in section 2.2, by generating times to failure in a repeated failure setting, where each failure time is conditional on being greater than the previous one. This time, however, we will repeat the process 1,000 times, and for each replication we observe the random quantity N , the number of failure times that are less than $t = 4$. That is, for each replication we count the number of failures that occur in the interval $(0, 4)$, and record this count. In Stata, this may be done via the `simulate` command; see [R] `simulate`.

```
. set seed 12345
. program genfail
1.     drop _all
2.     set obs 200
3.     gen u = uniform()
4.     gen t = (-ln(1-u))^(1/3) in 1
5.     replace t = ((t[_n-1])^3 - ln(1-u))^(1/3) in 2/1
6.     count if t<4
7. end
. simulate "genfail" nfail=r(N), reps(1000)
command:    genfail
statistic:  nfail      = r(N)
. summarize
```

Variable	Obs	Mean	Std. Dev.	Min	Max
nfail	1000	63.788	7.792932	38	95

This simulation thus helps to verify that $E(N) = H(4) = 64$, and in fact, if we replicated this experiment infinitely often, the mean of our simulated N values would equal 64 exactly.

<

□ Technical Note

In the above simulation, the line `set obs 200` sets the size of the data to 200 observations for each replication of the counting failures experiment. Theoretically, any number of failures is possible in the interval $(0, 4)$, yet the probability of observing more than 200 failures is small enough as to make this an acceptable limit.

□

There is no contradiction between the probability-of-survival interpretation and the repeated-failure count interpretation, but be careful. Consider a cumulative hazard equal to one. One interpretation is that we expect to observe one failure over the interval. The other interpretation is that the probability for which we observe no failures over the interval is $\exp(-1) = 0.368$, and thus the probability for which we observe the failure event is $1 - 0.368 = 0.632$. Are you surprised that the chances are not 50/50?

More correctly, we should have said that the chance that we will observe one of more failures per subject is 0.632, except that if this failure event is absorbing (i.e., death-like in that it can occur only once), we will never observe the second and subsequent failures because the first failure will prohibit that. Therefore, 0.632 is also the probability that we will observe a single failure. If, on the other hand, this failure event is not absorbing,

then 0.368 is the probability that we will observe no failures, and 0.632 is the probability that we will observe one or more failures and moreover, we can decompose that into the probability of one failure, the probability of two failures, etc., and in doing so we can compose a probability mass function for a random variable that has an expected value of one.

2.3.2 Interpreting the hazard rate

Hazard rates are rates, which is to say, they have units $1/t$. You can interpret hazard rates just as you interpret cumulative hazards if you multiply them by t . In that case, you are saying, "The hazard rate is such that, were that rate to continue for 1 time unit, we would expect that . . ."

For instance, if the hazard rate is 2/day, then the hazard rate is such that, were that rate to continue for an entire day, you would expect 2 failures, or, if you prefer, the chances of observing a failure would be $1 - \exp(-2) = 0.8647$.

There is a subtle distinction here. If the cumulative hazard over a period is 2—if the integrated instantaneous hazard rate over the period is 2—then over that period you would expect 2 failures, regardless of the time profile of the hazard rates themselves. During that period, the hazard rate might be constant, increasing, decreasing, or any combination of these. If, on the other hand, the hazard rate is 2/day at some instant, then failures are happening at the rate of 2/day at that instant. However, you would only expect 2 failures over a period of a day if that hazard rate stayed constant over that period or varied in such a way as to integrate to 2 over that day.

Hazard rates—were they to stay constant—have a third interpretation. Hazard rates have units $1/t$; hence, the reciprocal of the hazard has units t , and represents how long you would expect to have to wait for a failure if the hazard rate stayed at that level. If the hazard rate is 2/day, then were the hazard rate to remain at that level, we would expect to wait half a day for a failure. In fact, a constant hazard rate is what characterizes the classic Poisson counting process, where it can be shown that if the expected time between failure is half a day with constant hazard, then the number of failures that occur in any given day (if failures were repeatable) is a Poisson random variable with expected value (and variance) equal to 2.

2.4 Means and medians

Given a random failure time T with probability density function $f(t)$, the mean time to failure, μ_T , is defined to be

$$\mu_T = \int_0^{\infty} t f(t) dt$$

Equivalently, one can show that $\mu_T = \int_0^{\infty} S(t) dt$, which often simplifies the calculation.

The median failure time, $\tilde{\mu}_T$, is defined as the 50th percentile of the failure time distribution; that is, that value such that half of all failure times are less than $\tilde{\mu}_T$ and half are greater than $\tilde{\mu}_T$. Thus, for continuous distributions, $\tilde{\mu}_T$ is the value such that $F(\tilde{\mu}_T) = S(\tilde{\mu}_T) = 0.5$ and can be obtained directly from the quantile function

$$\tilde{\mu}_T = Q(0.5)$$

since $Q(u)$ may be used to obtain any percentile, the 50th just being a special case. For the Weibull distribution with shape parameter p ,

$$\begin{aligned}\mu_T = \int_0^\infty t f(t) dt &= \int_0^\infty p t^p \exp(-t^p) dt \\ &= \Gamma(1 + 1/p)\end{aligned}$$

where $\Gamma()$ is the gamma function, and the median $\tilde{\mu}_T = Q(0.5) = \{\ln(2)\}^{1/p}$.

The mean and median formulas above are often used in parametric regression to obtain predictions of failure times for survivor functions that are specified given the values of certain predictors and estimated regression coefficients, through direct application of the above formulas. In semiparametric or nonparametric schemes, predicting the mean and median is possible through some adaptation of these formulas, taking into account that for these models, the estimated survivor function is not continuous but rather a step function where the steps occur at each observed failure time. In any case, predictions of the mean and of the median failure time are useful in that they give a sense of the typical time to failure for a particular distribution. Remember that survival-time distributions can have long, right tails, and in these cases, some care is required when interpreting the mean failure time. For instance, with a constant hazard of 0.01 per day, the mean time to failure is 100 days, and yet the median is only 69 days. The difference is caused by those few who survive much longer than 100 days.

Survival regression models are often fit in the $\ln(\text{time})$ metric; that is, one forms the model by hypothesizing a distribution for the natural logarithm of time to failure. Defining the random variable $Y = \ln(T)$ with probability distribution $f_Y(y)$, one can obtain predictions of the mean and median of Y using the above techniques, and then transform the results back into the time metric by exponentiating the obtained values. In general, the exponentiated mean of Y is not equal to μ_T and thus would represent yet another flavor for a prediction of the typical survival time. In the case of medians, however, the transformation is invariant. Taking the median of the distribution of Y and exponentiating will produce $\tilde{\mu}_T$. When obtaining a "predicted" time to failure from statistical software, be sure you know what you are getting, or be prepared to specify what you want and see if it is available.

3 Hazard models

We began our discussion in Chapter 1 by writing models of the form $t_j = \beta_0 + \beta_1 x_j + \epsilon_j$ or $\ln(t_j) = \beta_0 + \beta_1 x_j + \epsilon_j$, where ϵ_j is suitably distributed. These days, people seldom write models in this form. Instead, they write

$$h_j(t) = g(t, \beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x)$$

which is to say, the hazard—the intensity with which the event occurs—for person j is some function $g()$ of $\beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x$, where we now allow for the presence of multiple predictors via the row vector \mathbf{x}_j , in which case $\boldsymbol{\beta}_x$ is a column vector of regression coefficients. For instance, we might write

$$h_j(t) = (\beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x)t$$

or something more complicated.

Regardless, this change from $t_j = \beta_0 + \beta_1 x_j + \epsilon_j$ to $h_j(t) = g(t, \beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x)$ is just a change in notation and has no substantive implications. Remember, there is a one-to-one mapping from distributions to hazard functions. The distributional assumption we make for ϵ_j is now wrapped up in the hazard function that we choose. If we choose $g()$ appropriately, the likelihood function we obtain is exactly the same, and therefore the resulting estimates are the same. We can even write linear regression using this notation; we just need to work out what the hazard function is for linear regression. Remember that, in general, the hazard function $h(t) = f(t)/S(t)$.

So, why should we write our models in this form? The answer is that this notation can also incorporate the semiparametric models that we discussed. The idea here is to write the model as

$$h_j(t) = \text{somefunction}(h_0(t), \beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x)$$

where $h_0(t)$ is called the *baseline hazard*. That is, the hazard subject j faces is *somefunction()* of the hazard everyone faces, modified by \mathbf{x}_j . A particularly popular way to parameterize these models is

$$h_j(t) = h_0(t) \exp(\beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x)$$

These are called proportional hazards models. It is proportional in that the hazard subject j faces is multiplicatively proportional to the baseline hazard, and the function $\exp()$ was chosen simply to avoid the problem of $h_j()$ ever turning negative. Actually,

even if we choose some function different from $\exp()$, it is still called the proportional hazards model.

There is nothing magical about proportional hazards models and, for some problems, the proportional hazards assumption may be inappropriate. One could just as easily write an additive-hazards model; for instance,

$$h_j(t) = h_0(t) + \exp(\beta_0 + \mathbf{x}_j\beta_x)$$

3.1 Parametric models

Any parametric survival model can be written in the hazard notation (although not necessarily the proportional hazards notation), and doing that is just an exercise in translating from one notation to another. All parametric models have a corresponding hazard function.

It turns out that a number of popular survival parametric models, which are naturally written in the form $\ln(t_j) = \beta_0 + \mathbf{x}_j\beta_x + \epsilon_j$, have corresponding $h()$ functions that naturally decompose into the $h_0(t) \exp(\beta_0 + \mathbf{x}_j\beta_x)$ notation. The exponential and Weibull models are two examples. When that occurs, the model is said to have both a (log)-time metric and a proportional hazards metric formulation.

Familiarity, plus the fact that some $\ln(t)$ models have a proportional hazards interpretation, has caused many researchers to focus on this model exclusively:

$$h_j(t) = h_0(t) \exp(\beta_0 + \mathbf{x}_j\beta_x)$$

The proportional hazards parametric model is just a matter of picking a functional form for $h_0(t)$. The $\exp()$ part is just how researchers have chosen to parameterize in all proportional hazards models the shift caused by subjects having different covariate (\mathbf{x}) values.

We can literally pick any positive function for $h_0(t)$ that we wish, although, if we pick a strange one, we will probably have to write our own maximum likelihood estimator. Certain functions for $h_0(t)$ are popular, and those are preprogrammed in Stata; for instance, if we choose

$$h_0(t) = c$$

for some constant c , then we fit what is called the exponential regression model. It is called exponential regression because, were we to translate the model back to the time metric, we would find that t , the time to failure, follows the exponential waiting-time distribution. That is, if we worked out t 's distribution $f(t)$ —which we could easily do because it is always true that $f(t) = h(t) \exp\{-H(t)\}$ —we would discover that $f(t)$ is the exponential density.

In any case, parameter estimates for these models are obtained by maximum likelihood. The likelihood of the data (with no censored observations, which we will discuss later) is

$$L(\beta|t_1, t_2, \dots) = f(t_1|\beta, \mathbf{x}_1)f(t_2|\beta, \mathbf{x}_2) \dots$$

for $\beta = (\beta_0, \beta_x)$. In the loose sense of the term, $L()$ can be taken to be the “probability” of observing a failure time for the first subject, t_1 , given the value of \mathbf{x}_1 , times the probability of observing a failure time t_2 for the second subject given \mathbf{x}_2 , etc. The basic idea behind maximum likelihood estimation is that, given a set of observations (t_1, t_2, \dots, t_n) , the best estimate of β is the one that maximizes the probability, or likelihood, of observing those particular data. Maximum likelihood estimates also have very nice statistical properties that make inference and testing analogous to what you would see in simple OLS regression; see Casella and Berger (2002) or some other graduate text on mathematical statistics for a thorough discussion of likelihood theory.

And so, despite our modern way of thinking in terms of hazard functions, the likelihood above is still in terms of the density function of t . Mathematically, we could have just as well have thought about this in terms of densities from the outset. We can write this more modernly as

$$L(\beta|t_1, t_2, \dots) = S(t_1|\beta, \mathbf{x}_1)h(t_1|\beta, \mathbf{x}_1)S(t_2|\beta, \mathbf{x}_2)h(t_2|\beta, \mathbf{x}_2) \dots$$

because $f(t) = S(t)h(t)$. However we write it, we maximize this likelihood to estimate β .

3.2 Semiparametric models

Semiparametric models, as mentioned in Section 1.2, amount to nothing more than combining individual binary-outcome analyses at each of the failure times. We want to show you exactly how that works and how it fits into the proportional hazards way of thinking.

We will start with the same proportional hazards model that we started with in the parametric case:

$$h(t) = h_0(t) \exp(\beta_0 + \mathbf{x}_j \beta_x)$$

This time, however, rather than specifying a function for $h_0(t)$, we will leave it unspecified, and it will turn out, it will cancel from our calculations when we perform the binary-outcome analyses at the individual failure times. This is called the Cox proportional hazards model, discussed in much more detail later in the text.

The likelihood function is calculated over the separate binary-outcome analyses that we perform:

$$L(\beta|\text{data}) = L(\text{analysis 1})L(\text{analysis 2}) \dots \quad (3.1)$$

We perform one analysis per failure time, and each analysis returns the probability of failing for those who did in fact fail at that time. For instance, in our small dataset,

subject	time	x
1	1	3
2	5	2
3	9	4
4	20	9
5	22	10

we would perform five analyses:

1. *Analysis at time 1:* The probability that subject 1 is the one that fails in a dataset containing all five subjects.
2. *Analysis at time 5:* The probability that subject 2 is the one that fails in a dataset containing subject 2 and subjects 3, 4, and 5.
3. *Analysis at time 9:* The probability that subject 3 is the one that fails in a dataset containing subject 3 and subjects 4 and 5.
4. *Analysis at time 20:* The probability that subject 4 is the one that fails in a dataset containing subject 4 and subject 5.
5. *Analysis at time 22:* The probability that subject 5 is the one that fails in a dataset containing subject 5.

Well, at least the last analysis is easy; the probability is one. The next to the last analysis is the next easiest.

At time 20 in our data, only subjects 4 and 5 survived up to that point, and subject 4 failed at that point. Per our model, the hazard of failure at time t is

$$h_j(t) = h_0(t) \exp(\beta_0 + \mathbf{x}_j \beta_x)$$

and so the hazard at time 20, the only time we care about for this analysis, is

$$h_j(20) = h_0(20) \exp(\beta_0 + \mathbf{x}_j \beta_x)$$

The only subjects we care about are 4 and 5, who have scalar \mathbf{x} values of 9 and 10, respectively:

$$h(\text{subject 4 at time 20}) = h_4(20) = h_0(20) \exp(\beta_0 + 9\beta_x)$$

$$h(\text{subject 5 at time 20}) = h_5(20) = h_0(20) \exp(\beta_0 + 10\beta_x)$$

Given that we observe one failure at this time, the probability that the failure is subject 4 is given by

$$\begin{aligned}
\Pr(4 \text{ fails} | a \text{ failure}) &= \frac{h_4(20)}{h_4(20) + h_5(20)} \\
&= \frac{h_0(20) \exp(\beta_0 + 9\beta_x)}{h_0(20) \exp(\beta_0 + 9\beta_x) + h_0(20) \exp(\beta_0 + 10\beta_x)} \\
&= \frac{\exp(9\beta_x)}{\exp(9\beta_x) + \exp(10\beta_x)}
\end{aligned} \tag{3.2}$$

So, tell us a value of β_x , and we will tell you a probability. If $\beta_x = -1$, then the probability is 0.731059, and we can tell you that, even though we make no assumptions about the shape of the baseline hazard $h_0(t)$.

Where does the value of β_x come from? We find the value of β_x that maximizes the overall likelihood (3.1). We do analyses just as the one shown for each of the failure times and then find the value of β_x that maximizes $L(\beta_x | \text{data})$. Note that the intercept, β_0 , drops out of the above calculations. This is a property of the semiparametric proportional hazards (Cox) model, which we cover in more detail in Chapter 9. For now, we merely take satisfaction in the fact that we have one less parameter to estimate.

The story we told you earlier about performing separate analyses was quite literally correct, although the jargon usually used to describe this is different. Rather than calling the ingredients separate *analyses*, they are called the “risk groups based on ordered survival times”, and we obtain our estimates by “pooling over the risk groups based on ordered survival times”.

□ Technical Note

Arriving at (3.2) required a bit of hand waving, and so here we fill in the details for the interested reader. We are interested in the probability that subject 4 fails at time $t = 20$, given that (a) subjects 4 and 5 survive to just before $t = 20$ and (b) either subject 4 or subject 5 will fail at $t = 20$. Let Δ be some arbitrarily small number, and let T_4 and T_5 denote the failure times of subjects 4 and 5, respectively. Then the probability that only subject 4 fails within Δ after time 20, given that both subjects survive to time 20 and that one of the subjects fails within Δ after time 20, is

$$\Pr\{T_4 < 20 + \Delta, T_5 > 20 + \Delta | T_4 > 20, T_5 > 20, \text{one failure in } (20, 20 + \Delta)\}$$

which equals

$$\frac{\Pr(20 < T_4 < 20 + \Delta, T_5 > 20 + \Delta | T_4 > 20, T_5 > 20)}{\Pr\{\text{one failure in } (20, 20 + \Delta) | T_4 > 20, T_5 > 20\}} \tag{3.3}$$

Define \mathcal{E}_i to be the event $20 < T_i < 20 + \Delta$, \mathcal{F}_i to be the event $T_i > 20 + \Delta$, and \mathcal{G}_i to be the event $T_i > 20$. Then, (3.3) becomes

$$\begin{aligned}
 & \frac{\Pr(\mathcal{E}_4, \mathcal{F}_5 | \mathcal{G}_4, \mathcal{G}_5)}{\Pr(\mathcal{E}_4, \mathcal{F}_5 | \mathcal{G}_4, \mathcal{G}_5) + \Pr(\mathcal{E}_5, \mathcal{F}_4 | \mathcal{G}_4, \mathcal{G}_5)} \\
 &= \frac{\Pr(\mathcal{E}_4 | \mathcal{G}_4) \Pr(\mathcal{F}_5 | \mathcal{G}_5)}{\Pr(\mathcal{E}_4 | \mathcal{G}_4) \Pr(\mathcal{F}_5 | \mathcal{G}_5) + \Pr(\mathcal{E}_5 | \mathcal{G}_5) \Pr(\mathcal{F}_4 | \mathcal{G}_4)} \\
 &= \frac{\{\Pr(\mathcal{E}_4 | \mathcal{G}_4) / \Delta\} \Pr(\mathcal{F}_5 | \mathcal{G}_5)}{\{\Pr(\mathcal{E}_4 | \mathcal{G}_4) / \Delta\} \Pr(\mathcal{F}_5 | \mathcal{G}_5) + \{\Pr(\mathcal{E}_5 | \mathcal{G}_5) / \Delta\} \Pr(\mathcal{F}_4 | \mathcal{G}_4)} \quad (3.4)
 \end{aligned}$$

Since $\lim_{\Delta \rightarrow 0} \{\Pr(\mathcal{E}_i | \mathcal{G}_i) / \Delta\} = h_i(20)$ by definition and $\lim_{\Delta \rightarrow 0} \Pr(\mathcal{F}_i | \mathcal{G}_i) = 1$, taking the limit as Δ goes to zero of (3.4) yields (3.2). \square

3.3 Analysis time (time at risk)

We have considered hazard models of the form

$$h_j(t) = g(t, \beta_0 + \mathbf{x}_j \beta_x)$$

and, in the particular case of the proportional hazards model, we write

$$h_j(t) = h_0(t) \exp(\beta_0 + \mathbf{x}_j \beta_x) \quad (3.5)$$

Either way, we are assigning explanatory power to time. That is absurd. If we are analyzing a health outcome, it may be that the accumulation of toxins in someone's system makes the subject more likely to die—that is correlated with time—but time is not the cause. Or, if we are studying employment, it may be that the more information an unemployed person has about the job market—information he collects on the days he looks for employment—the more likely he is finally to accept a job offer. It then appears as if duration of unemployment is a function of time, but time is not the cause. To assume that the ticking of the clock somehow, by itself, changes the hazard is absurd unless we are physicists analyzing the true nature of time.

If we fully understand the process and had sufficient data, there would be no role for time in our model, and as a matter of fact, our current models can deal with that. In the proportional hazards case, we would simply define $h_0(t)$ as a constant (call it c) so that

$$h_j(t) = c \exp(\beta_0 + \mathbf{x}_j \beta_x) = c^* \exp(\mathbf{x}_j \beta_x)$$

for some other constant c^* . This model is known as the exponential model, and it is the only proportional hazards model that leaves nothing unsaid.

► **Example**

Consider the case where our baseline hazard function is the Weibull hazard with shape parameter $p = 3$; that is, $h_0(t) = 3t^2$. The proportional hazards model (3.5) then becomes

$$\begin{aligned} h_j(t) &= 3t^2 \exp(\beta_0 + \mathbf{x}_j \beta_x) \\ &= \exp\{\beta_0 + \ln(3) + \mathbf{x}_j \beta_x + 2 \ln(t)\} \end{aligned}$$

If we define the time-varying covariate $z(t) = \ln(t)$, then this model may be reformulated as an exponential (constant baseline hazard) model with covariates \mathbf{x} and $z(t)$, resulting in equivalent estimates of β_x and an intercept term that is shifted by $\ln(3)$.

Including a time-varying covariate among the explanatory variables is more easily said than done—in Stata, you must split observations into pieces, something we will discuss in Section 13.1.1. In any case, it is not our intention to recommend this as a way to fit Weibull models. The point is that Weibull models assign a role to time, a fact demonstrated by reformulating the model in terms of the exponential model—the model that leaves nothing unsaid—and that in the reformulation, time itself appears as an explanatory variable.

◄

In the fully defined hazard model, we would just write $h_j(t, \mathbf{x}_j) = h_j(\mathbf{x}_j)$. The shift in the hazard would be fully explained by our \mathbf{x} variables.

With the exception of the exponential models, every other model assigns a role to time, and that includes the semiparametric models. In semiparametric models, we get around ever having to define $h_0(t)$ because the individual analyses are performed on subjects at the same value of t , and therefore the same value of $h_0(t)$. The important part is that we compare subjects when their values of t are the same, meaning that subjects with equal t values (and \mathbf{x} values) must share the same risk of the event.

When we assign a role to time, we are doing that to proxy other effects that we do not fully understand, cannot measure, are too expensive to measure, or are unknown. We need to consider how we are going to measure t so that it properly fulfills its role as a proxy. There are two aspects to the definition of t :

1. Ensuring that whenever two subjects have equal t values, the risk they face would be the same if they also shared the same \mathbf{x} values.
2. Deciding which particular value of t should be labeled $t = 0$, denoting the onset of risk.

Property (1) is important to all models. Property (2) only matters when we are fitting parametric models.

So, when do two subjects have the same risk? You have to think carefully about your problem. Here are three situations:

1. Say that you are analyzing the association between lung cancer and smoking. It would be unreasonable to define t as age. You would be stating that two persons with equal characteristics would have the same risk of cancer when their ages are equal, even if one had been smoking for 20 years and the other for one month.
2. Continuing with the analysis of the association between lung cancer and smoking, it might be reasonable to assume that two smokers face the same risk when they have been smoking for the same amount of time. That would be reasonable if all smokers smoked with the same frequency. If there are large variations, and we had some "smokers" who smoked one cigarette per day and others who smoked 60 per day, time since onset of smoking might not be a good proxy for what we want to measure (assuming we did not account for the discrepancy with our x variables). If all our subjects were smokers, we might think about measuring time as cigarette-days, so that a person who smokes 1 cigarette per day for 20 days is at time $t = 20$ just as a person at day 1 who smokes 20 cigarettes per day.
3. You are studying the time to death for a certain type of cancer patient. When do two identical patients face the same risk? At the same time since the onset of cancer, you decide. Fine, but when is the onset of the cancer? At detection? Are two patients really the same when, at $t = 0$, one reports to the clinic with metastatic cancer invading other organs and the other with a small, barely detectable tumor? Or is the onset of risk when the first cancer cell differentiates, and, in that case, how would you know when that was?

Defining t is worth the thought, and it is usually easiest to think in terms of the onset of risk—to find the time at which risk began and before which the event could not happen for the reason under analysis. If that onset of risk is well defined in the data, then defining $t = 0$ is easy enough and you will probably let t increment in lock step with calendar time, although you may want to think about time-intensity measures such as cigarette-days.

If there is no well-defined onset of risk, think about when two subjects with the same x values would face the same risk and define t so that they have the same value of t at those times. That will not uniquely label one of the times $t = 0$, but that may not matter.

Pretend that, for some problem, you have defined t in such a way that identical subjects facing the same risk do have equal t values. We could define $t' = t - 5$, and this new time would work just as well in terms of matching subjects. So, is the onset of risk with $t = 0$ or $t' = 0$? That is a substantive question to which you will have to turn to your science for an answer.

If you engage in semiparametric modeling, however, how you answer that question does not matter because semiparametric results are determined only by the matching and ordering of failure times. Time is used only to order the data, and no special significance is attached to $t = 0$.

In many parametric survival models, however, special significance is attached to $t = 0$. For these hazard functions, you can fit a parametric model, write down the results, add 5 to every t value, re-estimate the parameters, and get different results. The results will not be merely apparently different—somehow undoing the +5 addition you just made. The results will be substantively different in that predictions made using the model will differ.

Although in most parametric models the value chosen for $t = 0$ matters, the units in which t is measured (minutes, hours, days, years, etc.) do not matter. More correctly, for no popularly used hazard function $h_0(t)$ do the units of t matter. Since you can choose any nonnegative function for $h_0(t)$, you could choose one in which the units do matter, but that would be a poor choice.

From now on, we are going to refer to t as *analysis time* to emphasize that t is as defined above and is not just some time or date variable you happen to have laying around in your data:

$$t = \text{somefunction}(\text{various time or date variables in your data})$$

For instance,

$$t = \frac{\text{date_of_failure} - \text{date_of_birth}}{365.25}$$

defines t as years of age and birth as corresponding to the onset of risk.

$$t = \text{date_of_failure} - \text{date_of_diagnosis}$$

defines t as days since diagnosis and diagnosis as corresponding to the onset of risk.

$$t = \text{hour_of_failure} - \text{hour_of_start}$$

defines t as hours from start and start as corresponding to the onset of risk.

$$t = \text{cigs_per_day} \times (\text{date_of_failure} - \text{date_started_smoking})$$

defines t as cigarette-days since the start of smoking and defines the start of smoking as corresponding to the onset of risk. Just remember, this choice of analysis time is not arbitrary.

Analysis time t does not have to be time. For instance, we have a machine that produces bolts of cloth, and we then examine those bolts for defects. It might be reasonable to measure “time” as the linear distance from the beginning of the bolt so that, if we were to find a defect at 4 meters into the bolt, the bolt “failed” at $t = 4$ meters.



4 Censoring and truncation

4.1 Censoring

In real data-analysis situations, we often do not know when failures occurred, at least not for every observation in the dataset. Rather than knowing that failures occurred at analysis times 1, 5, 9, 20, and 22, we might know that failures occurred at 1, 5, 9, and 20, and that, in the last case, failure had not yet occurred by analysis time 22, when we stopped our experiment:

time	failed	x
1	1	3
5	1	2
9	1	4
20	1	9
22	0	10

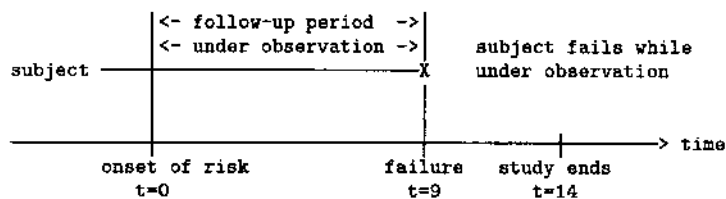
This is called *censoring*, or more precisely, right censoring. This happens so commonly that some researchers will write their datasets

time	censored	x
1	0	3
5	0	2
9	0	4
20	0	9
22	1	10

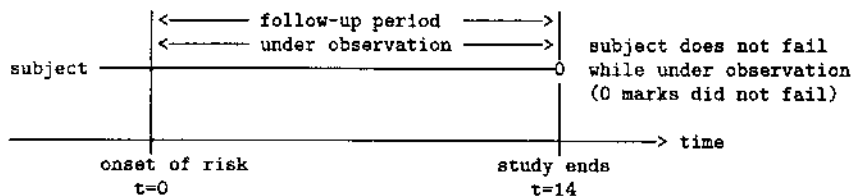
but Stata has a marked preference for the first form. `failed==1` means the observation was observed to fail at that time, and `failed==0` means the observation was not observed to fail, the implication being that the observation was right censored.

Censoring is defined as when the failure event occurs and the subject is not under observation. Think of censoring as being caused by a censor standing between you and reality, preventing you from seeing the exact time of the event that you know occurs.

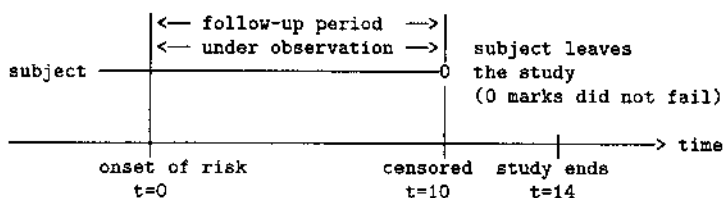
Before we get too deeply into this, let us define some popularly used jargon. We are following subjects over time, and that data collection effort is typically called a *study*. During the study period, subjects are *enrolled* and data are collected for a while, called the subject's *follow-up* period, or if you prefer, the period during which the subject was under observation. Data collection stops on a subject because the subject fails,



the study ends,



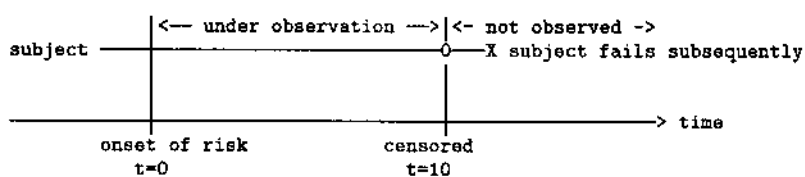
or the subject leaves the study for other reasons:



In the above diagrams, we drew the beginning of "under observation" to coincide with the onset of risk; however, the subject may come at risk for the failure event before enrollment, at the instant of enrollment, or afterwards. From a statistical perspective, we are interested in the period when the subject is at risk.

In the discussions that follow, we are going to assume that failure is a culminating or absorbing event such as death. The event can only occur once, and once it does occur, the subject can no longer be observed. Censoring can also be extended to repeatable failures.

4.1.1 Right censoring



When most investigators say censoring, they mean right censoring. In this type of censoring, the subject participates in the study for a time and, thereafter, is no longer observed. This can occur, for example,

1. when one runs a study for a pre-specified length of time, and by the end of that time, the failure event has not yet occurred for some subjects (this is common in studies with limited resources or with time constraints),
2. when a subject in the study withdraws prematurely (in trials of a new drug therapy, a patient might experience intolerable side effects, and thus cease participation), or
3. when a subject is lost to follow-up, meaning that he or she disappears for unknown reasons (in a longitudinal study, for example, a subject moves (say, because of marriage) and cannot be located).

The analytic tools we use assume that, if censoring occurs, it occurs randomly and is unrelated to the reason for failure. It will not do at all if, just prior to failure, subjects are highly likely to disappear from our view.

In parametric models, right censoring is dealt with quite easily. First, start by thinking of a dataset with no censoring in which all failure times are measured exactly. The likelihood function is (for a sample of size n , failure times t_1, \dots, t_n)

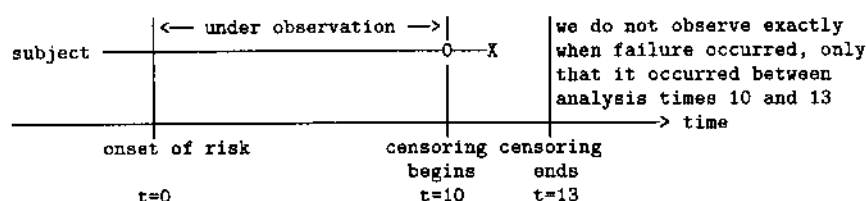
$$L\{\beta|(t_1, \mathbf{x}_1), \dots, (t_n, \mathbf{x}_n)\} = \prod_{i=1}^n S(t_i|\mathbf{x}_i, \beta)h(t_i|\mathbf{x}_i, \beta) \quad (4.1)$$

since the probability density function of t_i , $f(t_i) = S(t_i)h(t_i)$.

Equation (4.1) actually becomes simpler in the presence of censoring because, for those failure times that are censored, the density of t_i , $S(t_i|\mathbf{x}_i, \beta)h(t_i|\mathbf{x}_i, \beta)$, is replaced by only the survivor function $S(t_i|\mathbf{x}_i, \beta)$. In other words, all that is known about a censored observation is that failure occurs *after* time t_i , yet exactly when remains unknown (recall that by definition $S(t) = \Pr(T > t)$). Our software has to be up to making this substitution, but that is all.

Right censoring is also easily dealt with in semiparametric models. Think of the individual binary-outcome analyses that we perform. If subject i is censored at t_i , then that subject enters all the individual failure-time studies up to and including t_i (the subject did not fail at that time) and after that is simply ignored. The only difference between a subject that fails and one that does not is that the latter never appears in an individual study while being marked as having failed. Again, our software has to know not to perform certain individual analyses, but that is all.

4.1.2 Interval censoring



With interval censoring, rather than knowing the exact time of failure or that failure occurred past a certain point, all we know is that the failure event occurred between two known time points—perhaps a short interval or a long one.

Interval censoring can result when subjects are evaluated or inspected at fixed time points throughout their follow-up period. In a clinical study, patients may be required to visit the clinic once per month over a period of several years. Assume that a patient visits a clinic at month 6 and then at month 7. At the 6-month evaluation, the patient was negative for the event of interest, and at month 7, she was positive. That is, the failure event occurred at some point between the 6th and 7th evaluations. Because we do not know exactly when the failure event occurred, this observation is interval censored.

Stata does not directly handle this kind of censoring, but a strong argument can be made that it should. This kind of censoring is easy to handle in parametric and difficult in semiparametric models.

Let's start with parametric models. For interval-censored observations, we know that subject i failed between times t_{0i} and t_{1i} . The probability that the failure occurred sometime before t_{1i} is $1 - S(t_{1i})$, and the probability that a failure occurred before t_{0i} is $1 - S(t_{0i})$. Thus,

$$\Pr(\text{failure between } t_{0i} \text{ and } t_{1i} | \mathbf{x}_i, \beta) = S(t_{0i} | \mathbf{x}_i, \beta) - S(t_{1i} | \mathbf{x}_i, \beta)$$

and this would be subject i 's contribution to the likelihood given by (4.1).

If we have a dataset in which the first subject failed at t_1 , the second subject was right censored at t_2 , and the third subject was only known to fail between t_{03} and t_{13} , then the likelihood would be

$$L\{\beta | (t_1, \mathbf{x}_1), \dots, (t_n, \mathbf{x}_n)\} = S(t_1 | \mathbf{x}_1, \beta) h(t_1 | \mathbf{x}_1, \beta) S(t_2 | \mathbf{x}_2, \beta) \times \{S(t_{03} | \mathbf{x}_3, \beta) - S(t_{13} | \mathbf{x}_3, \beta)\} \times \dots$$

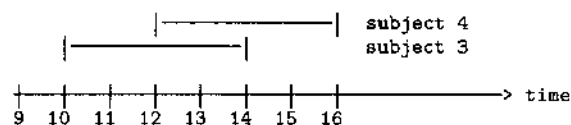
In semiparametric analysis, the problem is more difficult. Remember, we approach semiparametric analyses by performing separate binary-outcome analyses for each of the failure times and then combining those studies into one. Interval censoring is only a problem if it brings into dispute the ordering of the observations.

Pretend that we are uncertain about the failure time of subject 3, knowing only that he failed in the interval (10,14). If no subject has an observed or censored failure time in that interval, then really our uncertainty about subject 3 makes no difference. Whether death occurs at time 10, 14, or any time in between would still place subject 3 in the same temporal ordering, and so that binary analysis would be unaffected. All the other binary analyses would, of course, be unaffected, so combining the analyses must yield the same overall result.

This fact is a real lifesaver. Very few of us have datasets that record the exact time of failure—the failure occurred at 12:51:02.395852 on 12Feb2000. Say that you have data that records the time of failure in days. You can still order the events—and therefore the analyses—except for the failures that happen on the same day. If subjects 3 and 4 both failed on day 20, you do not know whether they failed in the order 3 and then 4 or 4 and then 3, and so you do not know in which order to perform your individual binary-outcome analyses. That, however, turns out not to be a difficult problem to solve, and as we discuss in Chapter 9, the software is capable of handling this.

You are equally saved if you record your data in terms of, say, integer number of months. In this case, you do not have the detail of information that you would have were the times recorded in, say, days, but that only means that you will get less-efficient estimates should more precise time measurements prove effective in breaking the (apparent) tied failure times caused by less precise measurements of time to failure. The bottom line is that you can still perform the analysis using standard tools.

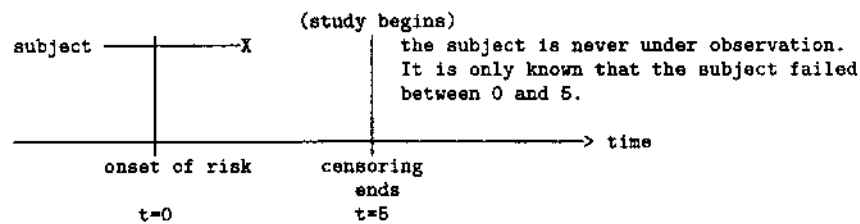
The difficulty arises when exact times are uncertain and only partially overlap. For example, subject 3 is known to fail within the interval (10,14), and subject 4 is known to fail in (12,16):



The best way to handle this is still a subject of research, but one solution would be to assume that the distribution of failure time is uniform (flat) within each interval and to break the combined interval into three sections. Between (10,12), subject 3 fails first; between (12,14), the two subjects are "tied", and we can handle that appropriately in Stata; and between (14,16), subject 3 fails first again. Assuming uniformly distributed survival times within each interval, the probability that we observe a tie (both subjects fail in (12,14)) is $1/4$, and thus we can weight the analysis that treats these as tied values by $1/4$, and the analysis that has subject 3 failing first by $3/4$.

Of course, this system of weighting relied on the failure time having a uniform distribution within the interval—meaning the hazard must be falling in the interval. As such, these weights may prove to be invalid for certain types of data, in which case we would need to adjust our weighting scheme.

4.1.3 Left censoring



Mathematically, left censoring is no different from interval censoring. The event occurred at some time when the subject was not under observation, in this case prior to the subject being under observation, but that makes no mathematical difference. The event happened in an interval of time and the solution is the same.

How left censoring arises in reality is probably different from interval censoring. Left censoring means the failure event occurs prior to the subject entering the study. For example, in a study of time to employment, an individual who is employed when first interviewed is considered as being left censored because the transition from unemployment to employment (the failure event) happened prior to the beginning of the follow-up period.

4.2 Truncation

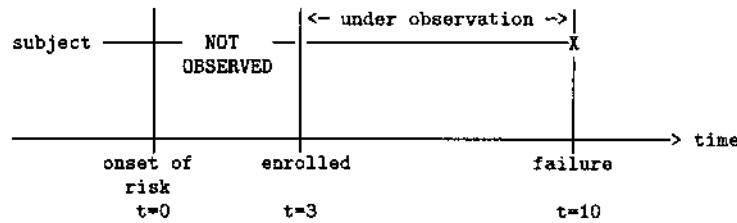
Truncation is often confused with censoring because it also gives rise to incomplete observations over time. Truncation, in most statistical applications, refers to complete ignorance about the event of interest and about the covariates over a portion of the distribution.

In survival-data applications, it is difficult to be so ignorant when the event is absorbing, such as death. The very fact that we observe you today means that you did not die yesterday. Therefore, for such absorbing events, truncation is defined as a period over which the subject was not observed but is, *a posteriori*, known not to have failed. The statistical difficulty that truncation causes is that, had the subject failed, he or she would never have been observed.

If multiple failures for the same subject are possible, then truncation is defined as a period during which you do not know how many (or even if) events occurred during the period.

However, for the purposes of this discussion, we will restrict ourselves to absorbing failure events such as death.

4.2.1 Left truncation (delayed entry)



In left truncation, the period of ignorance extends from on or before the onset of risk to some time after the onset of risk. For a while, the subject was not observed, but then the subject came under observation. Left truncation usually arises because we encounter a subject who came at risk some time ago.

Can we include this subject in our study? The answer is yes, but we must account for the fact that, had the subject failed earlier, we never would have encountered this subject. The subject's subsequent survival can be analyzed, but we do not want to make too much out of the fact that the subject survived up until the point we encountered him. Consider an 89-year-old smoker who just arrives to enroll in our study. Does he supply whopping evidence that smoking lengthens life? No, because, had he died, he would have never enrolled in our study; the only way this 89-year-old made it to our study was conditioned upon him having survived all 89 prior years.

In parametric models, left truncation is easily dealt with. The contribution to the overall likelihood of subject i (when failure time is measured exactly and without truncation) is

$$L_i(\beta|t_i, \mathbf{x}_i) = S(t_i|\mathbf{x}_i, \beta)h(t_i|\mathbf{x}_i, \beta)$$

If subject i , who is ultimately observed to fail at time t_i , arrived late to our study (say, at time t_{0i}), then we simply need to add one more condition to subject i 's contribution to the likelihood; namely, that he had already survived up until time t_{0i} , which has probability $S(t_{0i}|\mathbf{x}_i, \beta)$,

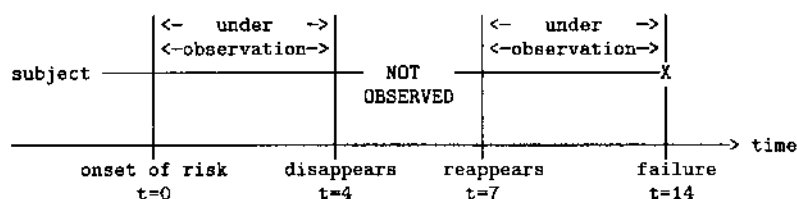
$$L_i(\beta|t_i, \mathbf{x}_i) = \{S(t_i|\mathbf{x}_i, \beta)/S(t_{0i}|\mathbf{x}_i, \beta)\}h(t_i|\mathbf{x}_i, \beta) \quad (4.2)$$

because $S(t_i|\mathbf{x}_i, \beta)/S(t_{0i}|\mathbf{x}_i, \beta)$ is the probability of surviving to t_i , given survival up to t_{0i} .

In fact, (4.2) even applies to cases where there is no left truncation and the subject enrolled at time $t_{0i} = 0$, since $S(0|\mathbf{x}_i, \beta) = 1$.

Left truncation is easily handled in semiparametric models, too. In that case, one simply must omit the subject from all individual binary-outcome analyses during the truncation period because the subject could not possibly have failed during that period and still manage to be around when the period ended. In other words, we cannot treat the subject as being at risk of failure during the truncation period since the only reason we observe him at all is because he survived the period.

4.2.2 Interval truncation (gaps)



Interval truncation is just a variation on left truncation. In this case, think of a subject who disappears for a while but then reports back to the study, causing a gap in our follow-up. The statistical issue, of course, is that had the subject died, he or she never could have reported back. Throughout this text, and throughout the Stata documentation, we treat the terms “gaps” and “interval intruncation” as synonyms.

For parametric models, assume that subject i is observed up until time t_{1i} goes unobserved until time t_{2i} , and then is observed up until time t_{3i} , at which point the subject dies. Then the probability of observing those parts that we did observe is

$$S(t_{1i}|\mathbf{x}_i, \beta) \{ S(t_{3i}|\mathbf{x}_i, \beta) / S(t_{2i}|\mathbf{x}_i, \beta) \} h(t_{3i}|\mathbf{x}_i, \beta)$$

and thus this is subject i 's contribution to the overall likelihood, denoted by L_i .

In semiparametric models, this problem is handled the same way as left truncation. One simply omits the subject from all individual binary-outcome analyses during the truncation period because the subject could not possibly have failed at those times.

4.2.3 Right truncation

In point of fact, right truncation is indistinguishable from right censoring, which was previously discussed. There is a point beyond which the subject is not observed, and since time may extend all the way to infinity, failure is certain to occur eventually.

5 Recording survival data

5.1 The desired format

Given the possibility of censoring and truncation, the simple way we have shown for the recording of survival data, using a single observation per subject and recording just time of failure,

time	x
1	3
5	2
9	4
20	9
22	10

is simply not adequate. Instead, we want to record (and think about) survival data as follows:

id	t0	t1	outcome	x
1	0	1	1	3
2	0	5	1	2
3	0	9	1	4
4	0	20	1	9
5	0	22	1	10

In this format, we make explicit the beginning and ending times of each record, using variables `t0` and `t1`—recording the analysis times during which the subject was under observation—and we record the outcome at the end of the span in variable `outcome`, where `outcome==1` denotes a failure and `outcome==0` denotes nonfailure (censoring). You can call the variables what you wish, but these three variables concisely describe the information.

This method will allow us to record observations that are right censored,

id	t0	t1	outcome	x
1	0	1	1	3
2	0	5	1	2
3	0	9	1	4
4	0	20	1	9
5	0	22	0	10

`<- id 5 did not fail yet`

that have left truncation,

id	t0	t1	outcome	x
1	0	1	1	3
2	0	5	1	2
3	3	9	1	4
4	0	20	1	9
5	0	22	0	10

<- id 3 entered late, at t=3

or that have interval truncation,

id	t0	t1	outcome	x
1	0	1	1	3
2	0	5	1	2
3	3	9	1	4
4	0	9	0	9
4	11	20	1	9
5	0	22	0	10

<- id 4 was unobserved
<- between times 9 and 11

Do you see in this last example how the subject with `id==4` was not observed between times 9 and 11? The first record for the subject records the span 0 to 9, and the second records the span 11 to 20. Because there is no record for the span 9 to 11, the subject was unobserved (or perhaps just not at risk).

Each record in the data records a time span during which the subject was under observation, and it includes an outcome variable recording whether a failure was observed at the end of the span. Subjects may have multiple records, as does subject 4 in the example above, and so subjects can have delayed entry, multiple gaps, and truly complicated histories. For example, look at subject 5 below:

id	t0	t1	outcome	x
1	0	1	1	3
2	0	5	1	2
3	3	9	1	4
4	0	9	0	9
4	11	20	1	9
5	2	4	0	10
5	6	8	0	10
5	10	15	0	10
5	17	21	0	10

<- subj. enrolled late, disappeared
<- then reappeared for a while
<- then reappeared for a while again
<- finally reappeared and was censored

Stata can understand data presented in this manner. Every one of its statistical commands will do the statistically appropriate thing with these data, which is really not as amazing as it looks. If you examine the developments of the previous chapter, you will find that they follow a pattern:

For parametric analysis, for each time span,

1. The likelihood contribution is $S(t_1|x_i, \beta)/S(t_0|x_i, \beta)$ for the observational period.
2. The likelihood contribution is multiplied by $h(t_1|x_i, \beta)$ if the period ends in a failure event.

For semiparametric analysis,

1. Individual studies are performed at all t 's for which `outcome` is equal to 1.
2. Each observation enters the individual study at time t if for that subject $t_0 < t < t_1$.

Moreover, these individual time-span observations record the independent pieces of information of your data, meaning that records can be omitted from analysis without bias, even if that means including only pieces of a subject's history in an analysis. There is, of course, a loss of efficiency, but there is no bias. Let us explain.

First, understand that there is no difference in recording

id	t0	t1	outcome	x	
2	0	5	1	2	(subject observed between 0 and 5)

OR

id	t0	t1	outcome	x	
2	0	2	0	2	(subject observed between 0 and 2)
2	2	5	1	2	(subject observed between 2 and 5)

OR

id	t0	t1	outcome	x	
2	0	2	0	2	(subject observed between 0 and 2)
2	2	4	0	2	(subject observed between 2 and 4)
2	4	5	1	2	(subject observed between 4 and 5)

All three of these variations record the same information: subject 2 was under continuous observation between analysis times 0 and 5 and, at $t = 5$, exhibited a failure. Why would you want to split this information into pieces? Perhaps times 0, 2, and 4 were the times the subject reported to the diagnostic center—or filled in a survey—and at those times you gathered information about characteristics that change with time:

id	t0	t1	outcome	x	exercise
2	0	2	0	2	0
2	2	4	0	2	1
2	4	5	1	2	0

You could, if you wished, restrict your analysis to the subsample `exercise==0`. That would make it appear as if this subject had a gap between times 2 and 4. In truth, the subject was really not outside of observation, but he or she would still be interval truncated for the purposes of this analysis. Or, you could restrict your analysis to the subsample `exercise==1`, in which case this subject would appear to enter late into the sample and be right censored.

You could even include `exercise` among the explanatory variables in your model, stating that the hazard shifted according to the current value of this variable.

Finally, this format allows recording multiple-failure data. In the following example, the subject fails at times 4 and 7, and even after the second failure, we observe the subject a little while longer:

id	t0	t1	outcome	x	exercise
14	0	2	0	2	0
14	2	4	1	2	1
14	4	5	0	2	1
14	5	7	1	2	0
14	7	9	0	2	0

In our theoretical discussions, we seldom mentioned failure events that could reoccur, but it is an easy enough generalization. Assuming the failure events are of the same kind, such as first heart attack and so on, the main issue is how you want to treat second and subsequent failure events. Is the hazard of failure at times 4 and 7 equal to $h(4)$ and $h(7)$, respectively, meaning the clock keeps ticking; or is it $h(4)$ and $h(3)$, meaning the clock gets reset; or is it $h_1(4)$ and $h_2(3)$, meaning they are on different clocks altogether?

□ Technical Note

In contexts other than survival analysis, the “size” of a dataset is usually taken to be the number of physical observations or data records, where (all other things equal) datasets of larger sizes result in more efficient estimates of model parameters, since the larger datasets contain more information. However, as we demonstrate above, with survival data, measuring the “size” of a dataset in this manner is misleading, since we could use any arbitrarily large number of physical observations to express exactly the same information by splitting records as frequently as needed to obtain any number of physical observations.

With survival data, measuring the amount of information contained in your data (relative to other datasets) is thus not as simple as counting the physical observations—the notion of “sample size” is different. One way to measure sample size would be to count (at a particular time) the number of subjects who are at risk of failure. In a semiparametric analysis, such a measure of sample size would correspond to the sample size for the binary-outcome analysis performed at that time. Regardless of how the survival data are recorded, the sample size for the binary analysis is the same, provided that the information contained in the survival data remains the same. As time passes, the number of subjects at risk of failure will change. Thus, if we choose to measure sample size as the number at risk, then the sample size will change with time. For this reason, the application of methods for determining required sample sizes (in order to achieve certain power at a certain level) can be quite difficult.

With survival data, another (time-invariant) measure of the sample size is to take the total time at risk for all subjects in the study.

□

5.2 Other formats

Unless you have the dubious pleasure of having entered the data yourself, the data probably do not come in the ever-so-desirable duration format. Instead of seeing

id	t0	t1	outcome	x	exercise
2	0	2	0	2	0
2	2	4	0	2	1
2	4	5	1	2	0

you have something that looks like

id	t	event	x	exercise
2	0	enrolled	2	0
2	2	checkup		1
2	4	checkup		0
2	5	failed		

or like

id	x	t1	event1	ex1	t2	event2	ex2	t3	event3	ex3	...
1	2	0	enrolled	0	2	checkup	1	4	checkup	1	...

This is called snapshot or transaction data. The first example is called “long-form snapshot data”, and the second, “wide-form snapshot data”. Each observation records a time and the measurements taken at that time. As long as we are being realistic about this, the data probably do not record analysis time, but instead calendar time, so in the long form, the data might appear as

id	date	event	x	exercise
2	20jan2000	enrolled	2	0
2	22jan2000	checkup		1
2	24jan2000	checkup		0
2	25jan2000	failed		

and would appear similarly in the wide form. The event variable is seldom called that. Instead, it is often called a record type, especially when the data are in the long form, and the variable is often coded numerically. Record type 1 means an enrollment, 2 a checkup, and 9 a failure, for instance.

id	date	rectype	x	exercise
2	20jan2000	1	2	0
2	22jan2000	2		1
2	24jan2000	2		0
2	25jan2000	9		

Anyway, the problem that you must address—and you must address it long before you get to use your first survival analysis command—is to convert these data into the duration format.

For now, ignore the date versus analysis time issue—that the “time” variable in the data might not correspond to analysis time. Let’s focus on what must happen to convert these data into duration data. Later, after conversion, we can worry about defining a reasonable analysis-time variable. Also, for now we consider only the problem

of long-form snapshot data because, if we can solve that problem, we can solve the other, since converting wide-form data into long-form data is easy in Stata using the `reshape` command; see [R] [reshape](#).

Let's consider the following set of transactions:

id	date	rectype	x	exercise
2	20jan2000	1	2	0
2	22jan2000	2		1
2	24jan2000	2		0
2	25jan2000	9		

What we are going to do is form a record from each pair of these records that contains a beginning date, an ending date, and the appropriate values. In choosing these appropriate values, we need to realize that snapshot datasets really contain two kinds of variables:

1. *Enduring or characteristic variables.*

These variables state characteristics that are valid for longer than an instant, or which you are at least willing to treat as valid for longer than an instant. A subject's gender, their age rounded to years, and blood pressure at the time of hospitalization are examples. In the examples above, we will assume that `x` and `exercise` are enduring variables.

2. *Instantaneous or event variables.*

These variables state events that occurred at an instant in time, such as having a heart attack, going bankrupt, finding a job, or other failure-type events. In the example above, `rectype` is an instantaneous variable because it records, among other things, whether the subject failed. In some other dataset, `rectype` might not record that (it might just record whether the record was a result of a visit or a telephone interview), and some other variable might record whether the subject failed at that instant. In that case, both of those variables would be instantaneous variables. The interview process does not continue after the date, nor does the act of failing.

Our goal is to take pairs of records, such as the first two in our snapshot dataset,

id	date	rectype	x	exercise
2	20jan2000	1	2	0
2	22jan2000	2		1

and form one time-span record from them:

id	date0	date1	rectype	x	exercise
2	20jan2000	22jan2000	?	?	?

The question is, what is the value of each variable over this time span? For the enduring variables, we will obtain their values from the first observation of the pair:

id	date0	date1	rectype	x	exercise
2	20jan2000	22jan2000	?	2	0

For the instantaneous variables, we will obtain the value from the second observation of the pair:

id	date0	date1	rectype	x	exercise
2	20jan2000	22jan2000	2	2	0

The reason for this is as follows. In duration data, we want a span of time and the values of the variables over that particular span of time, and we want events that occurred at the *end of the period*. So, we are going to ignore the values of the transient variables in the first-of-pair observation and get those values from the second-of-pair observation. The net result of this will be to discard the values of the instantaneous variables recorded in the first observation of the snapshot dataset. It is dangerous to discard any data because you never know when you are going to need the data. In order to get that back, when we look at our snapshot data,

id	date	rectype	x	exercise
2	20jan2000	1	2	0
2	22jan2000	2		1
2	24jan2000	2		0
2	25jan2000	9		

we will pretend that we see

id	date	rectype	x	exercise	
2	<- new
2	20jan2000	1	2	0	
2	22jan2000	2		1	
2	24jan2000	2		0	
2	25jan2000	9			

and convert those records. The result, then, of carrying out this procedure is

id	date0	date1	rectype	x	exercise
2	.	20jan2000	1	.	.
2	20jan2000	22jan2000	2	2	0
2	22jan2000	24jan2002	2	.	1
2	24jan2002	25jan2000	9	.	0

Thus, the interpretation of a single record is

1. A record spans the time period **date0** to **date1**.
2. The values of the enduring variables (in this case **x** and **exercise**) are the values they had on **date0**.
3. The values of the instantaneous variables (in this case **rectype**) are the values they had on **date1**.

At this point, you may be concerned about the missing values for **x**. Would it not be better if the enduring variables had their values carried down from one observation to the next? We agree, but let's not try to solve all the problems at one time. We could solve this problem later by typing

```
. sort id date1
. by id: replace x = x[_n-1] if x>=.
```

The hard part is transforming

id	date	rectype	x	exercise
2	20jan2000	1	2	0
2	22jan2000	2		1
2	24jan2000	2		0
2	25jan2000	9		

into

id	date0	date1	rectype	x	exercise
2	.	20jan2000	1	.	.
2	20jan2000	22jan2000	2	2	0
2	22jan2000	24jan2002	2	.	1
2	24jan2002	25jan2000	9	.	0

but we do not have to do this manually. Stata has one command, `snapspan`, which will do that for us. Its syntax is

```
snapspan idvar time_var instantaneous_vars, generate(new_begin_date)
```

and so, to convert this dataset, all we need to type is

```
. snapspan id date rectype, gen(date0)
. rename date date1
```

That just leaves the problem of defining analysis time t in terms of the date and time variables already in our data. That also turns out to be no problem at all, but we have to get a little ahead of ourselves.

Once we have the data in duration form,

id	date0	date1	rectype	x	exercise
2	.	20jan2000	1	.	.
2	20jan2000	22jan2000	2	2	0
2	22jan2000	24jan2002	2	.	1
2	24jan2002	25jan2000	9	.	0

we use a command called `stset` to tell Stata about the data, and in that process, we can tell `stset` how to form the analysis-time variable for us. Basically, given the above, we might type

```
. stset date1, id(id) time0(date0) origin(time date0) failure(rectype==9)
```

and why we would do that is covered in the next chapter.

5.3 Example

Let's consider the following dataset that is in wide form:

```
. use http://www.stata-press.com/data/cgg/wide1
. describe
Contains data from http://www.stata-press.com/data/cgg/wide1.dta
  obs:                3
 vars:                14      28 Jan 2002 14:42
size:                180 (99.9% of memory free)
```

variable name	storage type	display format	value label	variable label
id	float	%9.0g		Subject id
sex	float	%9.0g		Sex (1=female)
date1	float	%d		1st interview
event1	float	%9.0g		
x1	float	%9.0g		
date2	float	%d		2nd interview
event2	float	%9.0g		
x2	float	%9.0g		
date3	float	%d		3rd interview
event3	float	%9.0g		
x3	float	%9.0g		
date4	float	%d		4th interview
event4	float	%9.0g		
x4	float	%9.0g		

Sorted by:

```
. list
```

1.	id	sex	date1	event1	x1	date2	event2	x2
	1	0	20jan2000	1	5	22jan2000	9	3
	date3		event3	x3	date4		event4	x4
		
2.	id	sex	date1	event1	x1	date2	event2	x2
	2	1	14feb2000	1	8	18feb2000	2	5
	date3		event3	x3	date4		event4	x4
	22feb2000		2	5			.	.
3.	id	sex	date1	event1	x1	date2	event2	x2
	3	0	11nov1999	1	2	14nov1999	2	2
	date3		event3	x3	date4		event4	x4
	18nov1999		2	3	22nov1999		9	3

In order to convert the data into the desired duration format, we must first **reshape** the data and then issue a **snapspan** command:

```
. reshape long date event x, i(id)
(note: j = 1 2 3 4)
```

Data	wide	->	long
Number of obs.	3	->	12
Number of variables	14	->	6
j variable (4 values)		->	_j
xij variables:			
date1 date2 ... date4		->	date
event1 event2 ... event4		->	event
x1 x2 ... x4		->	x

```
. drop if missing(date)
(3 observations deleted)
. snapspan id date event, gen(date0) replace
. list, noobs sepby(id)
```

id	_j	sex	date	event	x	date0
1	.	.	20jan2000	1	.	.
1	1	0	22jan2000	9	5	20jan2000
2	.	.	14feb2000	1	.	.
2	1	1	18feb2000	2	8	14feb2000
2	2	1	22feb2000	2	5	18feb2000
3	.	.	11nov1999	1	.	.
3	1	0	14nov1999	2	2	11nov1999
3	2	0	18nov1999	2	2	14nov1999
3	3	0	22nov1999	9	3	18nov1999

We admit that we left out the `drop if missing(date)` on our first try, and when we did, we got

```
. snapspan id date event, gen(date0) replace
3 observations have date==.
either fix them or drop them
by typing drop if date==.
r(459);
end of do-file
r(459);
```

So, we followed the instructions and dropped the observations with missing `date`. **snapspan** refuses to work with records for which it does not know the date. In our original wide dataset, we had missing dates when we had no information. Note that we also specify the `replace` option to **snapspan**, which just permits Stata to change the data in memory without requiring that the data be saved first (after **reshape** and **drop** in this case.)

6 Using stset

We left off in the last chapter having agreed that we would record our survival data in duration format:

id	date0	date1	event	x	exercise
1	20jan2000	21jan2000	9	3	1
2	15dec1999	20dec1999	6	2	0
3	04jan2000	13jan2000	4	4	1
4	31jan2000	08feb2000	3	9	1
4	10feb2000	19feb2000	9	9	0
5	12jan2000	14jan2000	3	10	0
5	16jan2000	18jan2000	3	10	1
5	20jan2000	25jan2000	3	10	1
5	27jan2000	01feb2000	9	10	0

Each record above documents a span of time for a subject. We have made our example realistic in that, rather than recording analysis time measured from the instant of the onset of risk, our data records calendar dates. Our dataset contains two types of variables, instantaneous and enduring, and we just know which are which.

In this example, `event` is an instantaneous variable. It records the event that happened at the end of the time span, which is to say, `date1`. In this case, our `event` variable is coded so that `event==9` means failure, and the other codes mean something else.

The remaining two variables, `x` and `exercise`, are enduring variables, meaning that they hold their given values over the time span indicated, or at least we are willing to treat them that way.

The first step is to `stset` the data. `stset`'s simplest syntax is

```
. stset time
```

but that is for those with a simple dataset like

time	x
1	3
5	2
9	4
20	9
22	10

In this dataset, every observation records a failure time, and that is it; every observation is assumed to fail, and `time` is already defined as or assumed to be analysis time. `stset`'s second simplest syntax is

```
. stset time, failure(failed)
```

and that is for datasets like

time	failed	x
1	1	3
5	1	2
9	1	4
20	1	9
22	0	10

Every observation records a failure or censoring time, and the variable **failed** tells us which.

An example of **stset**'s more complete syntax is

```
. stset date1, id(id) time0(date0) origin(time date0) failure(event==9)
```

and that is for those of us who have more complicated survival datasets such as

id	date0	date1	event	x	exercise
1	20jan2000	21jan2000	9	3	1
2	15dec1999	20dec1999	6	2	0
3	04jan2000	13jan2000	4	4	1
4	31jan2000	08feb2000	3	9	1
4	10feb2000	19feb2000	9	9	0
5	12jan2000	14jan2000	3	10	0
5	16jan2000	18jan2000	3	10	1
5	20jan2000	25jan2000	3	10	1
5	27jan2000	01feb2000	9	10	0

6.1 A short lesson on dates

Much of **stset**'s convenience stems from its interaction with dates stored in Stata's date format. Dates are recorded as integers representing the number of days from January 1, 1960. That is, 0 means January 1, 1960, 1 means January 2, 1960, -1 means December 31, 1959, and so on.

What makes dates display in the nice form 01jan1960, 21may1872, etc., is that we put a date format on this numeric variable. So, in fact, if we have a dataset that looks like

id	date0	date1	event	x	exercise
1	14629	14630	9	3	1
2	14593	14598	6	2	0
3	14613	14622	4	4	1
4	14640	14648	3	9	1
4	14650	14659	9	9	0
5	14621	14623	3	10	0
5	14625	14627	3	10	1
5	14629	14634	3	10	1
5	14636	14641	9	10	0

we type

```
. format date0 %d
. format date1 %d
```

and we get the nicely formatted dates from above. The convenient thing about storing dates in this way is that we can subtract them to obtain the number of days between dates. For instance, if we were curious as to how many days there were between `date1` and `date0`, we could type

```
. gen span = date1 - date0
```

and the result would be

id	date0	date1	event	x	exercise	span
1	20jan2000	21jan2000	9	3	1	1
2	15dec1999	20dec1999	6	2	0	5
3	04jan2000	13jan2000	4	4	1	9
4	31jan2000	08feb2000	3	9	1	8
4	10feb2000	19feb2000	9	9	0	9
5	12jan2000	14jan2000	3	10	0	2
5	16jan2000	18jan2000	3	10	1	2
5	20jan2000	25jan2000	3	10	1	5
5	27jan2000	01feb2000	9	10	0	5

Dates are just integers, and so all the ordinary mathematical operators work on them.

Converting dates recorded in your raw data into Stata date variables is a little tricky. The procedure is (1) read the data into string variables, (2) convert the strings into Stata date variables using the `date()` function, and (3) format the variable with the `%d` format.

1. The first step, reading the dates into a string variable, can itself be difficult. If the date recorded in the original is all run together—such as 1/21/98, 1/21/1998, 21/1/98, 21/1/1998, 21-1-1998, 21jan1998, 21January1998, etc.—then it is easy enough. Whatever method you use to read your data—infile, infix, etc.—Stata likes to keep things together.

Problems arise when the original date is recorded with blanks, such as 21 jan 1998, January 21, 1998, etc. Fortunately, most datasets on computers seldom come in this format but, when they do, Stata wants to break the pieces into separate variables unless the whole thing is in quotes.

In this case, it is best to just read the date into separate variables, making a day, month, and year variable (in whatever order and in whatever format). You can then assemble these components into a single string variable and then convert this variable into an integer with a date format. As an example, pretend you have just read your data and are now left with string variable `month`, numeric variable `day`, and numerical variable `year`. Assemble them into one string by typing

```
. gen date = string(day) + " " + month + " " + string(year)
```

If `month` is instead a numeric variable (1, 2, ..., 12), put a `string()` function around it to convert it into a string.

2. The second step, converting the string variable containing the date into a Stata date variable, is easy. Type

```
. gen mydate = date(date, "dmy")
```

The second argument to `date()` specifies the order of the day, month, and year in the string `date`. The argument `dmy` means day followed by month followed by year. Thus, if your string date is instead in the order month, day, year, type `mdy`.

This works assuming your year variable is a full, four-digit year. If your years are two-digit years (98 meaning 1998, 02 meaning 2002), type

```
. gen mydate = date(date, "dmy", 2040)
```

The last argument gives Stata a rule for converting two-digit years to four-digit years. Specifying 2040 says that, in making the translation, the largest year to be produced is 2040. So, year 39 would be interpreted as 2039 and year 40 as 2040, but year 41 is interpreted as 1941. Specify a reasonable value, given your data, for this third argument.

3. The third step is just a matter of putting a date format on your new date variable. Type

```
. format mydate %d
```

If later you want to see the date in its numeric form, change the format back to a numeric format.

```
. format mydate %8.0g
```

You can then switch back and forth at will.

6.2 The purpose of the *stset* command

The first purpose of the *stset* command is to tell Stata about the structure of your survival data so that you do not have to repeat that information over and over again every time you issue a survival command. This reduces the likelihood of mistakes.

The second purpose of the *stset* command is to burn considerable computer time by performing checks to verify that what you claim as true indeed makes sense. You would complain about performance if these checks were done every time you issued a survival analysis command, and because of that, most software packages simply never check at all. Stata checks when you *stset* your dataset, and it checks again anytime you type *stset* without arguments.

The third purpose of *stset* is to allow you to describe complicated rules for when an observation is included and excluded, what defines onset of risk and failure, and how analysis time is defined in terms of the time and date variables that are in your data.

6.3 The syntax of the `stset` command

Given everything the `stset` command can do, the syntax is truly impressive. `stset` is a wonderful command once you know it, but it has become a hurdle that may look insurmountable to new users—one only need to look at [R] `st setset` to see how daunting this command can appear.

To break through this, the basic syntax of `stset` is

```
stset time_of_failure_var
```

or

```
stset time_of_failure_or_censoring_var, failure(one_if_failure_var)
```

Everything else is a detail. The time variable specified immediately following `stset` specifies when the failure or censoring occurs. The optional *one_if_failure_var* contains 1 if the observation contains a failure and 0 otherwise.

Examples

1. All observations record failure times:

failtime	x
1	3
5	2
9	4
20	9
22	10

The `stset` command is

```
. stset failtime
```

because `failtime` is the name of the variable in this dataset that contains the failure time.

2. Some observations are censored:

lasttime	x	failed
1	3	1
5	2	1
9	4	1
20	9	1
22	10	0

The `stset` command is

```
. stset lasttime, failure(failed)
```

because `failed` is the name of the variable in this dataset that contains 1 when variable `lasttime` records a failure time and 0 when `lasttime` records a censoring time.

6.3.1 Specifying analysis time

Recall from Section 3.3 our discussion of analysis time. It is 0 at the onset of risk, and so it is probably shifted from calendar time by a different amount for each subject—analysis time might be time from diagnosis or time from beginning of unemployment. The duration might not even be a time measure but instead a time-intensity measure such as cigarette-days, or in rare instances, analysis time might have nothing whatsoever to do with time—it might be distance from the end of a bolt of cloth.

In any case, before any analysis can begin, you need to define analysis time:

$$t = \text{somefunction}(\text{time/date variables in your data})$$

If you are lucky, all the time/date variables in your data are already in terms of analysis time, but in case they are not, you have two alternatives:

1. Define *t* yourself and then **stset** your data in terms of *t*.
2. Specify **stset**'s **origin()** and possibly **scale()** options so that **stset** can calculate *t* based on the time/date variables in your data.

We prefer the latter because **stset** has lots of error checking built into it.

From now on, we are going to write *time* to mean time as you have it recorded in your data and *t* to mean analysis time. Let *time* be a variable recorded in the units of time for your data. We obtain *t* from

$$t = \frac{\text{time} - \text{origin}}{\text{scale}}$$

The default definitions for *origin* and *scale* are 0 and 1, respectively; that is, by default, *time* = *t*.

By specifying **stset**'s **origin()** and **scale()** options, you can change the definitions of *origin* and *scale*. For instance, in your data, *time* might be calendar date, and you might want to define **origin()** as the date of diagnosis, thus making time from diagnosis analysis time and diagnosis itself as denoting the onset of risk.

Recall that *t* = 0 is the time of the onset of risk, and that Stata treats this very seriously: all of the survival analysis commands ignore data before *t* = 0. They do this automatically, irrevocably, and without emphasizing the fact. Mostly, this is desirable, but sometimes a user is surprised. Consider the user for whom calendar time is the appropriate definition of *t*. Now remember how Stata records dates: *t* = 0 corresponds to January 1, 1960. This means that

1. Without specifying **origin()**, records before January 1, 1960 are ignored, even if the user is only interested in estimating semiparametric models for which the exact timing of the onset of risk is otherwise irrelevant. If the user has data going back before January 1, 1960, **origin()** should be specified to be some date before then.

2. Risk begins accumulating on January 1, 1960. If the user is fitting a parametric model, he or she may want to think carefully about this, especially if the earliest date in the data is much later than this.

Defining t to be the calendar date is indeed a rare occurrence, and we emphasize that such a definition usually involves an explicit definition of `origin()`. If t is already defined to be the analysis time (with a meaningful origin), then `origin()` need not be specified.

The `scale()` option is usually of no analytical significance. Most people use `scale()` to put t in more readable units. They define a reasonable `origin()` so that $t = \text{time} - \text{origin}$ provides an analytically sufficient definition but then decide they would rather have time recorded in months rather than in days, and so define `scale(30)`, meaning $t = (\text{time} - \text{origin})/30$. `scale()` can also be used to define time-intensity measures.

Anyway, focusing on the ever-so-important `origin()` option, here are some valid examples:

- `origin(time 20)`

This sets `origin = 20`. It is a rather odd thing to do, since it specifies that *time* is almost analysis time, but it needs 20 subtracted from it; the onset of risk occurred at $\text{time} = 20$.

- `origin(time d(15feb1999))`

This sets `origin = 14290`, the integer value for the date February 15, 1999. Matching on calendar date is appropriate for date-formatted data, and for this example the onset of risk occurred on February 15, 1999. Writing `d(15feb1999)` is just how to write date constants in Stata and is preferable to having to figure out that February 15, 1999 translates to the integer 14290.

- `origin(time bdate)`

This sets `origin = bdate`, where `bdate` is a variable in the dataset, which we assume records the birthdate of subjects, so analysis time is now age and onset of risk is now birth.

There is a little something extra to think about here: If the data contain multiple records for some subjects, which value of `bdate` should be used for the onset of risk? Well, your first reaction is probably that one would hope that `bdate` was the same across all the records for the same subject. So would we, but perhaps `bdate` is missing in some records. Perhaps `bdate` is only recorded on the first record of every subject, or only the last. It would not matter. Stata would find the nonmissing `bdate` and use it.

If `bdate` took on different values in different records within a subject, Stata would find and use the earliest value of `bdate`. In this case, that is not so useful, and you would probably prefer that Stata flag this as an error. In other cases, however, it is useful; for instance:

- `origin(time diagdate)`

This sets *origin* = *diagdate*, *diagdate* being a variable in the dataset. If subjects have multiple records, Stata will use the earliest (smallest) nonmissing value of *diagdate*, meaning that if a subject was diagnosed twice, it would be the first diagnosis date that mattered.

In any case, analysis time is now time from (earliest) diagnosis, and (earliest) diagnosis corresponds to the onset of risk.

- `origin(time min(diagdate,d2date))`

You can specify any expression for *origin*, and the one we just chose is the minimum value recorded in the two variables *diagdate* and *d2date*. Perhaps *diagdate* is the actual date of diagnosis and *d2date* is the judgment by a professional, for a few patients, as to when diagnosis would have occurred had the patients shown up earlier at the health center.

In any case, analysis time is now the earliest time of either the earliest diagnosis or imputed diagnosis, and (possibly imputed) diagnosis is the onset of risk.

- `origin(event==3)`

This is something new: *event* is a variable in the data. We are instructing Stata to thumb through the records for each patient and find the earliest date at which the variable *event* took the value 3. In doing this, variable *event* is assumed to be an instantaneous variable, meaning that it occurs at the instant of the end of the time span, so it is the end-of-span time that is important.

This construct is very useful in multiple-record situations. We have multiple records on each subject, and we have a variable that records what happened on various dates. In these situations, it is not uncommon for our failure event to be one value of the variable and for our onset-of-risk event to be another value of the same variable.

In any case, analysis time is now the time from *event*==3, and the occurrence of *event*==3 is the onset of risk. If *event*==3 never occurs, then the subject is automatically excluded from the analysis. If *event*==3 occurs more than once for a subject, it is the earliest occurrence that matters.

One final note: note that *event* is the name of a variable. If your event variable were named *code*, then you would type `origin(code==3)`. Distinguish this from the word *time* in, for instance, `origin(time diagdate)`. There you always type the word *time* and follow that by the name of your particular variable. (Actually, if you omit the word *time*, Stata will probably still understand you, and if it does not, Stata will complain.)

- `origin(event==3 4 7)`

This is just a variation on the above. You may specify more than one number following the double-equals sign, and in fact, you may specify what Stata calls a "numlist". See `help numlist` in Stata for more information on numlists. In this case, Stata will select the earliest occurrence of any of these three events.

- `origin(event==3 4 7 time stdate)`
This sets onset of risk to be the earliest of (1) the earliest time that variable `event` takes on the value 3, 4, or 7, and (2) the earliest time recorded in variable `stdate`.
- `origin(min)`
Do not specify this for the purposes of analysis. `origin(min)` is for playing a trick on Stata to get the data `stset`, which is useful for performing some data-management tasks. We will discuss this later.

6.3.2 Variables defined by *stset*

After you `stset` your data, you will find four new variables named `_t0`, `_t`, `_d`, and `_st` in your dataset.

- `_t0` and `_t`
These two variables record the time span in analysis time, *t*, units for each record. Each record starts at `_t0` and concludes at `_t`.
- `_d`
This variable records the outcome at the end of the span and contains 1 if the time span ends in a failure and 0 if it does not.
- `_st`
This variable records whether this observation is relevant (is to be used) in the current analysis. For each observation, the variable contains 1 if the observation is to be used and 0 if it is to be ignored. For example, if we specify `origin(time d(19feb1999))`, then observations dated before February 19, 1999, will have `_st` set to 0.

At a technical level, the purpose of `stset` is to set these variables, and then the other `st` commands—the analysis commands—do their calculations using them and ignoring your original time/date and event variables. If `_t0`, `_t`, `_d`, and `_st` are set correctly, it does not matter which `stset` command you typed. If you have them set incorrectly, it does not matter how much you think the `stset` command should have set them differently.

▷ Example

Let's go ahead and load the duration format dataset considered at the beginning of this chapter and `stset` it.

```
. use http://www.stata-press.com/data/cgg/stset_ex1
. list, noobs sepby(id)
```

id	date0	date1	event	x	exercise
1	20jan2000	21jan2000	9	3	1
2	15dec1999	20dec1999	6	2	0
3	04jan2000	13jan2000	4	4	1
4	31jan2000	08feb2000	3	9	1
4	10feb2000	19feb2000	9	9	0
5	12jan2000	14jan2000	3	10	0
5	16jan2000	18jan2000	3	10	1
5	20jan2000	25jan2000	3	10	1
5	27jan2000	01feb2000	9	10	0

```
. stset date1, origin(time date0) id(id) failure(event==9) time0(date0)
(output omitted)
. list id date0 date1 _t0 _t _d _st, noobs sepby(id)
```

id	date0	date1	_t0	_t	_d	_st
1	20jan2000	21jan2000	0	1	1	1
2	15dec1999	20dec1999	0	5	0	1
3	04jan2000	13jan2000	0	9	0	1
4	31jan2000	08feb2000	0	8	0	1
4	10feb2000	19feb2000	10	19	1	1
5	12jan2000	14jan2000	0	2	0	1
5	16jan2000	18jan2000	4	6	0	1
5	20jan2000	25jan2000	8	13	0	1
5	27jan2000	01feb2000	15	20	1	1

We have used options that we have not yet explained just to make *stset* work. However, the only thing we want you to focus on right now is *_t0* and *_t* and to remember that we typed

```
. stset date1, origin(time date0) ...
```

The variables *_t0* and *_t* record the time span of each record in *analysis time* units; the original *date0* and *date1* record the same thing in *time* units.

◁

6.3.3 Specifying what constitutes failure

In your data, either you already have a variable that marks the failures—contains 1 if failure and 0 otherwise—or you have a variable that contains various codes such that, when the code is a certain value, it means failure for the purposes of this analysis.

`stset`'s `failure()` option specifies the failure event.

1. *Simple example:* Variable `failed` contains ones and zeros, where `failed==1` means failure:

time	failed	x
1	1	3
5	1	2
9	1	4
20	1	9
22	0	10

The `failure()` option for this is

```
. stset time, failure(failed)
```

2. *Another example:* Variable `failed` contains various codes: 0 means nonfailure, and the other codes indicate that failure occurred and the reason:

time	failed	x
1	1	3
5	7	2
9	2	4
20	8	9
22	0	10

If you want to set this dataset so that all failures regardless of reason are treated as failures, then the `failure()` option would be

```
. stset time, failure(failed)
```

3. *A variation on the previous example:* Using the same dataset, you want to set it so that only failure for reason 9 is treated as a failure, and other values are treated as nonfailures (censorings). In this case, the `failure()` option would be

```
. stset time, failure(failed==9)
```

4. *A surprising example:* Variable `failed` contains zeros, ones, and missings.

time	failed	x
1	1	3
5	.	2
9	.	4
20	8	9
22	0	10

This is the same dataset as used in the above example, except that variable `failed` contains missing values in the second and third observations. Ordinarily, you would expect Stata to ignore observations with missing values, but `failure()` is

an exception. By default, if the failure variable contains missing, it is treated as if it contains 0.

If you wanted to treat all failures as failures, regardless of whether the reasons were known, you would type

```
. gen fail = cond(failed!=0,1,0)
. stset time, failure(fail)
```

5. *A more complicated example:* You have multiple records per subject and variable `event` contains various codes, and `event==9` meaning failure (in this analysis):

id	date0	date1	event	x	exercise
1	20jan2000	21jan2000	9	3	1
2	15dec1999	20dec1999	6	2	0
3	04jan2000	13jan2000	4	4	1
4	31jan2000	08feb2000	3	9	1
4	10feb2000	19feb2000	9	9	0
5	12jan2000	14jan2000	3	10	0
5	16jan2000	18jan2000	3	10	1
5	20jan2000	25jan2000	3	10	1
5	27jan2000	01feb2000	9	10	0

The `failure()` option for this is

```
. stset date1, failure(event==9) ...
```

6. *Another more complicated example:* Variable `event` contains various codes, `event` equal to 9, 10, or 11 means failure (in this analysis):

id	date0	date1	event	x	exercise
1	20jan2000	21jan2000	9	3	1
2	15dec1999	20dec1999	6	2	0
3	04jan2000	13jan2000	4	4	1
4	31jan2000	08feb2000	3	9	1
4	10feb2000	19feb2000	11	9	0
5	12jan2000	14jan2000	3	10	0
5	16jan2000	18jan2000	3	10	1
5	20jan2000	25jan2000	3	10	1
5	27jan2000	01feb2000	10	10	0

The `failure()` option for this is

```
. stset date1, failure(event==9 10 11) ...
```

That is, you can specify a numlist following the double-equals sign such as `failure(event==9 10 11)`, or equivalently, `failure(event==9/11)`.

After `stsetting` your data, the new variable `_d` in your dataset marks the failure event, and it contains 1 if failure and 0 otherwise. Looking back at the example at the end of Section 6.3.2, note how the values of `_d` were set. After `stsetting` your data, you can always check that `_d` is as you expect it to be.

6.3.4 Specifying when subjects exit from the analysis

When does a subject exit from the analysis, and at what point are his or her records irrelevant?

The default answer to this question is that a subject exits (1) when its data run out or (2) upon first failure. For some analyses, we may wish subjects to exit earlier or later.

1. If we are analyzing a cancer treatment and a subject has a heart attack, we may wish to treat the subject's data as censored at that point and all subsequent records as irrelevant to this particular analysis.
2. If we are analyzing an event for which repeated failure is possible, such as heart attacks, we may wish the subject to continue in the analysis even after having a first heart attack.

Consider the data on the following subject:

```
. use http://www.stata-press.com/data/cgg/id12
. list
```

	id	begin	end	event	x
1.	12	20jan2000	21jan2000	3	3
2.	12	21jan2000	25jan2000	8	3
3.	12	25jan2000	30jan2000	4	3
4.	12	30jan2000	31jan2000	8	3

First, pretend that the failure event under analysis is `event==9`, which, for this subject, never occurs. Then all the data for this subject would be used, and the corresponding `_d` values would be 0:

```
. stset end, failure(event==9) origin(time begin) id(id) time0(begin)
(output omitted)
. list id begin end _t0 _t _d _st, noobs
```

	id	begin	end	_t0	_t	_d	_st
	12	20jan2000	21jan2000	0	1	0	1
	12	21jan2000	25jan2000	1	5	0	1
	12	25jan2000	30jan2000	5	10	0	1
	12	30jan2000	31jan2000	10	11	0	1

(Continued on next page)

Now, instead pretend that the failure event is `event==8`. By default, Stata would interpret the data as

```
. stset end, failure(event==8) origin(time begin) id(id) time0(begin)
(output omitted)
. list id begin end _t0 _t _d _st, noobs
```

id	begin	end	_t0	_t	_d	_st
12	20jan2000	21jan2000	0	1	0	1
12	21jan2000	25jan2000	1	5	1	1
12	25jan2000	30jan2000	.	.	.	0
12	30jan2000	31jan2000	.	.	.	0

Variable `_st` is how `stset` marks whether an observation is being used in the analysis. In this case, only the first two observations for this subject are relevant because, by default, data are ignored following the first failure. If `event==8` marks a heart attack, for instance, Stata would ignore data on subjects after their first heart attack.

The `exit()` option is how one takes control of when subjects exit. Examples of `exit()` include

- `exit(failure)`

This is just the name for how Stata works by default. When do subjects exit from the analysis? They exit when they first fail, even if there are more data following that failure. Of course, subjects who never fail exit when they run out of data.

- `exit(event==4)`

If you specify the `exit()` option, you take complete responsibility for specifying the exit-from-analysis rules. `exit(event==4)` says that subjects exit when the variable `event` takes on value 4, and that is the only reason except for, of course, running out of data.

If you coded `failure(event==4) exit(event==4)`, that would be the same as coding `failure(event==4) exit(failure)`, which would be the same as omitting the `exit()` option altogether. Subjects would exit upon failure.

If you coded `failure(event==8) exit(event==4)`, subjects would not exit upon failure unless it just so happened that their data ended when `event` was equal to 8. Multiple failures per subject would be possible because, other than running out of data, subjects would be removed only when `event==4`. Understand that subjects would be dropped from the analysis the first time `event==4`, even if that was before the first failure.

- `exit(event==4 8)`

Now subjects exit when `event` first equals either 4 or 8.

If you coded `failure(event==8) exit(event==4 8)`, you are saying that subjects exit upon failure and that they may exit before that when `event` equals 4.

- `exit(time lastdate)`

This is another example that allows for multiple failures of the same subject. In this case, each subject exits as of the earliest date recorded in variable `lastdate` regardless of the number of failures, if any, and of course exiting when they run out of data. `lastdate`, it is assumed, is recorded in units of *time* (not *t*, analysis time).

- `exit(time .)`

This also allows for multiple failures of the same subject. It is a variation of the above. It is used to indicate that each subject should exit only when they run out of data, regardless of the number of failures, if any.

- `exit(time d(20jan2000))`

This is just a variation on the previous example, and in this case, the exit-from-analysis date is a fixed date regardless of the number of failures. This would be a very odd thing to do.

- `exit(event==4 8 time d(20jan2000))`

This example is not so odd. Subjects exit from the analysis at the earliest date of (1) the earliest date at which `event` 4 or 8 occurs and (2) January 20, 2000.

Consider coding

```
failure(event==8) exit(event==4 8 time(d20jan2000))
```

You would be saying that subjects exit upon failure, that they exit before that if and when `event` 4 occurs, and that anyone still left around is removed from the analysis as of January 20, 2000, perhaps because that is the last date at which you have complete data.

You can check that you have specified `exit()` correctly by examining the variables `_d` and `_st` in your data; `_d` is 1 at failure and 0 otherwise, and `_st` is 1 when an observation is used and 0 otherwise:

```
. stset end, failure(event==8) exit(event==4) /*
>          */ origin(time begin) id(id) time0(begin)
(output omitted)
. list id begin end event x _t0 _t _d _st, noobs
```

id	begin	end	event	x	_t0	_t	_d	_st
12	20jan2000	21jan2000	3	3	0	1	0	1
12	21jan2000	25jan2000	8	3	1	5	1	1
12	25jan2000	30jan2000	4	3	5	10	0	1
12	30jan2000	31jan2000	8	3	.	.	.	0

```
. stset end, failure(event==8) exit(event==4 8) /*
>          */ origin(time begin) id(id) time0(begin)
(output omitted)
```

```
. list id begin end event x _t0 _t _d _st, noobs
```

id	begin	end	event	x	_t0	_t	_d	_st
12	20jan2000	21jan2000	3	3	0	1	0	1
12	21jan2000	25jan2000	8	3	1	5	1	1
12	25jan2000	30jan2000	4	3	.	.	.	0
12	30jan2000	31jan2000	8	3	.	.	.	0

Note that there is nothing stopping us from *stsetting* the data, looking, and then *stsetting* again if we do not like the result. Specifying

```
failure(event==8) exit(event==4 8)
```

would make more sense in the above example if, in addition to subject 12, we had another subject for which event 4 preceded event 8.

6.3.5 Specifying when subjects enter the analysis

When do subjects enter the analysis? We want subjects to enter at the onset of risk or, if they are not under observation at that point, after that, and that is Stata's default rule, but Stata has to assume that "under observation" corresponds to the presence of data.

Stata's default answer is that subjects enter at analysis time $t = 0$ (as specified by *origin()*), or if their earliest records in the data are after that, they enter then.

Some datasets, however, contain records reporting values before the subject was really under observation. The records are historical; they were added to the data after the subject enrolled, and had the subject failed during that early period, the subject would never have been around to enroll in our study. Consider the following data:

id	begin	end	event	x
27	.	11jan2000	2	.
27	11jan2000	15jan2000	10	3
27	15jan2000	21jan2000	8	3
27	21jan2000	30jan2000	9	3

In this example, pretend that *event*==2 is the onset of risk but *event*==10 is enrollment in our study. Subject 27 enrolled in our study on January 15 but came at risk before that—on January 11, a fact we determined when the subject enrolled in our study. Another subject might have the events reversed,

id	begin	end	event	x
27	.	11jan2000	10	.
27	11jan2000	15jan2000	2	3
27	15jan2000	21jan2000	8	3
27	21jan2000	30jan2000	9	3

and yet another might have the events coincident (indicated, perhaps, by `event==12`):

id	begin	end	event	x
29	.	11jan2000	12	.
29	11jan2000	21jan2000	8	3
29	21jan2000	30jan2000	9	3

Option `enter()` specifies when subjects enter. This option works the same as `exit()`—it is just that the meaning is the opposite. Some examples are

- `enter(event==2)`

Subjects enter when time `event` is 2 or $t = 0$, whichever is later. Specifying `enter(event==2)` does not necessarily cause all subjects to enter at the point `event` equals 2 if, in the data, `event` takes on the value 2 prior to $t = 0$ for some subjects. Those subjects would still enter into the analysis at $t = 0$.

- `enter(event== 2 12)`

Temporarily defines t' as the earliest t (time in analysis-time units) that `event==2` or `event==12` is observed for each subject. Subjects enter at the later of t' or $t = 0$. For example, if t' happens to correspond to a date earlier than that specified in `origin()`, then the onset of risk is just taken to be $t = 0$. The result is no different than if you typed

```
. gen ev_2_12 = (event==2) | (event==12)
. stset ..., ... enter(ev_2_12 == 1)
```

- `enter(time intvdate)`

`intvdate` contains the date of interview recorded in *time* (not analysis time) units. For each subject, `stset` finds the earliest time given in `intvdate` and then enforces the rule that the subject cannot enter before then.

- `enter(event==2 12 time intvdate)`

This is a typical compound specifier. For each set of records that share a common `id()` variable, `stset` finds the earliest time at which events 2 or 12 occurred. It then finds the earliest time of `intvdate` and takes the later of those two times. Subjects cannot enter before then.

Specifying `enter()` affects `_st`. It does not affect how analysis time is measured—only `origin()` and `scale()` do that. In the example below, `event==2` is the onset of risk, `event==10` is enrollment in our study, and `event==12` is simultaneous enrollment and onset of risk. Remember, instantaneous variables such as `event` are relevant at the end of the time span, which is `end` in these data:

(Continued on next page)

```
. use http://www.stata-press.com/data/cgg/id27_29
. list, noobs sepby(id)
```

id	begin	end	event	x
27	.	11jan2000	2	.
27	11jan2000	15jan2000	10	3
27	15jan2000	21jan2000	8	3
27	21jan2000	30jan2000	9	3
28	.	11jan2000	10	.
28	11jan2000	15jan2000	2	3
28	15jan2000	21jan2000	8	3
28	21jan2000	30jan2000	9	3
29	.	11jan2000	12	.
29	11jan2000	21jan2000	8	3
29	21jan2000	30jan2000	9	3

```
. stset end, origin(event==2 12) enter(event==10 12) /*
>      */ failure(event==9) time0(begin) id(id)
(output omitted)
. list id begin end event x _t0 _t _d _st, noobs sepby(id)
```

id	begin	end	event	x	_t0	_t	_d	_st
27	.	11jan2000	2	0
27	11jan2000	15jan2000	10	3	.	.	.	0
27	15jan2000	21jan2000	8	3	4	10	0	1
27	21jan2000	30jan2000	9	3	10	19	1	1
28	.	11jan2000	10	0
28	11jan2000	15jan2000	2	3	.	.	.	0
28	15jan2000	21jan2000	8	3	0	6	0	1
28	21jan2000	30jan2000	9	3	6	15	1	1
29	.	11jan2000	12	0
29	11jan2000	21jan2000	8	3	0	10	0	1
29	21jan2000	30jan2000	9	3	10	19	1	1

In studying these results, look particularly at subject 28:

id	begin	end	event	x	_t0	_t	_d	_st
28	.	11jan2000	10	0
28	11jan2000	15jan2000	2	3	.	.	.	0
28	15jan2000	21jan2000	8	3	0	6	0	1
28	21jan2000	30jan2000	9	3	6	15	1	1

We specified `origin(event==2 12) enter(event==10 12)`, so the subject entered our study on January 11 (when `event==10`) but did not come at risk until January 15 (when `event==2`). So how is it that this subject's second record—11jan2000 to 15jan2000—has `_st==0` when the record occurred while under observation? Answer: because analysis time t was negative prior to January 15 (when `event` became 2), and subjects cannot be in the analysis prior to $t = 0$.

6.3.6 Specifying the subject-id variable

If there are multiple records per subject in your data, as there have been in many of our examples, you must specify a subject-id variable using the `id()` option to `stset`. We have been doing that all along but without explaining.

If you do not specify the `id()` option, each record is assumed to reflect a different subject. If you do specify `id(varname)`, subjects with equal values of the specified variable `varname` are assumed to be the same subject.

It never hurts to specify an id variable, even in single-record data, because for various reasons you may want to later create multiple records for each subject.

For multiple-record data, when you specify an id variable, Stata verifies that none of the records overlap:

```
. use http://www.stata-press.com/data/cgg/id101
. list, noobs
```

id	begin	end	event	x
101	20jan2000	21jan2000	3	3
101	21jan2000	26jan2000	8	3
101	25jan2000	30jan2000	4	3
101	30jan2000	31jan2000	8	3

```
. stset end, failure(event==9) origin(time begin) id(id) time0(begin)

      id: id
failure event: event == 9
obs. time interval: (begin, end]
exit on or before: failure
t for analysis: (time-origin)
origin: time begin
```

```
4 total obs.
1 overlapping records (end[_n-1]>begin)          PROBABLE ERROR
```

```
3 obs. remaining, representing
1 subject
0 failures in single failure-per-subject data
7 total analysis time at risk, at risk from t =      0
      earliest observed entry t =      0
      last observed exit t =     11
```

Notice the **PROBABLE ERROR** that `stset` flagged. What did `stset` do with these two records? It kept the earlier one and ignored the later one:

```
. list id begin end event x _t0 _t _d _st, noobs
```

id	begin	end	event	x	_t0	_t	_d	_st
101	20jan2000	21jan2000	3	3	0	1	0	1
101	21jan2000	26jan2000	8	3	1	6	0	1
101	25jan2000	30jan2000	4	3	.	.	.	0
101	30jan2000	31jan2000	8	3	10	11	0	1

stset will not complain when, rather than an overlap, there is a gap. Note the gap between the second and third records:

```
. use http://www.stata-press.com/data/cgg/id102
. list, noobs
```

id	begin	end	event	x
102	20jan2000	21jan2000	3	3
102	21jan2000	25jan2000	8	3
102	27jan2000	30jan2000	4	3
102	30jan2000	31jan2000	8	3

```
. stset end, failure(event==9) origin(time begin) id(id) time0(begin)
```

```
      id: id
failure event: event == 9
obs. time interval: (begin, end]
exit on or before: failure
t for analysis: (time-origin)
origin: time begin
```

```
4 total obs.
0 exclusions
```

```
4 obs. remaining, representing
1 subject
0 failures in single failure-per-subject data
9 total analysis time at risk, at risk from t = 0
   earliest observed entry t = 0
   last observed exit t = 11
```

```
. list id begin end event x _t0 _t _d _st, noobs
```

id	begin	end	event	x	_t0	_t	_d	_st
102	20jan2000	21jan2000	3	3	0	1	0	1
102	21jan2000	25jan2000	8	3	1	5	0	1
102	27jan2000	30jan2000	4	3	7	10	0	1
102	30jan2000	31jan2000	8	3	10	11	0	1

stset did not even mention the gap, since interval truncation is a valid ingredient of any statistical analysis we would perform with these data. Stata does, however, have the command `stdes`, which describes datasets that have been `stset` and, in particular, will tell you if you have any gaps. `stdes` is covered in more detail in Section 7.3.

6.3.7 Specifying the begin-of-span variable

Another option we have been using without explanation is `time0()`. The last `stset` command we illustrated was

```
. stset end, failure(event==9) origin(time begin) id(id) time0(begin)
```


and, in fact, the `time0()` option has appeared in most of our examples. `time0()` is how you specify the beginning of the span, and if you omit this option Stata will determine the beginning of the span for you. For this reason, you must specify `time0()` if you have time gaps in your data and you do not want `stset` to assume that you do not.

Rather than having the data

id	begin	end	event	x	exercise
12	20jan2000	21jan2000	3	3	1
12	21jan2000	25jan2000	8	3	0
12	25jan2000	30jan2000	4	3	1
12	30jan2000	31jan2000	8	3	1

which has no gaps, you might have the same data recorded as

id	enrolled	date	event	x	exercise
12	20jan2000	21jan2000	3	3	1
12		25jan2000	8	3	0
12		30jan2000	4	3	1
12		31jan2000	8	3	1

or even

id	date	event	x	exercise
12	20jan2000	1	.	.
12	21jan2000	3	3	1
12	25jan2000	8	3	0
12	30jan2000	4	3	1
12	31jan2000	8	3	1

In this last example, we added an extra record at the top. In any case, all these datasets report the same thing: this subject enrolled in the study on January 20, 2000, and we also have observations on January 21, January 25, January 30, and January 31.

We much prefer the first way we showed these data,

id	begin	end	event	x	exercise
12	20jan2000	21jan2000	3	3	1
12	21jan2000	25jan2000	8	3	0
12	25jan2000	30jan2000	4	3	1
12	30jan2000	31jan2000	8	3	1

since it makes it very clear that these are time-span records, and gaps are represented naturally. The interpretation of these records is

time span			enduring variables		
id	begin	end	event	x	exercise
12	20jan2000	21jan2000	3	3	1
12	21jan2000	25jan2000	8	3	0
12	25jan2000	30jan2000	4	3	1
12	30jan2000	31jan2000	8	3	1
end time			instantaneous		
			variable(s)		

The enduring variables are relevant over the entire span, and the instantaneous variables are relevant only at the instant of the end of the span. Instantaneous variables may or may not be relevant to any statistical models fit to these data but are relevant when *stsetting* the data. In general, event and failure variables are instantaneous, and variables that record characteristics are enduring.

The second way we showed of recording these data changes nothing; we are just omitting to mention the begin-of-span variable:

id	enrolled	date	event	x	exercise
12	20jan2000	21jan2000	3	3	1
12		25jan2000	8	3	0
12		30jan2000	4	3	1
12		31jan2000	8	3	1

Consider the second record here. Over what period is `exercise==0`? Between January 21 and January 25—not January 25 to January 30 (during which it is 1). The time span for a record is the period from the record before it to this record, except for the first record, in which case it is from `enrolled` in this dataset.

We can *stset* this dataset, and the only issue is setting analysis time appropriately:

- `stset date, id(id) origin(time enrolled) ...`
if the date `enrolled` corresponds to the onset of risk
- `stset date, id(id) origin(event==3) ...`
if the occurrence of `event==3` marks the onset of risk
- `stset date, id(id) origin(event==8) ...`
if the occurrence of `event==8` marks the onset of risk

Note that we may omit the `time0()` option, and when we do that, *stset* obtains the time span by comparing adjacent records and assuming no gaps:

```
. use http://www.stata-press.com/data/cgg/id12b
. stset date, failure(event==8) origin(time enrolled) id(id)
      id: id
      failure event: event == 8
obs. time interval: (date[_n-1], date]
exit on or before: failure
t for analysis: (time-origin)
origin: time enrolled
```

```
4 total obs.
2 obs. begin on or after (first) failure
```

```
2 obs. remaining, representing
1 subject
1 failure in single failure-per-subject data
5 total analysis time at risk, at risk from t = 0
      earliest observed entry t = 0
      last observed exit t = 5
```

```
. list id enrolled date event x exercise _t0 _t _d _st, noobs
```

id	enrolled	date	event	x	exercise	_t0	_t	_d	_st
12	20jan2000	21jan2000	3	3	1	0	1	0	1
12	.	25jan2000	8	3	0	1	5	1	1
12	.	30jan2000	4	3	1	.	.	.	0
12	.	31jan2000	8	3	1	.	.	.	0

The third way we showed that these data might be recorded is by adding an extra observation at the top:

id	date	event	x	exerszd
12	20jan2000	1	.	.
12	21jan2000	3	3	1
12	25jan2000	8	3	0
12	30jan2000	4	3	1
12	31jan2000	8	3	1

In this case, `event==1` marks enrollment. The best way to think about these data is

time span			enduring variables		
id	date	event	x	exerszd	
12	~infinity	20jan2000	1	.	.
12	20jan2000	21jan2000	3	3	1
12	21jan2000	25jan2000	8	3	0
12	25jan2000	30jan2000	4	3	1
12	31jan2000	31jan2000	8	3	1
end time			instantaneous variable(s)		

Records, just as before, are paired. Variable `date` records the end of a time span, and the beginning of a time span is obtained from the record before. For the first record, just pretend the beginning of the span is the beginning of time. Note that the enduring variables are missing over this span and that all we have is an `event`, which occurs at the end of each span. To `stset` this dataset, you would type

- `stset date, id(id) origin(event==1) ...`
if the date of enrollment corresponds to the onset of risk
- `stset date, id(id) origin(event==3) ...`
if the occurrence of `event==3` marks the onset of risk
- `stset date, id(id) origin(event==8) ...`
if the occurrence of `event==8` marks the onset of risk

6.3.8 Convenience options

The previous sections have covered all the important options—options that affect the definition of failure.

```
failure(): failure(varname[==numlist])
```

options that affect the definition of analysis time,

```
origin(): origin([varname==numlist] time exp | min)
```

```
scale(): scale(#)
```

options that affect when the subject is under observation,

```
enter(): enter([varname==numlist] time exp)
```

```
exit(): exit(failure | [varname==numlist] time exp)
```

and options that provide the details that Stata needs to know,

```
id(): id(varname)
```

```
time0(): time0(varname)
```

That leaves the convenience options: *if()*, *ever()*, *never()*, *before()*, and *after()*. Each of these options takes any valid Stata expression as an argument and simply provides a convenient way to select records.

- *if()*

if() is like the standard Stata syntax *if exp* except that it is to be preferred in cases where you do not specify the *time0()* option. When you have recorded in your data only the ending times of each span, Stata derives the span by pairing records and using the ending times. The standard *if exp* in Stata's syntax removes records from consideration, meaning that if you have

patno	mytime	x1	x2	event
3	7	20	5	14
3	9	22	5	23
3	11	21	5	29

and you type *stset mytime if x1!=22, ...*, it will appear to Stata as if the dataset is missing the middle record, and thus Stata will incorrectly determine that the last record spans the period 7 to 11.

If you type *stset mytime, if(x1!=22) ...*, Stata will correctly determine that the third record spans the period 9 to 11 because Stata will leave the middle record in while it determines the beginning and ending times, and only then will it remove the record from consideration.

The problem that *if()* solves only arises when you do not specify the *time0()* option because, with *time0()*, Stata knows exactly when each record begins and ends.

- *ever()*

The subject is eligible for inclusion in the analysis only if the stated expression is ever true, whether in the past—even before $t = 0$ —or in the future—even after the failure event or *exit()*.

- `never()`

The subject is eligible for inclusion in the analysis only if the stated expression is never true, whether in the past—even before $t = 0$ —or in the future—even after the failure event or `exit()`.

- `before()`

Only the subject's records that occur before the first time the expression is true are eligible for inclusion in the analysis. The expression is assumed to be relevant at the end of time spans (based on instantaneous variables).

- `after()`

Only the subject's records that occur after the first time the expression is true are eligible for inclusion in the analysis. The expression is assumed to be relevant at the end of time spans (based on instantaneous variables).

Be careful using `before()` and `after()` with time variables rather than event variables. There is no danger in using something like `before(event==7)` or `after(event==8)`. However, consider something such as

```
before(end==d(28jan2000))
```

with the following data:

id	begin	end	event	x	exercise
12	20jan2000	21jan2000	3	3	1
12	21jan2000	25jan2000	8	3	0
12	25jan2000	30jan2000	4	3	1
12	30jan2000	31jan2000	8	3	1

In this dataset, the variable `end` never takes on the value `28jan2000`. Specifying `before(end>=d(28jan2000))` does not solve the problem, either. That would select the first two observations but omit the span January 25 to January 28.

Coding things like `before(event==4)`, on the other hand, is perfectly safe.



7 After stset

After you have `stset` your data, there are five things you should do before starting your formal analysis:

1. Look at `stset`'s output.
2. List at least a little of your data, examining `_t0`, `_t`, `_d`, and `_st` to be sure that they appear as you want them.
3. Type `stdes`. The “des” part means describe, and `stdes` provides a brief description of your data so, if there are problems, perhaps you will spot them.
4. Type `stvary` if you have multiple-record (meaning multiple records per subject) data. It reports whether variables within subjects vary over time and on their pattern of missing values.
5. If you discover problems in Step 4, use `stfill` and `streset` to fix them.

7.1 Look at stset's output

In most of the examples in Chapter 6, we omitted the output from `stset` for the sake of brevity. For instance, in one of our examples, we showed

```
. use http://www.stata-press.com/data/cgg/id27_29
. list, noobs sepby(id)
```

id	begin	end	event	x
27	.	11jan2000	2	.
27	11jan2000	15jan2000	10	3
27	15jan2000	21jan2000	8	3
27	21jan2000	30jan2000	9	3
28	.	11jan2000	10	.
28	11jan2000	15jan2000	2	3
28	15jan2000	21jan2000	8	3
28	21jan2000	30jan2000	9	3
29	.	11jan2000	12	.
29	11jan2000	21jan2000	8	3
29	21jan2000	30jan2000	9	3

```
. stset end, origin(event==2 12) enter(event==10 12) /*
>          */ failure(event==9) time0(begin) id(id)
(output omitted)
```

Let us now examine what `stset` actually displayed:

```
. stset end, origin(event==2 12) enter(event==10 12) /*
>          */ failure(event==9) time0(begin) id(id)

      id:  id
failure event:  event == 9
obs. time interval:  (begin, end]
enter on or after:  event==10 12
exit on or before:  failure
t for analysis:  (time-origin)
      origin:  event==2 12
```

```
11 total obs.
3  entry time missing (begin>=.)          PROBABLE ERROR
1  obs. end on or before enter()
1  obs. end on or before origin()
```

```
6  obs. remaining, representing
3  subjects
3  failures in single failure-per-subject data
49 total analysis time at risk, at risk from t =      0
      earliest observed entry t =      0
      last observed exit t =      19
```

Notice the **PROBABLE ERROR** message? In our example, it was in fact not an error, but we did not want to explain that at the time. Now we will explain and note that most probable errors are in fact errors, and the source of probable errors is worth exploring.

There are two parts to `stset`'s output. The first simply repeats in a more readable format what you specified in the `stset` command, and fills in any defaults:

```
      id:  id
failure event:  event == 9
obs. time interval:  (begin, end]
enter on or after:  event==10 12
exit on or before:  failure
t for analysis:  (time-origin)
      origin:  event==2 12
```

This output is fairly self-explanatory. One subtle item of interest is `(begin, end]`, where Stata is reporting from where it obtained the time span for each record. For this, Stata uses the interval notation `(]`, which mathematically means an interval for which the left endpoint is excluded and the right endpoint is included. The interval starts just after `begin` and continues up to and including `end`, where `begin` and `end` are two variables in our data.

The notation has statistical meaning. Say that subject 1 fails at time $t = 5$ and subject 2 is right censored at the same time. When Stata's semiparametric analyses commands perform the individual binary-outcome analyses for each failure time, would

Stata include or exclude subject 2 from the analysis for $t = 5$? The answer is that Stata *would* include subject 2 because the duration is interpreted as being up to and including the time listed.

The second part of *stset*'s output summarizes the results of applying the above definitions to this particular dataset:

11	total obs.	
3	entry time missing (<i>begin</i> >=.)	PROBABLE ERROR
1	obs. end on or before <i>enter</i> ()	
1	obs. end on or before <i>origin</i> ()	

6	obs. remaining, representing	
3	subjects	
3	failures in single failure-per-subject data	
49	total analysis time at risk, at risk from t =	0
	earliest observed entry t =	0
	last observed exit t =	19

This second part splits into two pieces, separated by the solid horizontal line. The first piece contains some odd things you should know about, and the second piece reports the characteristics of the dataset that was just set.

Among the odd things you should know about, some may be flagged as probable errors. This is Stata's way of telling you that it thinks something is wrong and that you want to be sure that you understand why this occurred so that you can safely assert that Stata is wrong. The messages not flagged as probable errors are facts that Stata considers common enough. If pressed, you should be able to explain why each of those arose, too, but you can start out by assuming that there is a good reason and wait for the data to prove you wrong.

For probable errors, start with a presumption of guilt on your part; for the rest, start with the presumption of innocence.

In fact, what is being reported in the first piece of the output is a complete accounting for the records in your data. For this example, Stata reports that

1. We started with 11 records in the dataset.
2. We decided that 6 were relevant for this analysis given what you specified.
3. The 5 remaining records that we excluded break down this way:
 - a. Three were omitted because the variable *begin* contained missing values, and Stata finds this odd.
 - b. One was omitted because "*obs. end on or before enter*()", meaning the time span of the record was before the subject could enter into the analysis given the rule *enter(event==10 12)*, which you specified.

- c. One was omitted because “`obs. end on or before origin()`”, meaning the time span of the record was before $t = 0$ given the rule `origin(event==2 12)`, which you specified.

In our example, it was indeed the case that `begin` contained missing values, but that was okay because we never tried to use the enduring variables from that open-ended time span, and in fact, all the enduring variables contained missing. As a result, we noted the **PROBABLE ERROR** message and investigated to find that there was indeed no error.

The warnings that `stset` might issue include (in order from the least serious to the most serious)

<code>obs. end on or before enter()</code>	
<code>obs. end on or before origin()</code>	
<code>entry time missing</code>	PROBABLE ERROR
<code>entry on or after exit (etime>t)</code>	PROBABLE ERROR
<code>multiple records at same instant (t[_n-1]==t)</code>	PROBABLE ERROR
<code>overlapping records (t[_n-1]>entry time)</code>	PROBABLE ERROR
<code>ignored because patid missing</code>	
<code>weights invalid</code>	PROBABLE ERROR
<code>event time missing</code>	PROBABLE ERROR

7.2 List some of your data

As previously stated, `stset` does not change any existing data. All it does is define the new variables `_t0`, `_t`, `_d`, and `_st`, which incorporate the information contained in the data with what the user specifies in `stset`. As such, it is important to check these variables on at least a small part of your data to make sure that the results were as you intended.

- `_t0` and `_t` record the time spans in analysis time units.
- `_d` records the outcome (failure or censoring) at the end of each time span.
- `_st` records whether the observation is relevant to the current analysis.

These are the variables that matter in the sense that all of the other survival analysis commands (the `st` family) work with these variables rather than with your original variables.

This last fact means that if you change your data, you need to `stset` it again so that these variables can change along with the data. In fact, `stset` is smart enough to remember which syntax and options you had previously used, and thus all you would have to do is type `stset` without any arguments.

In many cases, you will have trimmed and cleaned your data to the point where all observations should be relevant to the analysis. Since `_st` is an indicator variable that marks inclusion into subsequent analyses, verify that `_st==1` in all observations. Either type `assert _st==1`, or type `summarize _st` and verify the mean of `_st` is one.

If you expect records to be excluded, on the other hand, look at a few of the records for which `_st==0` to verify that the right observations were excluded.

□ Technical Note

In survival models, the “response” is the triple (t_0, t, d) , where t_0 marks the beginning of a time span, t marks the end, and d indicates failure or censoring. As such, when we `stset` our data, all we are doing is generating the “response” variables based on the information in our data, and generating an indicator variable (`_st`) that determines whether our response makes sense given what we know about our data. Thus, by using `stset`, we guarantee that no matter what model we choose to fit, we are using the same response.

□

7.3 Use stdes

`stdes` presents a brief description of the dataset you have just set. Continuing our example from the previous section,

```
. stdes
      failure _d: event == 9
      analysis time _t: (end-origin)
              origin: event==2 12
      enter on or after: event==10 12
                  id: id
```

Category	total	per subject			
		mean	min	median	max
no. of subjects	3				
no. of records	6	2	2	2	2
(first) entry time		1.333333	0	0	4
(final) exit time		17.66667	15	19	19
subjects with gap	0				
time on gap if gap	0
time at risk	49	16.33333	15	15	19
failures	3	1	1	1	1

Of particular interest is the line beginning “subjects with gap”. Recall that `stset` will flag overlaps but will not even mention gaps, since gaps are not errors. `stdes`, on the other hand, will detect gaps and will even provide summary statistics on the lengths of the gaps.

The remainder of the output can be thought of as that from a specialized `summarize` command for survival data.

□ Technical Note

Notice the first part of the output from *stdes*, the part that reads

```
. stdes
      failure _d:  event == 9
      analysis time _t:  (end-origin)
                origin:  event==2 12
      enter on or after:  event==10 12
                id:  id
```

That portion of the output is not really being produced by *stdes* but by the *st* system itself. It will appear in the output of every *st* command—the idea being to remind you how you have *stset* your data.

You may get tired of seeing that reminder and, if so, type

```
. stset, noshow
```

Later, if you want to get it back again, type

```
. stset, show
```

□

7.4 Use *stvary*

Here is the result of running *stvary* on our current analysis:

```
. stvary
      failure _d:  event == 9
      analysis time _t:  (end-origin)
                origin:  event==2 12
      enter on or after:  event==10 12
                id:  id

      subjects for whom the variable is
```

variable	constant	varying	never missing	always missing	sometimes missing
x	3	0	3	0	0

With a larger dataset with more covariates, we might see something like

```
. stvary
      failure _d:  event == 9
      analysis time _t:  (end-origin)
                origin:  event==2 12
      enter on or after:  event==10 12
                id:  id

      subjects for whom the variable is
```

variable	constant	varying	never missing	always missing	sometimes missing
sex	500	0	20	0	480
bp	80	400	100	20	380
x	497	3	65	0	435

You want to become practiced at reading this output because it will uncover problems in your data.

Let us start with the variable `sex`. Within subject, it never varies—the varying column is 0. That is good—no one changes their sex in the middle of the study. Also good is that there is not one subject in the data for which the variable is “always missing”. For 480 subjects, however, the value of `sex` is sometimes missing, and only for 20 is it recorded in every observation. This is a problem that we will want to fix. In the 480 “sometimes missing” subjects, we are seeing something like

```
id      sex
42      .
42      1
42      .
```

For 480 subjects, the variable `sex` is at least filled in once and is at least missing once. This would be easy enough to fix,

```
. sort id end
. quietly by id: replace sex = sex[_n-1] if missing(sex)
. quietly by id: replace sex = sex[_N]
```

but for now we will focus instead on just making a list of the problems. In Section 7.5, we will see that there is actually an easier way to fix these types of problems.

Variable `bp` (blood pressure) is shown to be constant for 80 subjects and varying for 400. Varying means that it takes on different, nonmissing values at different times. For 20 subjects, according to `stvary`, the variable is always missing.

Some examples of how `stvary` categorizes constant, varying, and always missing are

id	bp		id	bp	
56	.	<- always missing	60	.	<- constant
56	.		60	180	
56	.		60	.	
57	180	<- constant	61	180	<- varying
57	180		61	190	
57	180		61	200	
58	180	<- constant	62	180	<- varying
58	.		62	190	
58	180		62	180	
59	180	<- constant	63	180	<- varying
59	.		63	.	
59	.		63	190	

Does `bp` have problems? It does if we want to use it as an explanatory variable in our models because any observations for which `bp` is missing would be dropped from the analysis. Perhaps that is necessary given that we are uncertain as to the value of `bp` when it is missing. On the other hand, we may be willing to fill in `bp` values from the last time they were known so that

id	bp	bp	<- becomes
58	180	180	
58	.	180	
58	180	180	
59	180	180	
59	.	180	
59	.	180	
60	.	.	
60	180	180	
60	.	180	
62	180	180	
62	.	180	
62	190	190	

which is easy enough to do for cases where bp is not always missing,

```
. sort id end
. quietly by id: replace bp = bp[_n-1] if missing(bp)
```

and in fact, we show an automatic way to do this in Section 7.5.

Returning to the **stvary** output, variable **x** in our data also looks problematic, since it is constant for 497 subjects and varies for only 3. This looks suspicious.

1. Should **x** really be constant for everyone and thus the 3 subjects for which it varies represent coding errors?
2. We recall from above that **stvary** has a strange concept of constant—if we observe the variable just once and all the other values are missing, then **stvary** categorizes that result as constant. Ergo, is it possible that all 497 “constant” subjects in our data have **x** observed just once? Some 435 could be like that, because 435 are “sometimes missing”. We should look, and if we find that to be true, we wonder about the reasonableness of filling in subsequent values.

7.5 Perhaps use stfill

stfill is an alternative to doing things like

```
. sort id end
. quietly by id: replace sex = sex[_n-1] if missing(sex)
. quietly by id: replace sex = sex[_N]
```

and

```
. sort id end
. quietly by id: replace bp = bp[_n-1] if missing(bp)
```

The basic syntax of *stfill* is

```
stfill varlist, { baseline|forward }
```

The *baseline* option will fill in all the records with the value first observed for each subject, even if the current value is nonmissing. This has the effect of making the variables in *varlist* constant within each subject. The *forward* option, alternatively, will fill in missing values by carrying forward prior values of variables.

For example, if you type

```
. stfill sex bp, forward
```

stfill will fix the variables *sex* and *bp* from our previous example. Almost. *stfill* will not backfill, which is to say, fill in earlier times with values from later times, and Stata's reason for refusing to is that backfilling is harder to justify.

There is another issue regarding the use of *stfill*, and this issue applies to other *st* data-management commands. When you *stset* the data, the result may be that only a subset of the available records is marked as valid for future analysis. For example, suppose you have the following recorded in your dataset,

<i>id</i>	<i>date0</i>	<i>date1</i>	<i>event</i>	...
71	01jan2000	04jan2000	2	
71	04jan2000	07jan1999	3	
71	07jan2000	13jan2000	6	
71	13jan2000	18jan2000	6	
71	18feb2000	24feb2000	9	
71	24jan2000	28jan2000	3	
71	28jan2000	31jan2000	3	

and you *stset* the data with

```
. stset date1, id(id) failure(event==9) origin(event==3)
```

which, for this subject, results in

<i>id</i>	<i>date0</i>	<i>date1</i>	<i>event</i>	<i>_t0</i>	<i>_t</i>	<i>_d</i>	<i>_st</i>
71	01jan2000	04jan2000	2	.	.	.	0
71	04jan2000	07jan1999	3	.	.	.	0
71	07jan2000	13jan2000	6	0	6	0	1
71	13jan2000	18jan2000	6	6	11	0	1
71	18feb2000	24feb2000	9	11	17	1	1
71	24jan2000	28jan2000	3	.	.	.	0
71	28jan2000	31jan2000	3	.	.	.	0

Thus, we have 7 records for this subject, but for our analysis we are only using the middle three.

We call this whole collection of data on the patient a *history*. We refer to the records before the onset of risk as the *past history*, the records being used as the *current history*, and the records after failure or censoring as the *future history*.

Now, if we are going to fill values forward, it would be a shame not to use the information we have in the past. If we are going to go to the effort of cleaning up data,

it would be a shame not to clean all of it, including the future. However, all of the *st* commands—including the *st* data-management commands—restrict themselves to the current history. They pretend the other data do not exist.

There exists a trick, however, to make Stata recognize the other records for purposes of data management. If you type

```
. streset, past
```

Stata will temporarily set the dataset so that it includes the *past* and the current history, and when you later type

```
. streset
```

Stata will make things just as they were, with only the current history set. If you type

```
. streset, future
```

Stata will temporarily set the dataset so that it includes the current and the future history, and of course, typing *streset* by itself will restore things to be as they were. If you type

```
. streset, past future
```

Stata will temporarily set the dataset to include the full history.

What do these commands really do? They concoct an absurd definition of analysis time and failure so that the observations temporarily appear as if they are in the analysis. The analysis time and failure definitions Stata concocts are not appropriate for analysis, but that does not matter because Stata knows when it is using these artificial definitions and will not let you do anything inappropriate in terms of using the *st* analysis commands.

So, returning to our example, the best way to fix *sex* and *bp* is

```
. stfill sex, baseline
. streset, past future
. stfill bp, forward
. streset
```

Note that we *stfill sex, baseline* before including the past history, just to make sure that the value used coincides with the onset of risk.

7.6 Example: Hip fracture data

Let us pretend that a study was performed to quantify the benefit of a new inflatable device to protect elderly persons from hip fractures resulting from falls. The device is worn around the hips at all times. It is hypothesized that the device will reduce the incidence of hip fractures in this population. Forty-eight (48) women over the age of 60, without previous histories of hip trauma, were recruited for this study. Of these 48 women, 28 were randomly given the device and instructed on how to wear it. The

remaining 20 women were not provided with the device and were used as study controls. All 48 women were followed closely, and blood calcium levels were drawn approximately every 5 months. The time to hip fracture or censoring was recorded in months. It was decided at study onset that, if at any time during the follow-up period, a woman was hospitalized for any reason, she would not be considered at risk of falling and fracturing her hip. This creates gaps in the data.

The dataset below is real, so feel free to use it. The data, however, are fictional.

```
. use http://www.stata-press.com/data/cgg/hip
(hip fracture study)
. describe
Contains data from http://www.stata-press.com/data/cgg/hip.dta
  obs:      106                hip fracture study
  vars:      7                30 Jan 2002 11:58
  size:      1,484 (99.9% of memory free)
```

variable name	storage type	display format	value label	variable label
id	byte	%4.0g		patient id
time0	byte	%5.0g		begin of span
time1	byte	%5.0g		end of span
fracture	byte	%8.0g		fracture event
protect	byte	%8.0g		wears device
age	byte	%4.0g		age at enrollment
calcium	float	%8.0g		blood calcium level

Sorted by:

```
. summarize
```

Variable	Obs	Mean	Std. Dev.	Min	Max
id	106	28.21698	13.09599	1	48
time0	106	4.792453	5.631065	0	15
time1	106	11.5283	8.481024	1	39
fracture	106	.2924528	.4570502	0	1
protect	48	.5833333	.4982238	0	1
age	48	70.875	5.659205	62	82
calcium	106	10.10849	1.407355	7.25	12.32

```
. sort id time0
```

```
. by id: list time0-calcium
```

```
-> id = 1
```

	time0	time1	fracture	protect	age	calcium
1.	0	1	1	0	76	9.35

-> id = 2

	time0	time1	fracture	protect	age	calcium
1.	0	1	1	0	80	7.8

(output omitted)

-> id = 17

	time0	time1	fracture	protect	age	calcium
1.	0	8	0	0	66	11.48
2.	8	15	1	.	.	10.79

-> id = 18

	time0	time1	fracture	protect	age	calcium
1.	0	5	0	0	64	11.58
2.	15	17	1	.	.	11.59

(output omitted)

-> id = 47

	time0	time1	fracture	protect	age	calcium
1.	0	5	0	1	63	12.18
2.	5	15	0	.	.	11.64
3.	15	35	0	.	.	11.79

-> id = 48

	time0	time1	fracture	protect	age	calcium
1.	0	5	0	1	67	11.21
2.	5	15	0	.	.	11.43
3.	15	39	0	.	.	11.29

In our data, time is already recorded in analysis time units, which just means we will not have to bother with the `origin()` option when we type `stset`.

Our data do, however, have multiple observations per subject to accommodate the time-varying covariate `calcium`, and we will assume that the value of this variable is fixed over the interval spanned by each record.

Our `age` variable records the age of each participant at the time of enrollment in the study. Glancing at our data, you will notice that `age` appears to be coded only in the first record for each subject. All the records are like that. If we later copy this

value of age down (i.e., propagate age values from past to future observations), we will be treating age as fixed.

In any case, the `stset` command for this dataset is

```
. stset time1, id(id) time0(time0) failure(fracture)
      id: id
      failure event: fracture != 0 & fracture < .
obs. time interval: (time0, time1]
exit on or before: failure
```

```
106 total obs.
  0 exclusions
```

```
106 obs. remaining, representing
 48 subjects
 31 failures in single failure-per-subject data
714 total analysis time at risk, at risk from t =      0
      earliest observed entry t =      0
      last observed exit t =     39
```

Let us now go through the data verification process described in this chapter. We begin by looking at the above output, then examine `_t0`, `_t`, `_d`, and `_st`, confirm that `stde` does not reveal any surprises, confirm that `stvary` makes a similarly unsurprising report, and finally use `stfill` to fix any problems uncovered by `stvary`.

First, we look at `_t0`, `_t`, `_d`, and `_st` in the beginning, middle, and end of the data.

```
. list id time0 time1 frac _t0 _t _d _st if id<=3
```

	id	time0	time1	fracture	_t0	_t	_d	_st
1.	1	0	1	1	0	1	1	1
2.	2	0	1	1	0	1	1	1
3.	3	0	2	1	0	2	1	1

```
. list id time0 time1 frac _t0 _t _d _st if 16<=id & id<=18, sepby(id)
```

	id	time0	time1	fracture	_t0	_t	_d	_st
23.	16	0	5	0	0	5	0	1
24.	16	5	12	1	5	12	1	1
25.	17	0	8	0	0	8	0	1
26.	17	8	15	1	8	15	1	1
27.	18	0	5	0	0	5	0	1
28.	18	15	17	1	15	17	1	1

```
. list id time0 time1 frac _t0 _t _d _st if id>=47, sepby(id)
```

	id	time0	time1	fracture	_t0	_t	_d	_st
101.	47	0	5	0	0	5	0	1
102.	47	5	15	0	5	15	0	1
103.	47	15	35	0	15	35	0	1
104.	48	0	5	0	0	5	0	1
105.	48	5	15	0	5	15	0	1
106.	48	15	39	0	15	39	0	1

Okay, that looks good to us. Let's see what *stdes* has to say:

```
. stdes
```

```
      failure _d: fracture
analysis time _t: time1
      id: id
```

Category	total	per subject			
		mean	min	median	max
no. of subjects	48				
no. of records	106	2.208333	1	2	3
(first) entry time		0	0	0	0
(final) exit time		15.5	1	12.5	39
subjects with gap	3				
time on gap if gap	30	10	10	10	10
time at risk	714	14.875	1	11.5	39
failures	31	.6458333	0	1	1

Starting with the first two lines, *stdes* reports that there are 48 subjects in our data and that there are a total of 106 records, and that the average number of records per subject is just over two with three being the maximum number of records for any one subject. From this, we see that the *st* system correctly recognizes that there are multiple records per subject. In cases where there is only one observation per subject, the reported totals in the first and second line would be equal, and the mean, max, and min number of records per subject would all be equal to one.

In lines 3 and 4, *stdes* reports that everyone entered at time 0—there is no delayed entry—and that the average exit time was 15.5 months, with a minimum of one and a maximum of 39. Be careful interpreting this reported average exit time. Note that this is just the average of the follow-up times; it is *not* the average survival time because some of our subjects are censored. When there are no censored observations, the average exit time reported by *stdes* does equal the average survival time.

In lines 5 and 6, *stdes* reports that there are three subjects with gaps each 10 months long. This is a strange finding. In most datasets with time gaps, the gaps vary in length. We were immediately suspicious of these results and wanted to verify them. One way to identify these three subjects is to make use of the fact that, when there

are no gaps between consecutive observations for a subject, the ending time of the first record equals the beginning time of the next record. So, we will sort the data by subject and time, and then we will generate a dummy variable `gap==1` for those observations with gaps. Then, we can list the observations.

```
. sort id _t0 _t
. quietly by id: gen gap=1 if _t0 != _t[_n-1] & _n>1
. list id if gap==1
```

	id
28.	18
55.	30
63.	33

```
. list id time0 time1 _t0 _t if id==18 | id==30 | id==33, sepby(id)
```

	id	time0	time1	_t0	_t
27.	18	0	5	0	5
28.	18	15	17	15	17
54.	30	0	5	0	5
55.	30	15	19	15	19
62.	33	0	5	0	5
63.	33	15	23	15	23

All appears well; each of these records had a gap lasting 10 months, yet perhaps we still might want to check that the data were entered correctly.

Returning to the `stdes` output, on line 7 we observe that subjects were at risk of failure for a total of 714 months. This is simply the sum of the time spanned by the records, calculated by `stdes` by calculating the length of the time interval represented by each record (`_t0`, `_t`) and then summing these lengths conditional on `_st==1`.

Finally, in line 8, `stdes` reports that there were 31 failures, or 31 hip fractures, in our data. Note that the maximum number of failures is 1, indicating that we have single failure-per-subject data, and that the minimum number of failures is 0, indicating the presence of censored observations. Of course, we can also see that there are censored observations when we compare the total number of failures and the total number of observations on line 2. As we may expect, for a dataset with multiple failures-per-observation, `stdes` will provide in line 8 summary statistics indicating the existence of multiple failures-per-subject data.

(Continued on next page)

So, all looks fine based on stdes. Next, let's try stvary:

```
. capture drop gap
. stvary
      failure _d: fracture
      analysis time _t: time1
      id: id

      subjects for whom the variable is
```

variable	constant	varying	never missing	always missing	sometimes missing
protect	48	0	8	0	40
age	48	0	8	0	40
calcium	8	40	48	0	0

By default, stvary reports on all variables in the dataset, omitting the variables used or created by stset. (stvary optionally allows variable lists, so we can specify those variables that we wish to examine.)

The variable **protect** records if the subject is in the experimental or control group, **age** records the age at enrollment in the study, and **calcium** records the subject's blood calcium concentration. Recall that this last characteristic was examined approximately every five months, so the variable varies over time.

Looking at stvary's output, what catches our eye is the large number of "sometimes missings". Well, we already knew that **protect** and **age** did not have the values filled in on subsequent records and that we would have to fix that. Let us now follow our own advice and use **streset** and **stfill** to fix this problem. In this case, **streset** is unnecessary since there are no observations for which **_st==0**, but it never hurts to be cautious:

```
. streset, past future
      (output omitted)
. stfill age protect, forward
      failure _d: fracture
      analysis time _t: (time1-origin)
      origin: min
      exit on or before: time .
      id: id

replace missing values with previously observed values:
      age: 58 real changes made
      protect: 58 real changes made
. streset
      (output omitted)
```

Now, stvary reports

```
. stvary
      failure _d: fracture
      analysis time _t: time1
      id: id
      subjects for whom the variable is
```

variable	constant	varying	never missing	always missing	sometimes missing
protect	48	0	48	0	0
age	48	0	48	0	0
calcium	8	40	48	0	0

Satisfied, we can save these data as hip2.dta.

```
. save hip2
file hip2.dta saved
```

Note that when you `stset` a dataset and `save` it, Stata remembers how the data were set the next time you use the data.



8 Nonparametric analysis

The previous two chapters served as a tutorial on `stset`. Once you `stset` your data, you can literally use any `st` survival command, and the nice thing is that you do not have to continually restate the definitions of analysis time, failure, and rules for inclusion.

As previously discussed in Chapter 1, the analysis of survival data can take one of three forms: nonparametric, semiparametric, and parametric, all depending on what we are willing to assume about the form of the survivor function and about how the survival experience is affected by covariates.

Nonparametric analysis follows the philosophy of letting the dataset speak for itself and makes no assumption about the functional form of the the survivor function (and thus no assumption about the hazard, cumulative hazard, etc.), and the effects of covariates are not modeled either—the comparison of the survival experience is done at a qualitative level across the values of the covariates.

Most of Stata's nonparametric survival analysis is performed via the `sts` command, which calculates estimates, saves estimates as data, draws graphs, and performs tests, among other things; see [ST] `sts`.

8.1 Inadequacies of standard univariate methods

Before we proceed, however, it is important to discuss briefly the reasons that the typical preliminary data analysis tools do not translate well into the survival analysis paradigm. For example, the most basic of analyses would be one that analyzed the mean time to failure or median time to failure.

Let us use the hip-replacement dataset, which we `stset` at the end of Chapter 7:

(Continued on next page)

```
. use http://www.stata-press.com/data/cgg/hip2
(hip fracture study)
. list id _t0 _t fracture protect age calcium if 20<=id & id<=22, sepby(id)
```

	id	_t0	_t	fracture	protect	age	calcium
32.	20	0	5	0	0	67	11.19
33.	20	5	15	0	0	67	10.68
34.	20	15	23	1	0	67	10.46
35.	21	0	5	0	1	82	8.97
36.	21	5	6	1	1	82	7.25
37.	22	0	5	0	1	80	7.98
38.	22	5	6	0	1	80	9.65

Putting aside for now the possible effects of the covariates, if we were interested in estimating the population mean time to failure, we might be tempted to use the standard tools such as

```
. ci _t
```

Variable	Obs	Mean	Std. Err.	[95% Conf. Interval]	
_t	106	11.5283	.8237498	9.894958	13.16165

We might quickly realize that this is not what we want because there exist multiple records for each individual. Being smarter, we could just consider those values of `_t` corresponding to the last record for each individual,

```
. sort id _t
. quietly by id: gen last = _n==_N
. ci _t if last
```

Variable	Obs	Mean	Std. Err.	[95% Conf. Interval]	
_t	48	15.5	1.480368	12.52188	18.47812

and we now have a mean based on 48 observations (one for each subject). This will not serve, however, because `_t` does not always correspond to failure time—some times in our data are censored, meaning that the failure time in these cases is known only to be greater than `_t`. As such, the estimate of the mean is biased downwards.

Dropping the censored observations and redoing the analysis will not help. Consider an extreme case of a dataset with just one censored observation and assume the observation is censored at time 0.1, long before the first failure. For all you know, had that subject not been censored, their failure might have occurred long after the last failure in the data and so had a large effect on the mean. Wherever the censored observation is located in the data, we can repeat that argument, and so, in the presence of censoring, obtaining estimates of the mean survival time calculated in the standard way is simply not possible.

Estimates of the median survival time are similarly not possible to obtain using standard, nonsurvival tools. The “standard” way of calculating the median is to order the observations and report the “middle” one as the median. In the presence of censoring, that ordering is impossible to ascertain. (The modern way of calculating the median is to turn to the calculation of survival probabilities and find the point at which the survival probability is 0.5.)

Thus, even the most simple analysis—never mind the more complicated regression models—will break down when applied to survival data. In addition, there are even more issues related to survival data—truncation, for example—that would only further complicate the estimation.

Instead, survival analysis is a field of its own. Given the nature of the role that time plays in the analysis, much focus is given to the functions that characterize the distribution of the survival time: the hazard function, the cumulative hazard function, and the survivor function being the most common ways to describe the distribution. Much of survival analysis is concerned with the estimation of and inference for these functions of time.

8.2 The Kaplan–Meier estimator

8.2.1 Calculation

The estimator of Kaplan and Meier (1958) is a nonparametric estimate of the survivor function $S(t)$, the probability of survival past time t (or equivalently, the probability of failing after t). For a dataset with observed failure times, t_1, \dots, t_k , where k is the number of distinct failure times observed in the data, the Kaplan–Meier estimate (also known as the *product limit* estimate of $S(t)$) at any time t is given by

$$\hat{S}(t) = \prod_{j|t_j \leq t} \left(\frac{n_j - d_j}{n_j} \right) \quad (8.1)$$

where n_j is the number of individuals at risk at time t_j and d_j is the number of failures at time t_j . The product is over all observed failure times less than or equal to t .

How does this estimator work? Consider the hypothetical dataset of subjects given in the usual format,

id	t	failed
1	2	1
2	4	1
3	4	1
4	5	0
5	7	1
6	8	0

and form a table that summarizes what happens at each time in our data (whether a failure time or a censored time):

At time	No. at risk	No. failed	No. censored
2	6	1	0
4	5	2	0
5	3	0	1
7	2	1	0
8	1	0	1

At time $t = 2$, the earliest time in our data, all 6 subjects were at risk, but at that instant, 1 failed ($\text{id}=1$). At the next time, $t = 4$, 5 subjects were at risk, and at that instant, 2 of the 5 subjects failed. At time $t = 5$, 3 subjects were left, and no one failed, but one subject was censored, leaving us with 2 subjects at time $t = 7$, of which one failed. Finally, at time $t = 8$, we had 1 subject left at risk, and this subject was censored at that time.

Now we ask the following:

- What is the probability of survival beyond time $t = 2$, the earliest time in our data? Since 5 out of the six subjects survived beyond this point, the estimate is $5/6$.
- What is the probability of survival beyond time $t = 4$, given survival right up to $t = 4$? Since we had 5 subjects at risk at time $t = 4$, of which 2 failed, we estimate this probability to be $3/5$.
- What is the probability of survival beyond time $t = 5$, given survival right up to $t = 5$? Since 3 subjects were at risk, and no one failed, the probability estimate is $3/3 = 1$.

and so on. We can now add these individual component probabilities (calling them p) to our table:

At time	No. at risk	No. failed	No. censored	p
2	6	1	0	$5/6$
4	5	2	0	$3/5$
5	3	0	1	1
7	2	1	0	$1/2$
8	1	0	1	1

- The first value of p , $5/6$, is the probability of survival beyond $t = 2$.
- The second value, $3/5$, is the (conditional) probability of survival beyond $t = 4$, given survival up until $t = 4$, which in these data is the same as survival beyond time $t = 2$. Thus, unconditionally, the probability of survival beyond $t = 4$ is $(5/6)(3/5) = 1/2$.

- The third value, 1, is the conditional probability of survival beyond $t = 5$, given survival up until $t = 5$, which in these data is the same as survival beyond time $t = 4$. Unconditionally, the probability of survival beyond $t = 5$ is thus equal to $(1/2)(1) = 1/2$.

Thus, the Kaplan–Meier estimate is the running product of the values of p that we have previously calculated, and we can add it to our table.

At time	No. at risk	No. failed	No. censored	p	$\hat{S}(t)$
2	6	1	0	5/6	5/6
4	5	2	0	3/5	1/2
5	3	0	1	1	1/2
7	2	1	0	1/2	1/4
8	1	0	1	1	1/4

Note that since the Kaplan–Meier estimate in (8.1) only operates on observed failure times (and not at censoring times), the net effect is simply to ignore the cases where $p = 1$ in our calculating our product, which changes nothing.

In Stata, the Kaplan–Meier estimate is obtained using the `sts list` command, which gives a table similar to the one we constructed:

```
. input id time failed
      id      time      failed
1. 1 2 1
2. 2 4 1
3. 3 4 1
4. 4 5 0
5. 5 7 1
6. 6 8 0
7. end
. stset time, fail(failed)
(output omitted)
. sts list
```

Time	Beg. Total	Fail	Net Lost	Survivor Function	Std. Error	[95% Conf. Int.]	
2	6	1	0	0.8333	0.1521	0.2731	0.9747
4	5	2	0	0.5000	0.2041	0.1109	0.8037
5	3	0	1	0.5000	0.2041	0.1109	0.8037
7	2	1	0	0.2500	0.2041	0.0123	0.6459
8	1	0	1	0.2500	0.2041	0.0123	0.6459

The column “Beg. Total” is what we called “No. at risk” in our table; the column “Fail” is “No. failed”; and the column “Net lost” is related to our “No. censored” column but is modified to handle delayed entry (see Section 8.2.3).

The standard error reported for the Kaplan-Meier estimate is that given by Greenwood's (1926) formula:

$$\widehat{\text{Var}}\{\hat{S}(t)\} = \hat{S}^2(t) \sum_{j|t_j \leq t} \frac{d_j}{n_j(n_j - d_j)}$$

These standard errors, however, are not used for confidence intervals. Instead, the asymptotic variance of $\ln\{-\ln \hat{S}(t)\}$,

$$\hat{\sigma}^2(t) = \frac{\sum \frac{d_j}{n_j(n_j - d_j)}}{\left\{ \sum \ln \left(\frac{n_j - d_j}{d_j} \right) \right\}^2}$$

is used, where the sums are calculated over $j|t_j \leq t$ (Kalbfleisch and Prentice 2002, 18). The confidence bounds are then calculated as $\hat{S}(t)^{\exp\{\pm z_{\alpha/2} \hat{\sigma}(t)\}}$, where $z_{\alpha/2}$ is the $(1 - \alpha/2)$ quantile of the normal distribution.

8.2.2 Censoring

When censoring occurs at some time other than an observed failure time, the effect is simply that the censored subjects are dropped from the “No. at risk” total without processing the subject as having failed. However, when some subjects are censored at the same time that others fail, we need to be a bit careful about how we order the censorings and failures. When we went through the calculations of the Kaplan-Meier estimate in Section 8.2.1, we did so without explaining this point, yet be assured that we were following some convention.

The Stata convention for handling a censoring that happens at the same time as a failure is to assume that the failure occurred before the censoring, and in fact, all of Stata's `st` commands follow this rule. Recall that in Chapter 7 that we defined a time span based on the `stset` variables `_t0` and `_t` to be the interval $(t_0, t]$, which is open at the left endpoint and closed at the right endpoint. Therefore, if we apply this definition of a time span, then any record that is shown to be censored at the end of this span can be thought of as instead being censored at some time $t + \epsilon$ for an arbitrarily small ϵ . In other words, the subject can fail at time t , but if the failure is censored, Stata assumes that the censoring took place just a little bit later, and thus, failures occur before censorings.

This is how Stata handles this issue, but there is nothing wrong with the convention that handles censorings as occurring before failures when they appear to happen concurrently. One can force Stata to look at things this way by subtracting a small number from the time variable in your data for those records that are censored, and most of the time the number may be chosen small enough as to not otherwise affect the analysis.

□ Technical Note

If you choose to force Stata to treat censorings as occurring before failures, be sure to modify the time variable in your data, and not the `_t` variable that `stset` has created. In general, manually changing the values of the `stset` variables `_t0`, `_t`, `_d`, and `_st` is dangerous, since these variables have relations to your variables, and some of the data-management `st` commands exploit that relationship.

Thus, instead of something such as

```
. replace _t = _t - 0.0001 if _d == 0
```

use

```
. replace time = time - 0.0001 if failed == 0
. stset time, failure(failed)
```

□

8.2.3 Left truncation (delayed entry)

Recall that left truncation refers to subjects who do not come under observation until after they are at risk. In other words, by the time you begin observing this subject, they have already survived for some time, and you are only observing them because they did not fail during that time.

At one level, such observations cause no problems with the Kaplan–Meier calculation. In (8.1), n_j is the number of subjects at risk (eligible to fail), and this number simply needs to take into account that subjects are not at risk of failing until they come under observation. When they enter, we simply increase n_j to reflect this fact.

For example, if you have the following data (note that subject 6 enters at time $t_0 = 4$ and is censored at time $t = 7$),

id	t0	t1	failed
1	0	2	1
2	0	4	1
3	0	4	1
4	0	5	0
5	0	7	1
6	4	7	0
7	0	8	0

then the “risk-group” table is

At time	No. at risk	No. failed	No. censored	No. added
2	6	1	0	0
4	5	2	0	1
5	4	0	1	0
7	3	1	1	0
8	1	0	1	0

and now it is just a matter of making the Kaplan–Meier calculations based on how many are in the “No. at risk” and “No. failed” columns. We will let Stata do the work:

```
. input id time0 time1 failed
      id    time0    time1    failed
1.  1      0      2      1
2.  2      0      4      1
3.  3      0      4      1
4.  4      0      5      0
5.  5      0      7      1
6.  6      4      7      0
7.  7      0      8      0
8. end
```

```
. stset time1, fail(failed) time0(time0)
(output omitted)
```

```
. sts list
```

```
      failure _d: failed
analysis time _t: time1
```

Time	Beg. Total	Fail	Net Lost	Survivor Function	Std. Error	[95% Conf. Int.]	
2	6	1	0	0.8333	0.1521	0.2731	0.9747
4	5	2	-1	0.5000	0.2041	0.1109	0.8037
5	4	0	1	0.5000	0.2041	0.1109	0.8037
7	3	1	1	0.3333	0.1925	0.0461	0.6756
8	1	0	1	0.3333	0.1925	0.0461	0.6756

Notice how Stata listed the delayed entry at $t = 4$: "Net Lost" is -1. In order to conserve columns, rather than listing censorings and entries separately, Stata combines them into a single column containing censorings-minus-entries and labels that column as "Net Lost".

There is a level at which delayed entries cause considerable problems. In their presence, the Kaplan-Meier procedure for calculating the survival curve can yield absurd results. This happens when some late arrivals to the study enter after everyone before them has failed.

Consider the following output from `sts list` for such a dataset:

```
. sts list
```

```
      failure _d: failed
analysis time _t: time1
```

Time	Beg. Total	Fail	Net Lost	Survivor Function	Std. Error	[95% Conf. Int.]	
2	6	1	0	0.8333	0.1521	0.2731	0.9747
4	5	2	-1	0.5000	0.2041	0.1109	0.8037
5	4	0	1	0.5000	0.2041	0.1109	0.8037
7	3	1	1	0.3333	0.1925	0.0461	0.6756
8	1	1	0	0.0000	.	.	.
9	0	0	-3	0.0000	.	.	.
10	3	1	0	0.0000	.	.	.
11	2	1	1	0.0000	.	.	.

As you can see, we constructed these data to include three additional subjects to enter at $t = 9$, after everyone who was previously at risk had failed. At $t = 8$, $\hat{S}(t)$ has reached zero, never to return. Why does this happen? Note the product form of (8.1). Once a product term of zero has been introduced (which occurs at $t = 8$), the product is zero, and further multiplication by anything nonzero is pointless. This is a shortcoming of the Kaplan–Meier method, and in Section 8.3 we show that there is an alternative.

□ Technical Note

There is one other issue regarding the Kaplan–Meier estimator in regards to delayed entry. In cases where the earliest entry into the study occurs after time $t = 0$, one may still calculate the Kaplan–Meier estimator, but the interpretation changes. Rather than estimating $S(t)$, you are now estimating $S(t|t_{\min})$, where t_{\min} is the earliest entry time.

□

8.2.4 Interval truncation (gaps)

Interval truncation is really no different than censoring followed by delayed entry. The subject disappears from the risk groups for a while and then reenters. The only issue is making sure that our “No. at risk” calculations reflect this fact, but Stata is up to that.

As with delayed entry, if a subject with a gap reenters after a final failure—meaning that a prior estimate of $S(t)$ is zero—then all subsequent estimates of $S(t)$ will also be zero regardless of future activity.

8.2.5 Relationship to the empirical distribution function

The cumulative distribution function is defined as $F(t) = 1 - S(t)$, and in fact, by specifying the `failure` option, you can ask `sts list` to list the estimate of $F(t)$, which is obtained as one minus the Kaplan–Meier estimate:

```
. input id time0 time1 failed
      id      time0      time1      failed
1.   1         0         2         1
2.   2         0         4         1
3.   3         0         4         1
4.   4         0         5         0
5.   5         0         7         1
6.   6         4         7         0
7.   7         0         8         0
8. end

. stset time1, fail(failed) time0(time0)
(output omitted)
```

```
. sts list, failure
      failure _d: failed
      analysis time _t: time1
```

Time	Beg. Total	Fail	Net Lost	Failure Function	Std. Error	[95% Conf. Int.]	
2	6	1	0	0.1667	0.1521	0.0253	0.7269
4	5	2	-1	0.5000	0.2041	0.1963	0.8891
5	4	0	1	0.5000	0.2041	0.1963	0.8891
7	3	1	1	0.6667	0.1925	0.3244	0.9539
8	1	0	1	0.6667	0.1925	0.3244	0.9539

For standard nonsurvival datasets, the *empirical distribution function* (edf) is defined to be

$$\hat{F}_{\text{edf}}(t) = \sum_{j|t_j \leq t} n^{-1}$$

where we have $j = 1, \dots, n$ observations. That is, $\hat{F}_{\text{edf}}(t)$ is a step function that increases by $1/n$ at each observation in the data. Of course, $\hat{F}_{\text{edf}}(t)$ has no mechanism to account for censoring, truncation, and gaps, but in cases where none of these exist, it can be shown that

$$\hat{S}(t) = 1 - \hat{F}_{\text{edf}}(t)$$

where $\hat{S}(t)$ is the Kaplan–Meier estimate. To demonstrate, consider the following simple dataset, which has no censoring or truncation:

```
. input t
      t
1. 1
2. 4
3. 4
4. 5
5. end
. stset t
(output omitted)
. sts list, failure
      failure _d: 1 (meaning all fail)
      analysis time _t: t
```

Time	Beg. Total	Fail	Net Lost	Failure Function	Std. Error	[95% Conf. Int.]	
1	4	1	0	0.2500	0.2165	0.0395	0.8721
4	3	2	0	0.7500	0.2165	0.3347	0.9911
5	1	1	0	1.0000	.	.	.

This reproduces $\hat{F}_{\text{edf}}(t)$, which is a nice property of the Kaplan–Meier estimator. Despite its sophistication in dealing with the complexities caused censoring and truncation, it reduces to the standard methodology when these complexities do not exist.

8.2.6 Other uses of sts list

The `sts list` command lists the Kaplan–Meier survivor function. Let us use our hip-fracture dataset (the version already `stset`):

```
. use http://www.stata-press.com/data/cgg/hip2
(hip fracture study)
. sts list
```

failure _d: fracture analysis time _t: time1 id: id							
Time	Beg. Total	Fail	Net Lost	Survivor Function	Std. Error	[95% Conf. Int.]	
1	48	2	0	0.9583	0.0288	0.8435	0.9894
2	46	1	0	0.9375	0.0349	0.8186	0.9794
3	45	1	0	0.9167	0.0399	0.7930	0.9679
4	44	2	0	0.8750	0.0477	0.7427	0.9418
(output omitted)							
13	21	1	0	0.5384	0.0774	0.3767	0.6752
15	20	1	-2	0.5114	0.0781	0.3507	0.6511
16	21	1	0	0.4871	0.0781	0.3285	0.6283
(output omitted)							
35	2	0	1	0.1822	0.0760	0.0638	0.3487
39	1	0	1	0.1822	0.0760	0.0638	0.3487

`sts list` can also produce less detailed output. For instance, we can ask to see 5 equally spaced survival times in our data by specifying the `at()` option:

```
. sts list, at(5)
```

failure _d: fracture analysis time _t: time1 id: id							
Time	Beg. Total	Fail		Survivor Function	Std. Error	[95% Conf. Int.]	
1	48	2		0.9583	0.0288	0.8435	0.9894
13	21	18		0.5384	0.0774	0.3767	0.6752
25	10	9		0.2776	0.0749	0.1443	0.4282
37	2	2		0.1822	0.0760	0.0638	0.3487
49	1	0	

Note: Survivor function is calculated over full data and evaluated at indicated times; it is not calculated from aggregates shown at left.

`sts list` will also list side-by-side comparisons of the estimated survivor function. Recall that our hip-fracture data has two study groups, one that was assigned to wear an experimental protective device (`protect==1`), and a control group that did not wear the device (`protect==0`).

```
. sts list, by(protect) compare
      failure _d: fracture
      analysis time _t: time1
      id: id
```

		Survivor Function	
protect		0	1
time	1	0.9000	1.0000
	5	0.6000	1.0000
	9	0.4364	0.8829
	13	0.1870	0.7942
	17	0.0831	0.7501
	21	0.0831	0.7501
	25	.	0.5193
	29	.	0.4544
	33	.	0.3408
	37	.	0.3408
	41	.	.

`sts list` has options that allow you to control these lists to get the desired output; see [ST] `sts` for more details.

8.2.7 Graphing the Kaplan–Meier estimate

`sts graph` graphs (among other things) the Kaplan–Meier estimate. Typed without arguments, `sts graph` graphs the overall (estimated) survival function for your data.

```
. use http://www.stata-press.com/data/cgg/hip2
(hip fracture study)
. sts graph
      failure _d: fracture
      analysis time _t: time1
      id: id
```

which produces the graph shown in Figure 8.1.

(Continued on next page)

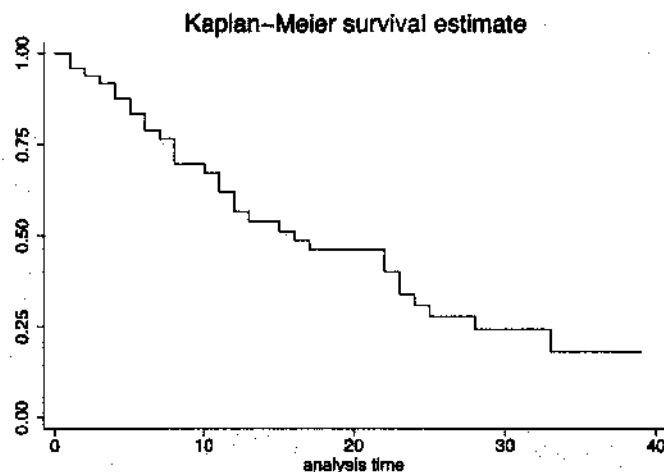


Figure 8.1: Kaplan-Meier estimate for hip-fracture data

`sts graph` has many options, including most of the options available with `twoway`. These options are designed to enhance the visual aspect of the graph and to show various aspects of the data.

`sts graph` with the `by()` option can plot multiple survival curves. We can type the following to compare the survival curves of the treatment group (`protect==1`) versus the control group (`protect==0`),

```
. sts graph, by(protect)
```

which produces the graph given in Figure 8.2.

(Continued on next page)

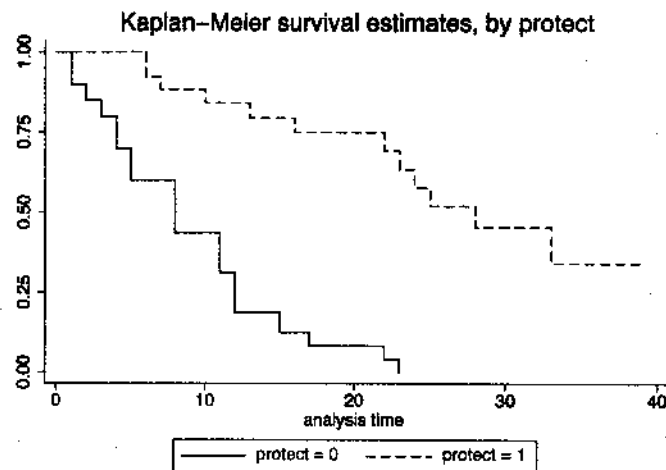


Figure 8.2: Kaplan-Meier estimates for treatment vs. control

As hoped, we see that the treatment group has a better survival experience than the control group.

`sts graph` also has options to indicate the number of censored observations, delayed entries, or the number of subjects at risk. Specifying `censored(number)`, for example, displays tick marks approximately at the times when censoring occurred and, above each tick mark, displays the number of censored observations,

```
. sts graph, censored(number)
```

which produces the graph given in Figure 8.3.

(Continued on next page)

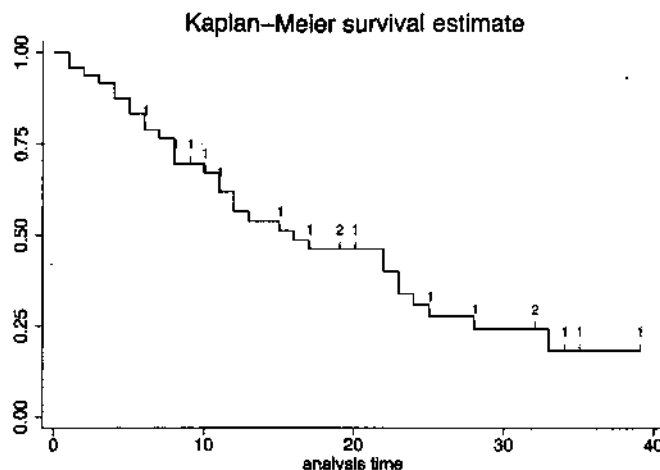


Figure 8.3: Kaplan–Meier with the number of censored observations

We could also combine `censored(number)` with `by(protect)` if we wish.

Option `enter` is an alternative to `censored()`. It attempts to show more information than `censored()` by adding the number who enter to the censoring information. The number that entered is displayed below the curve, and the number censored is displayed above the curve. Unlike `censored()`, these numbers are centered along the flat portions of the curve, not where the censorings or entries exactly occurred.

Option `lost` is another alternative. When we specify `lost`, the numbers displayed are censored minus entered, which is, say, the effective number lost. These numbers are displayed above the curve, centered along the flat portions.

Options `enter` and `lost` are most useful with delayed-entry data.

`sts graph` has other useful options for plotting Kaplan–Meier curves, including an option to add confidence bands, and the `noorigin` option to specify that we wish the plotted survival curve to begin at the first exit time instead of at $t = 0$. It is worth taking the time to read [ST] `sts graph` in the Stata manual and to explore the behavior of these and other options.

8.3 The Nelson–Aalen estimator

The cumulative hazard function is defined as

$$H(t) = \int_0^t h(u) du$$

where $h()$ is the hazard function. In Chapter 2, we discussed the count-data interpretation of $H(t)$, namely that it may be interpreted as the total number of expected failures in $(0, t)$ for a subject, if failure were a repeatable process.

In this section, we obtain a calculation formula for the empirical cumulative hazard function just as, in Section 8.2, we gave a calculation formula for the empirical survival function. One way we could do this would be to use the theoretical relationship between $H(t)$ and $S(t)$,

$$H(t) = -\ln\{S(t)\}$$

where for $S(t)$ we could use the Kaplan-Meier estimator. There exists, however, an alternative nonparametric method for estimating $H(t)$ that has better small-sample properties. The estimator is due to Nelson (1972) and Aalen (1978),

$$\hat{H}(t) = \sum_{j|t_j \leq t} \frac{d_j}{n_j}$$

where n_j is the number at risk at time t_j , d_j is the number of failures at time t_j , and the sum is over all distinct failure times less than or equal to t . That is, given some data,

id	t	failed
1	2	1
2	4	1
3	4	1
4	5	0
5	7	1
6	8	0

we can write the risk table

At time	n_j	d_j	No. censored
2	6	1	0
4	5	2	0
5	3	0	1
7	2	1	0
8	1	0	1

We calculate $e_j = d_j/n_j$, the expected number of failures at each observed time, and then sum these to form $\hat{H}(t)$:

At time	n_j	d_j	No. censored	e_j	$\hat{H}(t)$
2	6	1	0	0.1667	0.1667
4	5	2	0	0.4000	0.5667
5	3	0	1	0.0000	0.5667
7	2	1	0	0.5000	1.0667
8	1	0	1	0.0000	1.0667

$\hat{H}(t)$ is the Nelson-Aalen estimator of the cumulative hazard, and `sts list` with the `na` option will make this calculation:


```

. input id time failed
      id      time      failed
1.   1         2         1
2.   2         4         1
3.   3         4         1
4.   4         5         0
5.   5         7         1
6.   6         8         0
7. end

. stset time, fail(failed)
(output omitted)

. sts list, na
      failure _d: failed
analysis time _t: time

```

Time	Beg. Total	Fail	Net Lost	Nelson-Aalen Cum. Haz.	Std. Error	[95% Conf. Int.]	
2	6	1	0	0.1667	0.1667	0.0235	1.1832
4	5	2	0	0.5667	0.3283	0.1820	1.7639
5	3	0	1	0.5667	0.3283	0.1820	1.7639
7	2	1	0	1.0667	0.5981	0.3554	3.2015
8	1	0	1	1.0667	0.5981	0.3554	3.2015

The standard errors reported above are based on the variance calculation (Aalen 1978),

$$\widehat{\text{Var}}\{\hat{H}(t)\} = \sum_{j|t_j \leq t} \frac{d_j}{n_j^2}$$

and the confidence intervals reported are $\hat{H}(t) \exp\{\pm z_{\alpha/2} \hat{\phi}(t)\}$, where

$$\hat{\phi}^2(t) = \frac{\widehat{\text{Var}}\{\hat{H}(t)\}}{\{\hat{H}(t)\}^2}$$

estimates the asymptotic variance of $\ln \hat{H}(t)$ and $z_{\alpha/2}$ is the $(1 - \alpha/2)$ quantile of the normal distribution.

The `na` option is also valid for `sts graph`, in which case you are plotting the Nelson–Aalen cumulative hazard instead of the Kaplan–Meier survival function. Using the hip-fracture data, we can compare the Nelson–Aalen curves for treatment versus control,

```

. use http://www.stata-press.com/data/cgg/hip2
(hip fracture study)
. sts graph, by(protect) na

```

which produces Figure 8.4. Naturally, we see that the cumulative hazard is greater for the control group.

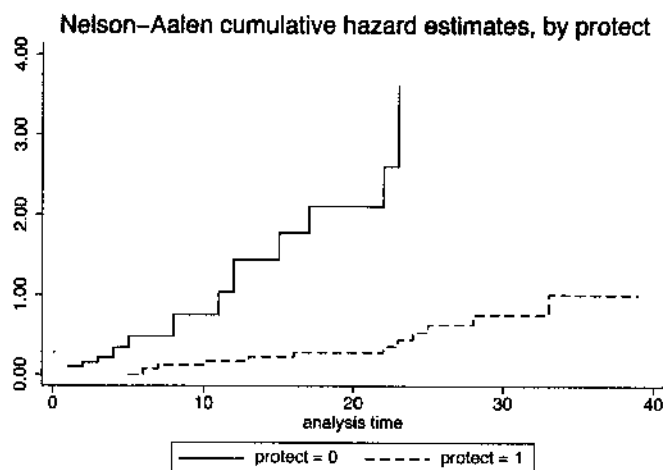


Figure 8.4: Nelson-Aalen curves for treatment vs. control

Theoretically, the survival and cumulative hazard functions are related by

$$S(t) = \exp\{-H(t)\}$$

or, if you prefer, $H(t) = -\ln\{S(t)\}$. We can use these relations to convert one estimate to the other. It has been shown that, in small samples, the Kaplan-Meier product-limit estimator is superior when estimating the survivor function, and the Nelson-Aalen estimator is superior when estimating the cumulative hazard function. For the survivor function and the cumulative hazard function, both the Kaplan-Meier estimator and the Nelson-Aalen estimator are consistent estimates of each, and the statistics are asymptotically equivalent (Klein and Moeschberger 2003, 104). That is, in very large samples, it does not matter how you estimate the survival function, whether by Kaplan-Meier or by transforming the Nelson-Aalen.

We can compare the survivor curves estimated both ways. The command `sts generate` will prove useful here. `sts generate` will create variables containing the Kaplan-Meier or Nelson-Aalen estimates, depending on which you request. If we specify the Kaplan-Meier, we can then convert this estimate into an estimate of the cumulative hazard, based on the Kaplan-Meier. If we specify the Nelson-Aalen, we can then convert this estimate into an estimate of the survivor function, based on the Nelson-Aalen.

With the hip-fracture data still in memory,

```
. sts generate kmS = s          /* obtain K-M survivor estimate */
. sts generate naH = na        /* obtain N-A cumulative hazard estimate */
. gen naS = exp(-naH)          /* calculate N-A survivor estimate */
. gen kmH = -log(kmS)          /* calculate K-M cumulative hazard estimate */
. label var kmS "K-M"
```

```
. label var naS "N-A"
. label var kmH "K-M"
. label var naH "N-A"
```

First, we graph the comparison of the survivor functions, producing Figure 8.5:

```
. line kmS naS _t, c(J J) sort
```

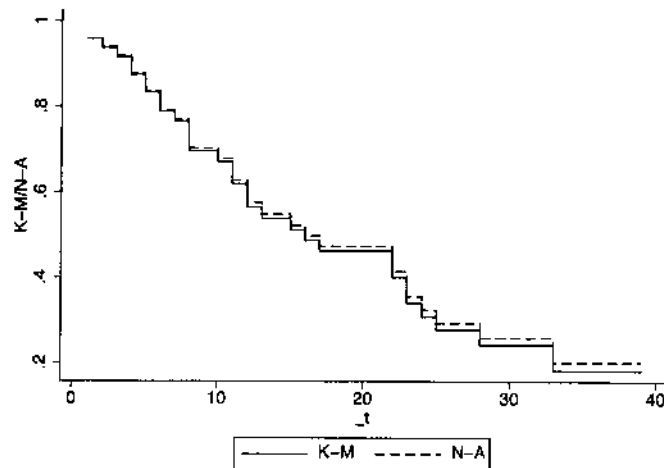


Figure 8.5: Estimated survivor functions

The top curve is the survivor function estimated by transforming the Nelson-Aalen estimator, and the bottom one is the Kaplan-Meier. These results are not unusual. It may be shown using a Taylor expansion that the Nelson-Aalen estimator of the survivor function is always greater than or equal to the Kaplan-Meier estimator. See, for example, Appendix 1 of Hosmer and Lemeshow (1999) for details.

Next, we graph the comparison of the cumulative hazard functions, producing Figure 8.6:

```
. line naH kmH _t, c(J J) sort
```

(Continued on next page)

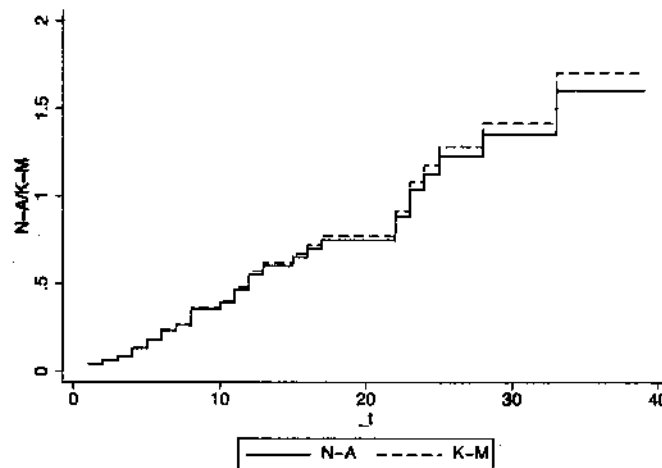


Figure 8.6: Estimated cumulative hazard functions

The top curve is the cumulative hazard obtained by transforming the Kaplan–Meier survivor function, and the bottom one is the Nelson–Aalen curve. By analogy to the above discussion, the Kaplan–Meier version of the cumulative hazard is always greater than or equal to the Nelson–Aalen estimator.

The above graphs are typical for most datasets in that they demonstrate that most of the time, the Kaplan–Meier and Nelson–Aalen estimators are quite similar once the transformation has been made so that they both estimate the same things. The Nelson–Aalen estimator, however, does offer one additional advantage. Recall in Section 8.2.3 where we discussed a shortcoming of the Kaplan–Meier estimator, namely, that once the estimator is zero, it remains zero regardless of future activity. The Nelson–Aalen estimator does not suffer from this problem, since it will increase with every failure event, even after the Kaplan–Meier estimate would have reached zero. In cases where this is at issue, the Nelson–Aalen estimator is to be preferred.

□ Technical Note

What lead to the Kaplan–Meier estimator falling to zero and staying at 0 in Section 8.2.3 was that (1) there was a gap during which no one in the data was at risk, and (2) there was no one in the group at risk before the gap who was known to have survived longer than when the gap started. That is, there was one group of people at risk, and that group evaporated before anyone from the second group came under observation, and in the first group, the empirically calculated survival function had indeed fallen to 0, meaning that at the last time anything was known about the group, all that were left were known to fail.

The Nelson–Aalen estimator deals better with data like that, but understand that if your data contain a gap during which no one is at risk, then your data are absolutely silent about what the risks might be during that period.

If there is such a gap, the Nelson–Aalen estimator produces a consistent estimate of

$$H^*(t) = \int_0^t h(u)I(u)du$$

where $I(u)$ is an indicator function equal to 0 over those periods where no subject is at risk, and 1 otherwise.

Hence, up until the gap, the Nelson–Aalen curve is an estimate of $H(t)$, but after the gap, it is an estimate of $H(t|\text{no risk during the gap})$.

Few datasets actually have such gaps. Understand, gaps in subject histories are not a problem; it is only when all the gaps in all the subjects add up in such a way that no one is at risk during a period that the dataset itself will have such a gap. When datasets do have such gaps, you want to avoid the Kaplan–Meier estimate and use the Nelson–Aalen, and still you need to think carefully as you interpret results.

For instance, let us imagine that you do a short study on smoking and follow subjects for only two years. The second oldest person in your data was 40 years old at enrollment, meaning that he was 42 at the end of the study. The oldest person was 80 at enrollment. You would not want to draw the conclusion that the risk of death due to smoking is 0 between ages 42 and 80. You observed no one during that period and so have no information on what the risk might be.

□

8.4 Estimating the hazard function

`sts` graph can also be used to plot an estimate of the hazard function, $h(t)$. Because the hazard is the derivative of the cumulative hazard, $H(t)$, it would seem straightforward to estimate the hazard itself. However, examination of Figure 8.4 and the subsequent graphs reveals that the estimated cumulative hazards available to us are step functions and thus cannot be directly differentiated. That is not to say that it is not straightforward to take Figure 8.4 and picture in our minds what the derivative of the cumulative hazard would look like; for the control group, it would be fairly linear (since the cumulative hazard is parabolic), and for the treatment group, the derivative would start off as constant for some time (cumulative hazard is initially linear) and then increase.

It turns out that we can estimate the hazard by taking the steps of the Nelson–Aalen cumulative hazard and smoothing them with a kernel smoother. More precisely, for each observed death time t_j , if we define the estimated hazard contribution to be

$$\Delta\hat{H}(t_j) = \hat{H}(t_j) - \hat{H}(t_{j-1})$$

we can obtain these hazard contributions using `sts generate newvar = h`. Then we can estimate $h(t)$ with

$$\hat{h}(t) = b^{-1} \sum_{j=1}^D K\left(\frac{t-t_j}{b}\right) \Delta\hat{H}(t_j)$$

for some symmetric density function $K()$ (the kernel) and bandwidth b ; the summation is over the D times at which failure occurs (Klein and Moeschberger 2003, 167).

This whole process can be automated by specifying option `hazard` to `sts graph`. Using our hip-fracture data, we can graph the estimated hazards for both the treatment and control groups as follows:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. sts graph, hazard by(protect) kernel(gaussian) width(4 5)
```

This produces Figure 8.7. The graph agrees with our informal analysis of the Nelson–Aalen cumulative hazards. Note that in applying the kernel smoother, we specified a Gaussian (normal) kernel function and bandwidths of four for the control group (`protect==0`) and five for the treatment group (`protect==1`), although suitable defaults would have been provided had we not specified these; see [R] `kdensity` for a list of available kernel functions and their definitions.

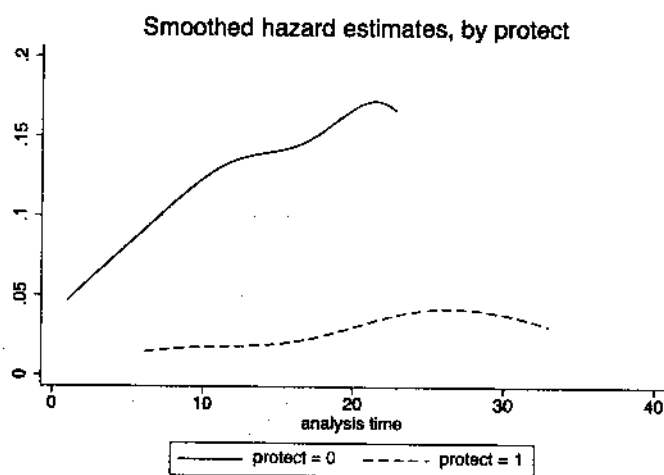


Figure 8.7: Smoothed hazard functions

One interesting feature of smoothed hazard plots is that you can assess the assumption of proportional hazards (the importance of which will be discussed in Chapter 9 on the Cox model) by plotting the estimated hazards on a log scale.

```
. sts graph, hazard by(protect) kernel(gaussian) width(4 5) yscale(log)
```

By examining Figure 8.8, we find the lines to be somewhat parallel, meaning that the proportionality assumption is violated only slightly. When hazards are proportional, the proportionality can be exploited by using a Cox model to assess the effects of treatment more efficiently; see Chapter 9.

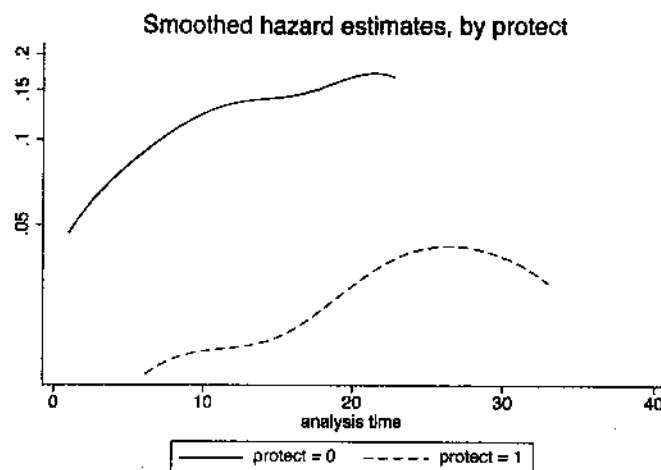


Figure 8.8: Smoothed hazard functions, log scale

8.5 Tests of hypothesis

In previous sections, we have discussed how to compare the survival experience between two (or more) groups using the `by()` option to `sts list` and `sts graph`. In order to form formal tests of hypothesis for the equality of survivor functions across groups, you can use the `sts test` command.

`sts test` will allow you to test the equality of survivor functions using one of several available nonparametric tests, namely, the log-rank (Mantel and Haenszel 1959), Wilcoxon (Breslow 1970; Gehan 1965), Tarone-Ware (1977), Peto-Peto-Prentice (Peto and Peto 1972; Prentice 1978), and generalized Fleming-Harrington (Harrington and Fleming 1982) tests.

All these tests are appropriate for testing the equality of survivor functions across two or more groups. Note that these tests do not test the equality of the survivor functions at a specific time point. Instead, they are global tests in the sense that they compare the overall survival functions. These tests work by comparing (at each failure time) the expected versus the observed number of failures for each group, and then combining these comparisons over all observed failure times. The above tests differ only in respect to how they weight each of these individual comparisons that occur at each failure time when combining these comparisons to form one overall test statistic.

8.5.1 The log-rank test

Typing

```
. sts test varname, logrank
```

performs the log-rank test. *varname* should be a variable taking on different values for different groups, such as 1, 2, 3, ..., or any distinct set of values. In our hip-fracture data, we have the variable *protect*==0 for the controls and *protect*==1 for the treatment group, and so we can type

```
. use http://www.stata-press.com/data/cgg/hip2
(hip fracture study)
. sts test protect, logrank
      failure _d: fracture
      analysis time _t: time1
                        id: id
```

Log-rank test for equality of survivor functions

protect	Events observed	Events expected
0	19	7.14
1	12	23.86
Total	31	31.00
	chi2(1) =	29.17
	Pr>chi2 =	0.0000

Although the log-rank test is a rank test, it can be viewed as an extension of the familiar Mantel-Haenszel (1959) test applied to survival data. Let's say that we are interested in comparing the survival experience of r groups. Assume that in all groups combined there are k distinct failure times. Further assume that at failure time t_j there are n_j subjects at risk, of which d_j fail and $n_j - d_j$ survive. Then the log-rank test statistic is computed by constructing, at each of the k distinct failure times, an $r \times 2$ contingency table and then combining results from these k tables. To be clear, for each time t_j , we would have a table of the form given in Table 8.1.

Table 8.1: $r \times 2$ contingency table for time t_j

Group	Failures at t_j	Survived at t_j	At risk at t_j
1	d_{1j}	$n_{1j} - d_{1j}$	n_{1j}
2	d_{2j}	$n_{2j} - d_{2j}$	n_{2j}
.	.	.	.
.	.	.	.
.	.	.	.
r	d_{rj}	$n_{rj} - d_{rj}$	n_{rj}
Total	d_j	$n_j - d_j$	n_j

The expected number of failures in group i at time t_j , under the null hypothesis of no difference in survival among the r groups, is $E_{ij} = n_{ij}d_j/n_j$. The chi-squared test statistic (distributed as χ^2 with $r - 1$ degrees of freedom under the null) is calculated as a quadratic form $\mathbf{u}'\mathbf{V}^{-1}\mathbf{u}$ using the row vector

$$\mathbf{u}' = \sum_{j=1}^k W(t_j)(d_{1j} - E_{1j}, \dots, d_{rj} - E_{rj}) \quad (8.2)$$

and the $r \times r$ variance matrix \mathbf{V} , where the individual elements are calculated by

$$V_{il} = \sum_{j=1}^k \frac{W^2(t_j)n_{ij}d_j(n_j - d_j)}{n_j(n_j - 1)} \left(\delta_{il} - \frac{n_{ij}}{n_j} \right) \quad (8.3)$$

where $i = 1, \dots, r$, $l = 1, \dots, r$, and $\delta_{il} = 1$ if $i = l$ and 0 otherwise.

The weight function $W(t_j)$ is what characterizes the different flavors of the tests computed by **sts test** and is defined as a positive function equal to zero when n_{ij} is zero. In the case of the log-rank test, $W(t_j) = 1$ when n_{ij} is nonzero.

The important thing to recognize is that the test statistic is constructed by combining the information from the contingency tables obtained at every failure time, and consequently, the test takes into account the entire survival experience and not just a specific point in time.

Stata shows a summary of those k tables in the output. The above output included the "Events observed" category that refers to the number of failures observed—19 for the first group and 12 for the second—and "expected" refers to the number of events that would be expected if the two groups shared the same survival function—7.14 in the first group and 23.86 in the second. In this case, the observed values are different enough from the expected so as to produce a highly significant chi-squared value. The log-rank test clearly rejects the null hypothesis that the survivor functions of the two groups are the same.

Recall that the relative survival experiences of the distinct groups may be characterized by the groups' hazard functions, and thus the null hypothesis of the tests computed by **sts test** may be expressed in terms of the hazards. Namely, for **sts test** the null hypothesis is

$$H_0 : h_1(t) = h_2(t) = \dots = h_r(t)$$

This is the null hypothesis for all tests computed by **sts test**, and the different tests vary in power according to how H_0 is violated. For example, the log-rank test is most powerful when the hazards are not equal but instead are proportional to one another.

8.5.2 The Wilcoxon test

Typing

```
sts test varname, wilcoxon
```

will perform the generalized Wilcoxon test of Breslow (1970) and Gehan (1965).

```
. use http://www.stata-press.com/data/cgg/hip2
(hip fracture study)
. sts test protect, wilcoxon
      failure _d: fracture
      analysis time _t: time1
      id: id
```

Wilcoxon (Breslow) test for equality of survivor functions

protect	Events observed	Events expected	Sum of ranks
0	19	7.14	374
1	12	23.86	-374
Total	31	31.00	0
	chi2(1) = 23.08		
	Pr>chi2 = 0.0000		

The Wilcoxon test is also a rank test and is constructed in the same way as the log-rank test, except that for this test we set $W(t_j) = n_j$ in (8.2) and (8.3). That is, the Wilcoxon test places additional weight to tables at earlier failure times—when more subjects are at risk—than to tables for failures later in the distribution. The Wilcoxon test is preferred to the log-rank test when the hazard functions are thought to vary in ways other than proportionally. However, there is a drawback. Because of this weighting scheme, the Wilcoxon test can prove unreliable if the censoring patterns differ over the test groups.

8.5.3 Other tests

In addition to the log-rank and the generalized Wilcoxon, `sts test` can also perform other tests:

1. *The Tarone–Ware test:* `sts test varname, tware`

Based on Tarone and Ware (1977), this test is nearly identical to the Wilcoxon test, except that the weight function is $W(t_j) = \sqrt{n_j}$ instead of $W(t_j) = n_j$. As such, more weight is given to the earlier failure times when more subjects are at risk, but not as much as the Wilcoxon test. As a result, this test is less susceptible to problems should there exist vast differences in the censoring patterns among the groups.

2. *The Peto-Peto-Prentice test: sts test varname, peto*

Based on Peto and Peto (1972) and Prentice (1978), this test uses as a weight function an estimate of the overall survivor function; that is, $W(t_j) = \tilde{S}(t_j)$, where $\tilde{S}(t_j)$ is similar (but not exactly equal) to the Kaplan-Meier estimator. While more computationally intensive, this test is not susceptible to differences in censoring patterns among groups.

3. *The Fleming-Harrington test: sts test varname, fh(p q)*

Due to Harrington and Fleming (1982), this test uses $W(t_j) = \{\hat{S}(t_j)\}^p \{1 - \hat{S}(t_j)\}^q$, where $\hat{S}(t_j)$ is the Kaplan-Meier estimator, and p and q are chosen by the user so that the weighting scheme is customized. When $p > q$, more weight is given to earlier failures than to later ones. When $p < q$, the opposite is true. Note that when $p = q = 0$, the test reduces to the standard log-rank test.

8.5.4 Stratified tests

The tests of equality of survival functions performed by `sts test`, in all its incarnations, may be modified so that the tests are stratified. In a stratified test, we perform the test separately for different subgroups of the data and then combine the test results into a single, overall statistic. For instance, we might be testing that group A had similar survival experiences as group B but then decide to stratify on sex, so we would test (1) that females in Group A had similar survival as females in Group B and (2) that males in Group A had similar survival as males in Group B. Why do this? Because perhaps we think the survival experiences of males and females differ. If Group A had more females than Group B, then if we just compared the two groups, we could be misled into thinking there is a difference due to A and B.

For the stratified test, the calculations in (8.2) and (8.3) are formed over each stratum to form stratum-specific quantities \mathbf{u}_s and \mathbf{V}_s , which are then summed over the strata to form $\mathbf{u} = \sum_s \mathbf{u}_s$ and $\mathbf{V} = \sum_s \mathbf{V}_s$. The test statistic is then calculated as $\mathbf{u}'\mathbf{V}\mathbf{u}$, and is redefined this way.

In our hip study, we know that age is an important factor associated with hip fractures. As age increases, so does the degree of osteoporosis, making bones more fragile and thus more susceptible to fractures. As a way of controlling for this, we might categorize each of the subjects into one of three age groups and then stratify on the age-group variable. In our data, age varies from 62 to 82, as we see by using `summarize`:

```
. use http://www.stata-press.com/data/cgg/hip2
(hip fracture study)
```

```
. summarize age
```

Variable	Obs	Mean	Std. Dev.	Min	Max
age	106	70.46226	5.467087	62	82

```
. gen agegrp = 1
```

```
. replace agegrp = 2 if age>65
(78 real changes made)
. replace agegrp = 3 if age>75
(20 real changes made)
. tab agegrp
```

agegrp	Freq.	Percent	Cum.
1	28	26.42	26.42
2	58	54.72	81.13
3	20	18.87	100.00
Total	106	100.00	

```
. sts test protect, logrank strata(agegrp)
      failure _d: fracture
analysis time _t: time1
      id: id
```

Stratified log-rank test for equality of survivor functions

protect	Events observed	Events expected(*)
0	19	7.15
1	12	23.85
Total	31	31.00

(*) sum over calculations within agegrp

```
chi2(1) = 30.03
Pr>chi2 = 0.0000
```

Even accounting for age, we still find a significant difference. The **detail** option will allow us to see the individual tables that go into the stratified results:

```
. sts test protect, logrank strata(agegrp) detail
      failure _d: fracture
analysis time _t: time1
      id: id
```

Stratified log-rank test for equality of survivor functions

-> agegrp = 1

protect	Events observed	Events expected
0	4	0.93
1	1	4.07
Total	5	5.00

```
chi2(1) = 13.72
Pr>chi2 = 0.0002
```

```

-> agegrp = 2

```

protect	Events observed	Events expected
0	10	4.34
1	8	13.66
Total	18	18.00

```

                                chi2(1) =    11.36
                                Pr>chi2 =    0.0007

```

```

-> agegrp = 3

```

protect	Events observed	Events expected
0	5	1.89
1	3	6.11
Total	8	8.00

```

                                chi2(1) =    8.32
                                Pr>chi2 =    0.0039

```

```

-> Total

```

protect	Events observed	Events expected(*)
0	19	7.15
1	12	23.85
Total	31	31.00

```

(*) sum over calculations within agegrp
                                chi2(1) =    30.03
                                Pr>chi2 =    0.0000

```

The above illustrates how the test works. `sts test` ran separate log-rank tests for each of the age groups, with the results of each of those individual tests being the same as if we typed

```

. sts test protect if agegrp==1, logrank
. sts test protect if agegrp==2, logrank
. sts test protect if agegrp==3, logrank

```

and then `sts test` combined those three tests into one.

9 The Cox proportional hazards model

Recall that in Section 3.2 on semiparametric models, we formulated an analysis of survival data where no parametric form of the survivor function is specified and yet the effects of the covariates are parametrized to alter the baseline survivor function (that for which all covariates are equal to zero) in a certain way. The Cox (1972) model, which assumes that the covariates multiplicatively shift the baseline hazard function is by far the most popular of choices, due in part to its elegance and computational feasibility.

The Cox proportional hazards regression model (Cox 1972) asserts that the hazard rate for the j th subject in the data is

$$h(t|\mathbf{x}_j) = h_0(t) \exp(\mathbf{x}_j \boldsymbol{\beta}_x) \quad (9.1)$$

where the regression coefficients $\boldsymbol{\beta}_x$ are to be estimated from the data.

The nice thing about this model is that $h_0(t)$, the baseline hazard, is given no particular parametrization and, in fact, is left unestimated. The model makes no assumptions about the shape of the hazard over time—it could be constant, increasing, decreasing, increasing and then decreasing, decreasing and then increasing, or anything else you can imagine, but what is assumed is that whatever the shape, it is the same for everyone. One subject's hazard is a multiplicative replica of another's; comparing subject j to subject m , the model states that

$$\frac{h(t|\mathbf{x}_j)}{h(t|\mathbf{x}_m)} = \frac{\exp(\mathbf{x}_j \boldsymbol{\beta}_x)}{\exp(\mathbf{x}_m \boldsymbol{\beta}_x)}$$

which is constant, assuming the covariates \mathbf{x}_j and \mathbf{x}_m do not change over time.

How exactly is this possible, given that a parametric regression model, in its likelihood calculations, contains terms using the hazard function and survival function and that at first blush, a likelihood calculation involving the baseline hazard seems inevitable? For the answer, we refer you back to Section 3.2, where we gave a heuristic approach that explains that by confining our analysis to only those times for which failure occurs and by conditioning on the fact that failures occurred only at those times, the baseline hazard drops out from the calculations. For a more technical treatment of how this happens, see Kalbfleisch and Prentice (2002, 71–74).

For now however, just realize that estimation is still possible, even after leaving the baseline hazard function unspecified, and this offers a considerable advantage when we are not able to make reasonable assumptions about the shape of the hazard; for example, $h_0(t) = a$ (constant) or $h_0(t) = apt^{p-1}$ (Weibull). Compared to these parametric

approaches, the advantage of the semiparametric Cox model is that we do not need to make assumptions about $h_0(t)$, assumptions about which we may be wrong and which, if we are wrong, could produce misleading results about β_x . The cost is a loss in efficiency; if we knew the functional form of $h_0(t)$, we could do a better job of estimating β_x .

9.1 Using stcox

Stata's `stcox` command fits Cox proportional hazards models. After `stsetting` your dataset or loading a dataset that has already been `stset`, you type `stcox` followed by the `x` (independent) variables. Note that this syntax differs from most of Stata's other estimation commands in that you need not specify a response variable. For survival data, the response is the triple (t_0, t, d) , which denotes the time span $(t_0, t]$ with failure/censoring indicator d , and Stata remembers these variables from when you `stset` your data.

Below, we also specify the `nohr` option, for reasons explained later:

```
. use http://www.stata-press.com/data/cgg/hip2
(hip fracture study)
. stcox protect, nohr
      failure _d: fracture
      analysis time _t: time1
      id: id
Iteration 0:  log likelihood = -98.571254
Iteration 1:  log likelihood = -86.655669
Iteration 2:  log likelihood = -86.370792
Iteration 3:  log likelihood = -86.36904
Refining estimates:
Iteration 0:  log likelihood = -86.36904
Cox regression -- Breslow method for ties
No. of subjects =          48          Number of obs   =          106
No. of failures =          31
Time at risk   =          714
Log likelihood =   -86.36904          LR chi2(1)       =          24.40
                                      Prob > chi2      =          0.0000
```

_t _d	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
protect	-2.047599	.4404029	-4.65	0.000	-2.910773 -1.184426

These results report that for our hip-fracture data,

$$h(t|\text{protect}) = h_0(t) \exp(-2.047599 * \text{protect})$$

Recall that in our hip-fracture data, the variable `protect` equals 1 or 0; it equals 1 if the subject wears an inflatable hip-protection device and 0 otherwise. These results say that

$$\begin{aligned}h(t|\text{protect} == 0) &= h_0(t); \\h(t|\text{protect} == 1) &= h_0(t) \exp(-2.047599)\end{aligned}$$

and so the ratio of these hazards (the *hazard ratio*) is $\exp(-2.047599) = 0.12904437$.

Whatever the hazard rate at a particular time for those who do not wear the device, the hazard at the same time for those who do wear the device is 0.129 times that hazard, which is substantially less. More exactly, these results report that if we constrain the hazard rate of those who wear the device to a multiplicative constant of the hazard of those who do not wear the device, then that multiplicative constant is estimated to be 0.129.

In obtaining these results, we made no assumption about the time profile of the hazard, although we would guess, if forced, that the hazard probably increases. We did, however, assert that whatever the shape of the hazard for the group `protect==0`, it is that same shape for the group `protect==1` but multiplied by a constant.

9.1.1 The Cox model has no intercept

Note that Stata reported no intercept for the model above, which makes the Cox model different from many of Stata's other estimation commands. The Cox model has no intercept because the intercept is subsumed into the baseline hazard $h_0(t)$, and mathematically speaking, the intercept is unidentifiable from the data. Pretend that we added an intercept to the model,

$$h(t|\mathbf{x}_j) = h_0(t) \exp(\beta_0 + \mathbf{x}_j\boldsymbol{\beta}_x)$$

then

$$h(t|\mathbf{x}_j) = \{h_0(t) \exp(\beta_0)\} \exp(\mathbf{x}_j\boldsymbol{\beta}_x)$$

We would now call $\{h_0(t) \exp(\beta_0)\}$ our new baseline hazard. The value of β_0 is undefined because any value works as well as any other—it would merely change the definition of $h_0(t)$, which we do not define anyway.

9.1.2 Interpreting coefficients

Exponentiated individual coefficients have the interpretation of the ratio of the hazards for a one-unit change in the corresponding covariate. For instance, if the coefficient on variable `age_in_years` in some model is 0.18, then a 1-year increase in age increases the hazard by 20% because $\exp(0.18) = 1.20$. If the coefficient on the variable `weight_in_kilos` is -0.2231 , then a 1-kilogram increase in weight decreases the hazard by 20% because $\exp(-0.2231) = 0.8$. If the coefficient on variable `one_if_female` is 0.0488, then females face a hazard 5% greater than males because $\exp(0.0488) = 1.05$, and a 1-unit increase in `one_if_female` moves the subject from being male to female.

To see this more clearly, note that for a subject with covariates x_1, x_2, \dots, x_k , the hazard rate under this model is

$$h(t|x_1, x_2, \dots, x_k) = h_0(t) \exp(\beta_1 x_1 + \beta_2 x_2 + \dots + \beta_k x_k)$$

For a subject with the same covariates, except that x_2 is incremented by 1, the hazard would be

$$h(t|x_1, x_2 + 1, \dots, x_k) = h_0(t) \exp\{\beta_1 x_1 + \beta_2 (x_2 + 1) + \dots + \beta_k x_k\}$$

and the ratio of the two hazards is thus $\exp(\beta_2)$.

In our `stcox` example, we fit a model using the covariate `protect`, and the hazard ratio for this variable was estimated to be $\exp(-2.0476) = 0.129$. A subject wearing the device is thus estimated to face a hazard rate that is only 12.9% of the hazard faced by a subject who does not wear the device.

This hazard-ratio interpretation is so useful that, in the previous example, we had to go out of our way and specify the `nohr` (no hazard ratio) option to keep `stcox` from exponentiating the coefficients for us. Here is what we would have seen by default without the `nohr` option:

```
. use http://www.stata-press.com/data/cgg/hip2
(hip fracture study)
. stcox protect
      failure _d: fracture
      analysis time _t: time1
                   id: id
Iteration 0:  log likelihood = -98.571254
Iteration 1:  log likelihood = -86.655669
Iteration 2:  log likelihood = -86.370792
Iteration 3:  log likelihood = -86.36904
Refining estimates:
Iteration 0:  log likelihood = -86.36904
Cox regression -- Breslow method for ties
No. of subjects =          48          Number of obs   =          106
No. of failures =          31
Time at risk   =          714
Log likelihood = -86.36904          LR chi2(1)        =          24.40
                                      Prob > chi2       =          0.0000
```

_t _d	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
protect	.1290443	.0568315	-4.65	0.000	.0544336 .3059218

The only difference in results from typing `stcox protect`, `nohr` and `stcox protect` is that this time `stcox` reported hazard ratios—exponentiated coefficients—rather than the coefficients themselves. This is only a difference in how results are reported, not in the results themselves.

□ Technical Note

Note that the reported standard error also changed. When we specified the `nohr` option, we obtained the estimate $\hat{\beta}_x = -2.047599$ with the estimated standard error 0.4404029. When we left off the `nohr` option, we instead obtained $\exp(\hat{\beta}_x) = 0.1290443$ with the standard error 0.0568315, which is a result of fitting the model in this scale and was obtained by applying the *delta method* to the original standard error estimate. The delta method obtains the standard error of a transformed variable by calculating the variance of the corresponding first-order Taylor expansion, which in the case of the transform $\exp(\beta_x)$ amounts to multiplying the original standard error by $\exp(\hat{\beta}_x)$. This trick of calculation yields identical results as transforming the parameters prior to estimation and then re-estimating.

The next two columns in the original output report the Wald test of the null hypothesis $H_0: \beta_x = 0$ versus the alternative $H_a: \beta_x \neq 0$, and the numbers reported in these columns remain unchanged. In the new output, the test corresponds to $H_0: \exp(\beta_x) = 1$ versus $H_a: \exp(\beta_x) \neq 1$. Hazard ratios equal to one correspond to coefficients equal to zero because $\exp(0) = 1$. In the new output, the test corresponds to $\exp(\beta_x) = 1$, but what is in fact reported is the test for $\beta_x = 0$. That is, the z -statistic is calculated using the original coefficient and its standard error, not the transformed coefficient and the transformed standard error. Were it calculated the other way, you would get a different, yet asymptotically equivalent test. Confronted with this discrepancy, Stata leaves the results in the original metric because, in the original metric, they often have better small-sample properties. Tests based on estimates of linear predictors are often better left untransformed.

The confidence interval for the hazard ratio is that based on the transformed (exponentiated) endpoints of the confidence interval for the original coefficient. An alternative, asymptotically equivalent method would have been to base the calculation directly on the hazard ratio and its standard error, but this confidence interval would not be guaranteed to be entirely positive, even though we know a hazard ratio is always positive. For this reason, and because confidence intervals of the untransformed coefficients usually have better properties, Stata reports confidence intervals based on the “transform-the-endpoints” calculation.

□

`stcox`, like all Stata estimation commands, will redisplay results when invoked without arguments. Typing `stcox` would redisplay the above hazard-ratio output. Typing `stcox, nohr` would redisplay results but would report coefficients rather than hazard ratios.

9.1.3 The effect of units on coefficients

The units in which you measure the covariates \mathbf{x} make no substantive difference, but choosing the right units can ease interpretation.

If the covariate x_2 measures weight, it does not matter whether you measure that weight in kilograms or pounds—the coefficient will just change to reflect the change in units. If you fit the model using kilograms to obtain

$$h(t|\mathbf{x}) = h_0(t) \exp(\beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k)$$

and now you wish to substitute $x_2^* = 2.2x_2$, which is to say, the same weight but measured in pounds, then

$$h(t|\mathbf{x}) = h_0(t) \exp\{\beta_1 x_1 + (\beta_2/2.2)x_2^* + \cdots + \beta_k x_k\}$$

When estimating the coefficient on x_2^* , you are in effect estimating $\beta_2/2.2$, and results from `stcox` will reflect this in that the estimated coefficient for x_2^* will be the coefficient estimated for x_2 in the original model divided by 2.2. If the coefficient we estimated using kilograms was $\hat{\beta}_2 = 0.4055$, say, then the coefficient we estimate using pounds would be $\hat{\beta}_2/2.2 = 0.1843$, and this is what `stcox` would report. The models are, logistically speaking, the same. Weight would have an effect, and that effect in kilograms would be $0.4055 \times \text{kilograms}$. That same effect in pounds would be $0.1843 \times \text{pounds}$.

Note, however, that the effect on the reported hazard ratios is nonlinear. The estimated hazard ratio for a one-kilogram increase in weight would be $\exp(0.4055) = 1.5$. The estimated hazard ratio for a one-pound increase would be $\exp(0.1843) = 1.202$.

Changing the units of covariates to obtain hazard ratios reported in the desired units is a favorite trick among those familiar with proportional hazards models, and rather than remember the exact technique, it is popular to just let the software do it for you. For instance, if you are fitting a model that includes age,

```
. stcox protect age
```

and you want the hazard ratios reported for a five-year increase in age rather than a one-year increase, changing the units of age, type

```
. generate age5 = age/5
. stcox protect age5
```

If you do this with the hip-fracture dataset, in the first case you will get a reported hazard ratio of 1.110972, meaning that a 1-year increase in age is associated with an 11% increase in the hazard. In the second case, you will get a reported hazard ratio of 1.692448, meaning that a 5-year increase in age is associated with a 69% increase in the hazard, and note that $\ln(1.110972) = \ln(1.692448)/5$.

Changing units change coefficients and exponentiated coefficients (hazard ratios) in the expected way. On the other hand, shifting the means of covariates changes nothing that Cox reports, although, as we will demonstrate later, shifting a covariate pays dividends when using `stcox` to estimate the baseline survivor or cumulative hazard function, since doing so effectively changes the definition of what is considered “baseline”.

Nevertheless, coefficients and hazard ratios remain unchanged. Using the `hip2.dta` dataset, whether we measure age in years since birth or in years above 60 doesn’t matter, and in fact,

```
. stcox protect age
. generate age60 = age - 60
. stcox protect age60
```

will yield identical displayed results from `stcox`. In both cases, the estimated hazard ratio for age would be 1.110972.

In most linear-in-the-parameters models, shifting from where the covariates are measured causes a corresponding change in the overall intercept of the model, but since that overall intercept is wrapped up in the baseline hazard in the Cox model, there is no change in reported results. Note that for some constant shift c ,

$$\begin{aligned} h(t|\mathbf{x}) &= h_0(t) \exp\{\beta_1 x_1 + \beta_2(x_2 - c) + \cdots + \beta_k x_k\} \\ &= \{h_0(t) \exp(-\beta_2 c)\} \exp(\beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k) \end{aligned}$$

and so all we have really done is to redefine the baseline hazard (something we do not directly estimate anyway).

9.1.4 Estimating the baseline cumulative hazard and survivor functions

In the Cox model given in (9.1), $h_0(t)$ is called the baseline hazard function and

$$\exp(\mathbf{x}\beta_x) = \exp(\beta_1 x_1 + \cdots + \beta_k x_k)$$

is called the *relative hazard*, and thus $\mathbf{x}\beta_x$ is referred to as the *log-relative hazard*, also known as the risk score.

From (9.1), note that $h_0(t)$ corresponds to the overall hazard when $\mathbf{x} = \mathbf{0}$, because, in that case, the relative hazard is 1.

Although the Cox model produces no direct estimate of the baseline hazard, estimates of functions related to $h_0(t)$ can be obtained after the fact, conditional on the estimates of β_x from the Cox model. One may obtain estimates of the baseline survivor function $S_0(t)$ corresponding to $h_0(t)$, the baseline cumulative hazard function $H_0(t)$, and the baseline hazard contributions which may then be smoothed to estimate $h_0(t)$ itself.

We noted previously, when we fit the model of the relative hazard of hip fracture, that we suspected that the hazard was increasing over time, although nothing in the Cox model would constrain the function to have that particular shape. We can, indeed, verify our suspicion by asking `stcox` to return to us, along with the estimated Cox results, the estimate of the baseline cumulative hazard based upon them:

```
. use http://www.stata-press.com/data/cgg/hip2
(hip fracture study)
. stcox protect, basechazard(H0)
(output omitted)
. line H0 _t, c(J) sort
```

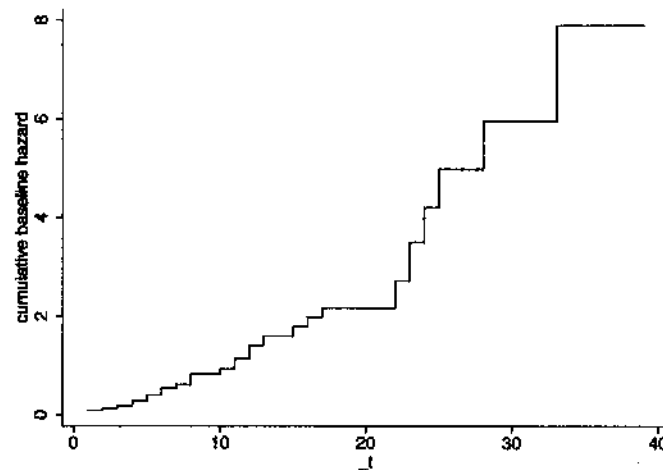


Figure 9.1: Estimated baseline cumulative hazard

We see that the cumulative hazard does appear to be increasing and at an increasing rate, meaning that the hazard itself is increasing (recall that the hazard is the derivative of the cumulative hazard).

Figure 9.1 is the cumulative hazard for a subject with all covariates equal to 0, which in our case, means `protect==0`, the control group. In general, the (nonbaseline) cumulative hazard function in a Cox model is given by

$$\begin{aligned}
 H(t|\mathbf{x}) &= \int_0^t h(u|\mathbf{x}) du \\
 &= \exp(\mathbf{x}\beta_{\mathbf{x}}) \int_0^t h_0(u) du \\
 &= \exp(\mathbf{x}\beta_{\mathbf{x}}) H_0(t)
 \end{aligned}$$

Thus, the cumulative hazard for those who do wear the hip-protection device is $H(t) = 0.129H_0(t)$, and we can draw both cumulative hazards on one graph,

```

. gen H1 = H0 * 0.1290
. label variable H0 H0
. line H1 H0 _t, c(J J) sort

```

which produces Figure 9.2.

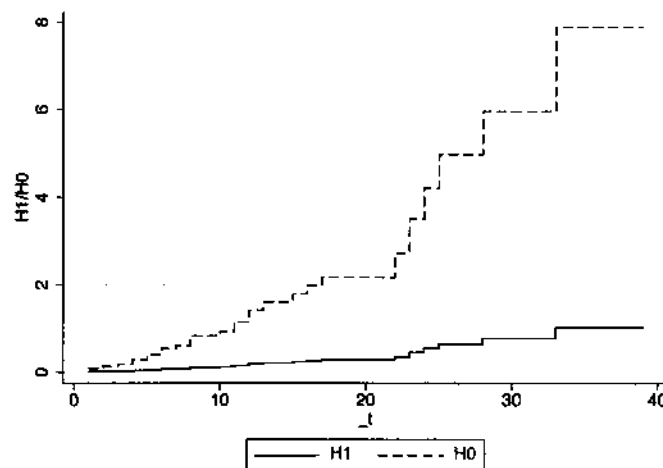


Figure 9.2: Estimated cumulative hazard: treatment vs. controls

We can also retrieve the estimated survivor function when we fit the model, using the `basesurv` option to `stcox`,

```
. stcox protect, basesurv(S0)
(output omitted)
. line S0 _t, c(J) sort
```

which produces Figure 9.3.

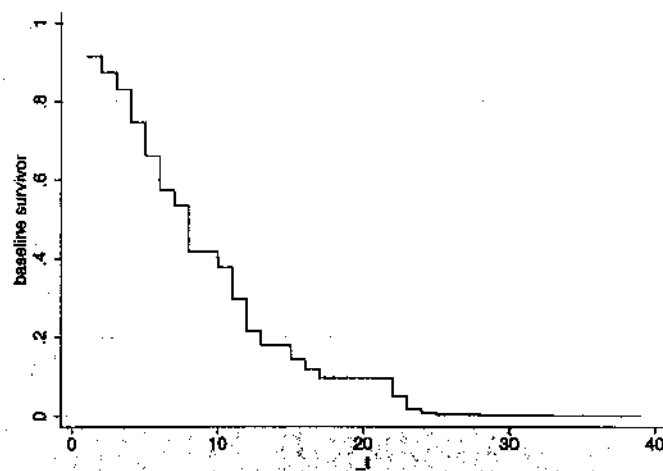


Figure 9.3: Estimated baseline survivor function

As with the cumulative hazard, the baseline survivor function $S_0(t)$ is the survivor function evaluated with all the covariates equal to zero. The formula for obtaining the value of the survivor function at other values of the covariates can be derived from first principles,

$$\begin{aligned} S(t|\mathbf{x}) &= \exp\{-H(t|\mathbf{x})\} \\ &= \exp\{-\exp(\mathbf{x}\boldsymbol{\beta}_x)H_0(t)\} \\ &= S_0(t)^{\exp(\mathbf{x}\boldsymbol{\beta}_x)} \end{aligned}$$

and so we can draw both survival curves on one graph by typing

```
. gen S1 = S0 ^ 0.1290
. label variable S0 S0
. line S1 S0 _t, connect(J J) sort
```

which produces Figure 9.4.

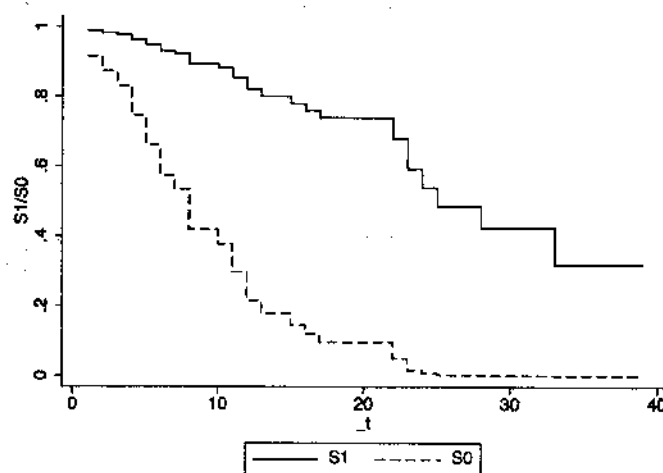


Figure 9.4: Estimated survivor: treatment vs. controls

In drawing these graphs, we have been careful to ensure that the points were connected with horizontal lines by specifying line's `connect()` option. We did that to draw the graphs so they would emphasize that the estimated functions are really step functions, no different than the Nelson–Aalen and Kaplan–Meier estimators of these functions in models without covariates. These functions are estimates of empirically observed functions, and failures only occur in the data at specific times.

□ Technical Note

If you fit a Cox regression model with no covariates and retrieve an estimate of the baseline survivor function, you will get the Kaplan–Meier estimate. For example, with the hip-fracture data, typing


```
. sts gen S2 = s
. stcox, estimate basesurv(S1)
```

will produce variables S1 and S2 that are identical up to calculation precision. (Note that to fit a Cox model with no covariates, we needed to specify the `estimate` option so that Stata knew we were not merely redisplaying results from the previous `stcox` fit.)

By the same token, if you fit a Cox regression model with no covariates and retrieve an estimate of the baseline cumulative hazard, you will get the Nelson–Aalen estimator.

For the details, we refer you to Kalbfleisch and Prentice (2002, 85–86) and just mention that the estimation of the baseline functions involves the estimation of quantities called *hazard contributions* at each failure time, and that each hazard contribution is the increase in the estimated cumulative hazard at each failure time. Nominally, these calculations take into account the estimated regression parameters, and so one can think of the estimated baseline survivor function from a Cox model as a “covariate-adjusted” Kaplan–Meier estimate—use the estimated β_x to put everyone on the same level by adjusting for the covariates, and then proceed with the Kaplan–Meier calculation.

In models with no covariates, these hazard contributions reduce to coincide with the calculations involved in the Kaplan–Meier and Nelson–Aalen curves. □

9.1.5 Estimating the baseline hazard function

In the previous section, we demonstrate how to use `stcox` to retrieve an estimate of the baseline survivor or baseline cumulative hazard function, $S_0(t)$ or $H_0(t)$, but note that estimates of $h_0(t)$ cannot be obtained directly from `stcox`. Since $h_0(t)$ is the derivative of $H_0(t)$, you might wonder, why not just take the derivative of the estimated $H_0(t)$ and use that as an estimate of $h_0(t)$? Or, since $h_0(t)$ is a function of the derivative of $S_0(t)$, why not follow a similar approach using the estimate of $S_0(t)$? The formal answer is that the derivative of these estimated functions is everywhere 0, except at the failure times, where it is undefined (recall that these are step functions).

This is all to say that if you want an estimate of the baseline hazard itself, you will have to somehow smooth out the discontinuities associated with these step functions. One way to do this is to use standard kernel-smoothing methodology, similar to what we did in Section 8.4. Formally, if we define the baseline hazard contribution for each observed failure time t_j as \hat{h}_{t_j} (see the technical note below), we can estimate $h_0(t)$ using

$$\hat{h}_0(t) = b^{-1} \sum_{j=1}^D K\left(\frac{t - t_j}{b}\right) \hat{h}_{t_j}$$

for some symmetric density function $K(\cdot)$ (the kernel) and bandwidth b ; the summation is over the D times at which failure occurs.

We can do this in Stata by first specifying option `basehc(newvar)` to `stcox` to obtain the baseline hazard contributions and then using `stcurve` to perform the smoothing.

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. stcox protect, basehc(h0)
(output omitted)
. stcurve, hazard at(protect=0)
```

This produces Figure 9.5, which is a graph of the estimated baseline hazard function (that is, the hazard for `protect==0`).

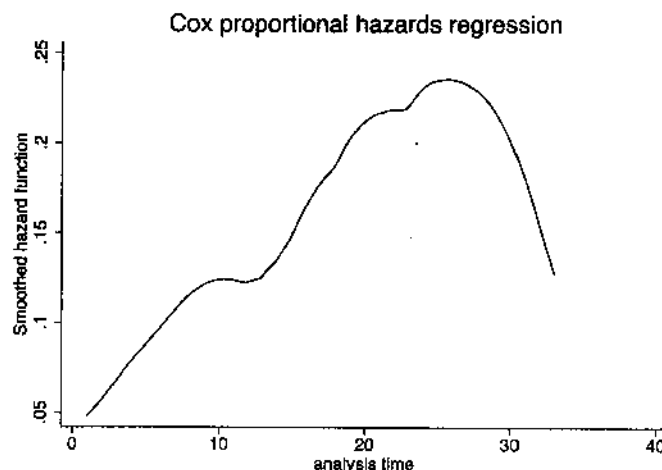


Figure 9.5: Estimated baseline hazard function

Alternately, using the same baseline hazard contributions, we can use `stcurve` to plot a comparison of the estimated hazards for treatments and controls, this time customizing the selection of kernel function and bandwidth:

```
. stcurve, hazard at1(protect=0) at2(protect=1) kernel(gaussian) width(4)
```

This produces Figure 9.6. Comparing this graph with Figure 8.7, we see the implications of the proportional hazards assumption. The hazards depicted in Figure 9.6 are indeed proportional, and if graphed on a log scale (we leave it to you as an exercise to try this!) they would be parallel, or at least close enough to parallel with respect to the smoothing. Still, the respective plots in both graphs are quite similar over the ranges they share in common on the x -axis.

Note that in Figure 8.7 the hazard functions are estimated over the range of observed failure times for each group while the hazards in Figure 9.6 are estimated over the complete range of observed failure times. This is one further consequence of the proportional hazards assumption. Under a Cox model, all failure times contribute to the estimate of the baseline hazard, not just those for which `protect==0`, and the baseline

hazard may in turn be transformed to the hazard for any covariate pattern using the proportionality assumption. However, when we estimate the hazard separately for each group (Figure 8.7), estimates are only valid over the range of observed failure times for that particular group.

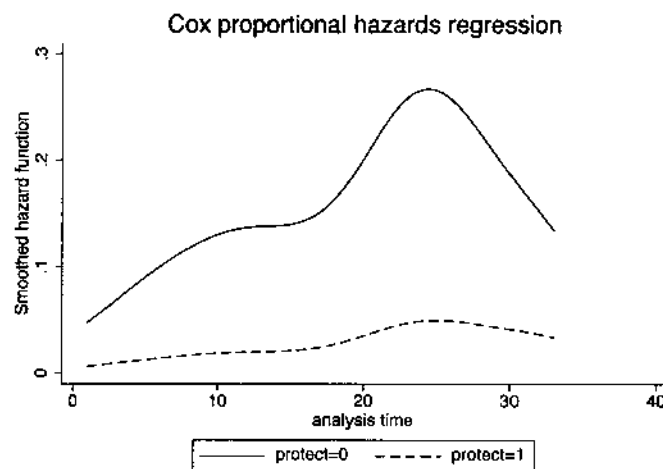


Figure 9.6: Estimated hazard functions, treatment vs. control

□ Technical Note

The baseline hazard contributions, \hat{h}_{t_j} , obtained from `stcox` are not the magnitudes of the steps of the estimated baseline cumulative hazard obtained from `stcox`. Instead, a form of the estimators derived from the estimated baseline survival function is used, as described in Kalbfleisch and Prentice (2002, 85–86). The difference between the estimators mirrors the difference between estimating a survival function using Kaplan–Meier and taking the Nelson–Aalen cumulative hazard and transforming it—they are asymptotically equivalent estimators of the same thing, and in practice, the difference is usually quite small; see Section 8.3.

□

`stcurve` is a wonderfully handy command for graphing estimated survival, cumulative hazard, and hazard functions after both `stcox` and `streg` that fits parametric survival models (Chapter 12). It is handy after `stcox` because it automates the process of taking quantities estimated at baseline using `stcox` and transforming them to adhere to covariate patterns other than baseline.

`stcurve` can graph

1. The survivor function. Type `stcurve, survival` after specifying option `basesurv(newvar)` to `stcox`.
2. The cumulative hazard function. Type `stcurve, cumhaz` after specifying option `basech(newvar)` to `stcox`.
3. The hazard function. Type `stcurve, hazard` after specifying option `basehc(newvar)` to `stcox`.

`stcurve` can graph any of those functions at the values for the covariates you specify. The syntax is as follows:

```
stcurve, ...at(varname=# varname=# ...)
```

If you do not specify a variable's value, the average value is used, and thus if the `at()` option is omitted altogether, a graph is produced for all the covariates held at their average values. This is why we had to specify `at(protect=0)` when graphing the baseline hazard function; had we not, we would have obtained a graph for the average value of `protect`, which would not be very meaningful considering `protect` is binary. The `at()` option can also be generalized to graph the function evaluated at different values of the covariates on the same graph. The syntax is

```
stcurve, ...at1(varname=# varname=# ...) at2(...) at3(...) ...
```

Earlier in this section, we graphed estimated cumulative hazard and survival functions, and we did so manually even though we could have used `stcurve` instead. We did this not to be mysterious but to emphasize the relationship between these functions at baseline and at covariate patterns other than baseline. We could have done the same with the hazard function (i.e., manually transform the baseline hazard contributions), but then we would have had to do the smoothing ourselves. In the case of hazard functions, we preferred to simply use `stcurve`.

To wit, Figures 9.1 through 9.4 all could have been produced with `stcurve` without having to **generate** any additional variables (outside of those produced by `stcox`). For example, Figure 9.2 can be replicated by

```
. stcox protect, basechazard(H0)
. stcurve, cumhaz at1(protect=0) at2(protect=1)
```

9.1.6 The effect of units on the baseline functions

The units in which you measure covariates (kilograms or pounds, inches or centimeters) change coefficients and hazard ratios in the obvious way but do not change the baseline cumulative hazard function, survivor function, and hazard contributions.

The origin from which you measure covariates—absolute zero or the freezing point of water, absolute weight or deviation from the normal (normal being the same for

everybody)—does not change coefficients β_x and hazard ratios (exponentiated coefficients), but it does change the estimated baseline cumulative hazard and baseline survivor, since you are changing how you define “all covariates equal to zero”.

Consider fitting the model

```
. stcox protect age
```

versus

```
. gen age60 = age - 60
. stcox protect age60
```

In the first case, $h_0(t)$ corresponds to a newborn who does not wear the hip-protection device (admittedly, not a very interesting hazard), and the second to a 60-year-old who does not wear the hip-protection device. Not surprisingly, the baseline cumulative hazard (and the baseline survivor) functions differ.

Yet, it seems innocuous enough to type

```
. stcox protect age, basesurv(S)
. line S _t, c(J) sort
```

and wonder what is wrong with Stata since the plotted baseline survivor function only varies between 0.99375 and 0.9999546, which appears incorrect since 30% of the subjects in the data were observed to fail. (Answer: you just plotted the survivor function for newborns.)

In the case of estimating the baseline survivor function, the problem can get worse than just misunderstanding a correctly calculated result. Numerical accuracy issues can arise. For example, try the following experiment:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
. gen age_big = age + 300
. stcox protect age_big, bases(S0)
```

All we have done is change the definition of what an age of “zero” means. In this new scaling, when you are born, your value of `age_big==300`, and you continue to age after that, so a person who is age 60 has `age_big==360`. The estimates of the coefficients and their associated hazard ratios will not be affected by this (nor should they).

Look, however, at the resulting estimate of the baseline survivor function:

```
. list S0
```

	S0
1.	1
2.	1
3.	1
(output omitted)	
105.	1
106.	1

What happened? The baseline survivor function is being predicted for a subject with `age_big==0`, meaning `age==300`. The probability of surviving is virtually 1. It is not exactly 1—it is really $1 - \epsilon$ where ϵ is a very small number. Upon further investigation, you would discover that the numbers the computer listed are not exactly 1, either; they merely round to 1 when displayed in a nine-digit (`%9.0g`) format. In fact, you would discover that there are actually 8 distinct values of `S0` in the listing, corresponding to the 3 bits of precision with which the computer was left when it struggled to present these numbers so very subtly different from 1 as accurately as it could.

The point is that this estimate is a poor estimate of the baseline survivor function, and it is not Stata's fault. If you push Stata the other way,

```
. gen age_small = age - 300
. stcox protect age_small, bases(S02)
```

you will again obtain fine estimates of β_x , but this time the baseline survivor function, corresponding to a person who is 300-years-old, will be estimated to be 0 everywhere:

```
. list S02
```

	S02
1.	0
2.	0
3.	0
(output omitted)	
105.	0
106.	0

This time the numbers really are exactly 0, even though they should not be, and even though the computer (in other circumstances) could store smaller numbers. Given the calculation formula for the baseline survivor function, this result could not be avoided.

The moral of the story is, if you intend to estimate the baseline survivor function, be sure that `x = 0` in your data corresponds to something reasonable. You only need to be concerned about this if you intend to estimate the baseline survivor function; the calculation of the baseline cumulative hazard (which is not bounded between 0 and 1) and the calculation of the baseline hazard contributions (upon which hazard function estimation is based) are more numerically stable.

9.2 Likelihood calculations

Cox regression results are based on forming, at each failure time, the *risk pool* or *risk set*, the collection of subjects who are at risk of failure and then maximizing the conditional probability of failure. The times at which failures occur are not relevant in a Cox model—the ordering of the failures is. As such, when subjects are tied (fail at the same time) and the exact ordering of failure is unclear, this requires special treatment. We first consider, however, the case of no ties.

9.2.1 No tied failures

Consider the straightforward data

```
. list
```

	subject	t	x
1.	1	2	4
2.	2	3	1
3.	3	6	3
4.	4	12	2

```
. stset t
```

```
(output omitted)
```

There are four failure times in these data—times 2, 3, 6, and 12—but the values of the times do not matter; only the order of the subjects matters. There are four distinct times from which we form four distinct risk pools:

1. *Time 2:*
Risk group (those available to fail): {1,2,3,4}
Subject #1 is observed to fail
2. *Time 3:*
Risk group: {2,3,4}
Subject #2 is observed to fail
3. *Time 6:*
Risk group: {3,4}
Subject #3 is observed to fail
4. *Time 12:*
Risk group: {4}
Subject #4 is observed to fail

At each of the failure times, we take as given that one of the subjects must fail, and we calculate the conditional probability of failure for the subject who actually is observed to fail. Thus, we have the likelihood function

$$L(\beta) = P_1 P_2 P_3 P_4$$

where each P_i , $i = 1, \dots, 4$ represents a conditional probability for each failure time.

The last conditional probability is the easiest to calculate. At time $t = 12$, given that one failure occurs, what is the probability that it will be Subject #4? The answer is $P_4 = 1$, since by that point only Subject #4 is available to fail.

The calculation of P_3 (corresponding to time $t = 6$) is the next easiest, and following the derivation of equation (3.2), we find that P_3 is the ratio

$$\begin{aligned} P_3 &= \frac{h(6|x_3)}{h(6|x_3) + h(6|x_4)} \\ &= \frac{\exp(x_3\beta)}{\exp(x_3\beta) + \exp(x_4\beta)} \end{aligned}$$

and note that this does not depend on the failure time $t = 6$. This is fundamental to Cox regression: the ordering of the failure times is what matters, not the actual times themselves. Similar arguments lead to the expressions for P_2 and P_1 ,

$$\begin{aligned} P_2 &= \frac{\exp(x_2\beta)}{\exp(x_2\beta) + \exp(x_3\beta) + \exp(x_4\beta)} \\ P_1 &= \frac{\exp(x_1\beta)}{\exp(x_1\beta) + \exp(x_2\beta) + \exp(x_3\beta) + \exp(x_4\beta)} \end{aligned}$$

and thus $L(\beta) = P_1 P_2 P_3 P_4$ can be expressed as

$$L(\beta) = \prod_{j=1}^4 \left(\frac{\exp(x_j\beta)}{\sum_{i \in R_j} \exp(x_i\beta)} \right)$$

where R_j is defined to be the risk set (those subjects at risk of failure) at time t_j .

Generalizing this argument to the case where we have k distinct observed failure times and multiple x variables gives the Cox likelihood function

$$L(\beta_x) = \prod_{j=1}^k \left(\frac{\exp(\mathbf{x}_j\beta_x)}{\sum_{i \in R_j} \exp(\mathbf{x}_i\beta_x)} \right) \quad (9.2)$$

Since we confine ourselves to only the individual binary analyses that occur at each failure time and make no assumption about the baseline hazard at times when failures do not occur, the Cox likelihood given in (9.2) is not a likelihood in the strict sense—it is a *partial likelihood*. However, for all intents and purposes, (9.2) can be treated as a likelihood; that is, maximizing it results in an estimate of β_x that is asymptotically normal with mean β_x and variance-covariance matrix equal to the inverse of the negative Hessian (matrix of second derivatives of (9.2) with respect to β_x). For a more technical discussion, see, for example, Kalbfleisch and Prentice (2002, 130–133).

As is often done with other likelihoods, estimates of β_x are obtained by maximizing the natural logarithm of $L(\beta_x)$, and not $L(\beta_x)$ itself. In Figure 9.7, we plot the log likelihood and label the value of β (a scalar in our small example) that maximizes it.

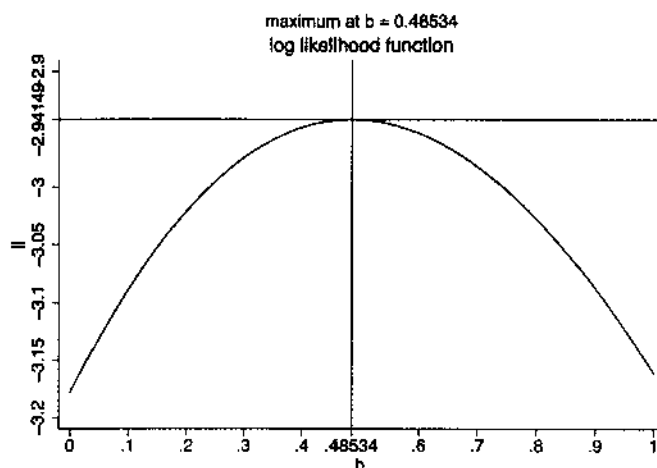


Figure 9.7: Log likelihood for the Cox model

The maximum (partial) likelihood estimate of β is $\hat{\beta} = 0.48534$, and this is exactly what `stcox` reports for these data:

```
. stcox x, nohr
      failure _d: 1 (meaning all fail)
      analysis time _t: t
Iteration 0:  log likelihood = -3.1780538
Iteration 1:  log likelihood = -2.9420159
Iteration 2:  log likelihood = -2.9414857
Iteration 3:  log likelihood = -2.9414857
Refining estimates:
Iteration 0:  log likelihood = -2.9414857
Cox regression -- no ties
No. of subjects =          4          Number of obs   =          4
No. of failures =          4
Time at risk   =          23
Log likelihood  =  -2.9414857          LR chi2(1)      =          0.47
                                          Prob > chi2     =          0.4915
```

	_t	_d	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
	x		.4853405	.7326297	0.66	0.508	-.9505874 1.921268

9.2.2 Tied failures

In the previous section, we calculated the probability that a particular subject failed, given that one failure was to occur. To do that, we used the formula

$$\Pr(j \text{ fails} | \text{risk set } R_j) = \frac{\exp(\mathbf{x}_j \boldsymbol{\beta}_x)}{\sum_{i \in R_j} \exp(\mathbf{x}_i \boldsymbol{\beta}_x)}$$

Introducing the notation $r_j = \exp(\mathbf{x}_j \boldsymbol{\beta}_x)$ allows us to express the above more compactly as $\Pr(j \text{ fails} | \text{risk set } R_j) = r_j / \sum_{i \in R_j} r_i$.

In our data from the previous section, it never arose that two subjects failed at the same time, but it is easy enough to imagine such a dataset:

. list

	subject	t	x
1.	1	2	4
2.	2	3	1
3.	3	3	3
4.	4	12	2

For these data, there are three risk pools:

1. *Time 2:*
Risk group (those available to fail): {1,2,3,4}
Subject #1 is observed to fail
2. *Time 3:*
Risk group: {2,3,4}
Subjects #2 and #3 are observed to fail
3. *Time 12:*
Risk group: {4}
Subject #4 is observed to fail

The question then becomes, How do we calculate the probability that both subjects 2 and 3 fail, given that two subjects from the risk pool are known to fail at $t = 3$? This is the conditional probability that we need for the second product term in (9.2).

The marginal calculation

One way we could make the calculation is to say to ourselves that subjects 2 and 3 did not really fail at the same time; it's just that we were limited as to how precisely we could measure the failure time. In reality, one subject failed and then the other, and we just do not know in which order they failed.

Define P_{23} to be the probability that subject 2 fails and then subject 3 fails given the current risk pool (which would no longer have subject 2 in it). Conversely, let P_{32} be the probability that subject 3 fails and then subject 2 fails given the risk pool reduced by the failure of subject 3. That is, if we knew that 2 failed before 3, then P_{23} is the contribution to the likelihood. If we knew that 3 failed before 2, then P_{32} is the contribution.

Following our previous logic, we find that

$$\begin{aligned} P_{23} &= \frac{r_2}{r_2 + r_3 + r_4} \frac{r_3}{r_3 + r_4} \\ P_{32} &= \frac{r_3}{r_2 + r_3 + r_4} \frac{r_2}{r_2 + r_4} \end{aligned} \quad (9.3)$$

and in fact, if we had the exact ordering then the substitution of either P_{23} or P_{32} in (9.2) would represent the two middle failure times (now that we have separated them).

However, since we do not know the order, we can instead take the probability that we observe subjects 2 and 3 to fail in any order as $P_{23} + P_{32}$, and use this term instead. Using our data, if $\beta = 0.75$, for example, then $P_{23} + P_{32} = 0.2786$, and this is what we would use to represent $t = 3$ in the likelihood calculations.

This method of calculating the conditional probability of tied failure events is called the marginal calculation, the exact-marginal calculation, or the continuous-time calculation. The last name arises because assuming continuous times makes it mathematically impossible that failures occur at precisely the same instant.

Using the `exactm` option to `stcox` specifies that ties are to be treated in this manner when calculating the likelihood.

□ Technical Note

Actually, the name `exactm` is a bit of a misnomer. The exact marginal method as implemented in most computer software (including Stata) is only an approximation of the method we have just described. Consider the (not too unusual) case where you have 10 tied failure times. The calculation of the exact marginal would then require $10! = 3,628,800$ terms and is computationally infeasible. Instead, the sum of the probabilities of the specific orderings is approximated using Gauss-Laguerre quadrature. However, we note that the approximation in this case is a computational issue and not one that simplifies any assumption about how we want to calculate the likelihood. When the approximation works well, we fully expect to retrieve (with negligible error) the true sum of the probability terms.

□

The partial calculation

Another way we could proceed is to assume that the failures really did occur at the same time and treat this as a multinomial problem. Given that two failures are to occur at the same time among subjects 2, 3, and 4, the possibilities are

- 2 and 3 fail
- 2 and 4 fail
- 3 and 4 fail

The conditional probability that 2 and 3 are observed from this set of possibilities is

$$p_{23} = \frac{r_2 r_3}{r_2 r_3 + r_2 r_4 + r_3 r_4}$$

Using $\beta = 0.75$, we would obtain $p_{23} = 0.3711$ for our data.

This is known as the partial calculation, the exact-partial calculation, the discrete-time calculation, or the conditional logistic calculation. The last name arises because this is also the calculation that conditional logistic regression uses to calculate probabilities when conditioning on more than one event taking place.

Of course, this raises the question: Which probability is correct, the marginal probability (0.2786) or the partial probability (0.3711)? The answer is a matter of personal taste in that you must decide how you want to think about tied failures: Do they arise from imprecise measurements (marginal method), or do they arise from a discrete-time model (partial method)? In practice, the difference between the calculations is usually not that severe, and admittedly we chose r_2 , r_3 , and r_4 to emphasize the difference.

Using the `exactp` option to `stcox` specifies that ties are to be treated in this manner when calculating the likelihood. The exact partial method implemented in Stata is exact; however, it can prove problematic since (1) in some cases it can take too long to calculate, and (2) numerical problems in the calculation can arise, and so it can produce bad results when risk pools are large and there exists many ties.

The Breslow approximation

Both the exact marginal and partial calculations are so computationally intensive that it has become popular to use approximations. When no option as to how to treat tied failures is specified, Stata assumes the `breslow` option and uses the Breslow (1974) approximation. This is an approximation of the exact marginal. In this approximation, the risk pools for the second and subsequent failure events within a set of tied failures are not adjusted for previous failures. So, for example, rather than calculating P_{23} and P_{32} from (9.3), the Breslow method instead uses

$$P_{23} = \frac{r_2}{r_2 + r_3 + r_4} \times \frac{r_3}{r_2 + r_3 + r_4} = \frac{r_2 r_3}{(r_2 + r_3 + r_4)^2}$$

$$P_{32} = \frac{r_3}{r_2 + r_3 + r_4} \times \frac{r_2}{r_2 + r_3 + r_4} = \frac{r_2 r_3}{(r_2 + r_3 + r_4)^2}$$

and thus the contribution to the likelihood is obtained as $P_{23} + P_{32} = 2r_2 r_3 / (r_2 + r_3 + r_4)^2$. Since the denominator is common to the failure events, this represents a significant reduction in the required calculations.

This approximation works very well when the number of failures in the risk group is small relative to the size of the risk group itself.

The Efron approximation

Efron's method of handling ties (Efron 1977) is also an approximation to the exact marginal, except that it adjusts the subsequent risk sets using probability weights. For the two failures that occur at $t = 3$, following the first failure, the second risk set is either $\{3, 4\}$ or $\{2, 4\}$. Rather than using $r_3 + r_4$ and $r_2 + r_4$ as the denominators for the second risk set, the approximation uses the average of the two sets, $(r_3 + r_4 + r_2 + r_4)/2 = (r_2 + r_3)/2 + r_4$. Thus, for our example,

$$P_{23} = \frac{r_2}{r_2 + r_3 + r_4} \times \frac{r_3}{\frac{1}{2}(r_2 + r_3) + r_4}$$

$$P_{32} = \frac{r_3}{r_2 + r_3 + r_4} \times \frac{r_2}{\frac{1}{2}(r_2 + r_3) + r_4}$$

and so

$$P_{23} + P_{32} = \frac{2r_2r_3}{(r_2 + r_3 + r_4)\{\frac{1}{2}(r_2 + r_3) + r_4\}}$$

This approximation is more accurate than Breslow's approximation but takes longer to calculate. Using the `efron` option to `stcox` specifies that ties are to be treated in this manner when calculating the likelihood.

9.2.3 Summary

The hip-fracture dataset is typical in terms of the number of tied failure times. The dataset contains 21 failure times and 31 failures, meaning an average of $31/21 = 1.48$ failures per failure time. In particular, at twelve of the times there is 1 failure, at eight times there are 2 failures, and at one time there are 3 failures.

Table 9.1 gives the results of fitting the model `stcox protect` using the various methods for handling tied failures:

Table 9.1: Comparison of methods for handling ties

Method	$\exp(\hat{\beta}_x)$	95% Conf. Interval	Command used
Exact marginal	0.1172	0.0481 0.2857	<code>stcox protect, exactm</code>
Exact partial	0.1145	0.0460 0.2846	<code>stcox protect, exactp</code>
Efron	0.1204	0.0503 0.2878	<code>stcox protect, efron</code>
Breslow	0.1290	0.0544 0.3059	<code>stcox protect</code>

If you do not specify otherwise, `stcox` uses the Breslow approximation. Specifying one of `efron`, `exactp`, or `exactm` will obtain results using the other methods. In any

case, the method used to handle ties is clearly displayed at the top of the output from `stcox`. Used with datasets with no ties, the output will state that there are no ties, in which case the specified method of handling ties becomes irrelevant.

To determine the number of ties in your data, you can do the following (after saving your data since this procedure will drop some observations):

```
. keep if _d                                /* Keep only the failures */
(75 observations deleted)
. sort _t
. by _t: gen number = _N                    /* Count the instances of _t */
. by _t: keep if _n==1                      /* Keep one obs. representing _t */
. * What is the average number of failures per failure time?
. summarize number
```

Variable	Obs	Mean	Std. Dev.	Min	Max
number	21	1.47619	.6015852	1	3

```
. * What is the frequency of number of failures?
. tabulate number
```

number	Freq.	Percent	Cum.
1	12	57.14	57.14
2	8	38.10	95.24
3	1	4.76	100.00
Total	21	100.00	

9.3 Stratified analysis

In stratified Cox estimation, the assumption that everyone faces the same baseline hazard, multiplied by their relative hazard,

$$h(t|\mathbf{x}_j) = h_0(t) \exp(\mathbf{x}_j \boldsymbol{\beta}_x)$$

is relaxed in favor of

$$\begin{aligned} h(t|\mathbf{x}_j) &= h_{01}(t) \exp(\mathbf{x}_j \boldsymbol{\beta}_x), & \text{if } j \text{ is in group 1,} \\ h(t|\mathbf{x}_j) &= h_{02}(t) \exp(\mathbf{x}_j \boldsymbol{\beta}_x), & \text{if } j \text{ is in group 2,} \end{aligned}$$

and so on. The baseline hazards are allowed to differ by group, but the coefficients $\boldsymbol{\beta}_x$ are constrained to be the same.

9.3.1 Obtaining coefficient estimates

Pretend that in our hip-displacement study we had a second dataset, this one for males, which we combined with the dataset for females to produce `hip3.dta`. We could imagine using that data to estimate a single effect:

```
. use http://www.stata-press.com/data/cgg/hip3, clear
. stcox protect
```

By fitting this model, however, we are asserting that males and females face the same hazard of hip fracture over time. Let's assume that, while we are willing to assume that the hip-protection device is equally effective for both males and females, we are worried that males and females might have different risks of fracturing their hips, and that this might bias our measurement of the effectiveness of the device.

One solution would be to include the variable `male` in the model (`male==0` for females and `male==1` for males) and so fit

```
. stcox protect male
```

If we did this, we would be asserting that the hazard functions for females and for males have the same shape, and that it is merely that one function is proportional to the other. Our concern is that the hazards might be shaped differently. Moreover, in this study, we do not care how the sexes differ; we just want to ensure that we measure the effect of `protect` correctly.

In this spirit, another solution would be to fit models for males and females separately:

```
. stcox protect if male
. stcox protect if !male
```

If we did this, we would be allowing the hazards to differ (as desired), but we would also obtain separate measures of the effectiveness of our hip-protection device. We want a single, more efficient estimate, or perhaps we want to test whether the device is equally protective for both sexes (regardless of the shape of the hazard for each), and this we cannot do with separate models.

The solution to our problem is stratified estimation:

```
. use http://www.stata-press.com/data/cgg/hip3
(hip fracture study)
. stcox protect, strata(male)
      failure _d: fracture
      analysis time _t: time1
      id: id
Iteration 0:  log likelihood = -124.27723
Iteration 1:  log likelihood = -110.21911
Iteration 2:  log likelihood = -109.47452
Iteration 3:  log likelihood = -109.46654
Iteration 4:  log likelihood = -109.46654
Refining estimates:
Iteration 0:  log likelihood = -109.46654
```

(Continued on next page)

Stratified Cox regr. -- Breslow method for ties					
No. of subjects =	148	Number of obs =	206		
No. of failures =	37				
Time at risk =	1703				
Log likelihood =	-109.46654	LR chi2(1) =	29.62		
		Prob > chi2 =	0.0000		
<hr/>					
t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
_d					
protect	.1301043	.0504375	-5.26	0.000	.0608564 .2781483
<hr/>					
Stratified by male					

Using this approach, we discover that wearing the hip-protection device reduces the hazard of hip fracture to just 13% of the hazard faced when the device is not worn.

In Table 9.2, we present the results of the analyses we would obtain from approaching the problem in each of the ways we have described.

Table 9.2: Comparison of various models for hip-fracture data

Command	$\exp(\hat{\beta}_x)$	95% Conf. Interval	
(1) <code>stcox protect</code>	0.0924	0.0442	0.1935
(2) <code>stcox protect male</code>	0.1315	0.0610	0.2836
(3) <code>stcox protect if !male</code>	0.1290	0.0544	0.3059
(4) <code>stcox protect if male</code>	0.1338	0.0268	0.6692
(5) <code>stcox protect, strata(male)</code>	0.1301	0.0609	0.2781

We obtained reasonably similar results when we fit the model separately for females and males (lines 3 and 4), but different results when we estimated constraining the baseline hazards to be equal (line 1). Our conclusion is that the hazards functions are not the same for males and females. In regards to how the hazard functions differ by gender, it makes virtually no difference whether we constrain the hazards to be multiplicative replicas of each other (line 2) or allow them to vary freely (line 5), but we would not have known that unless we had fit the stratified model.

9.3.2 Obtaining estimates of baseline functions

We obtain estimates of the baseline functions after stratified estimation in the same way as we do after unstratified estimation—we include the `basechazard()` or `basesurv()` options when we fit the model:

```
. stcox protect, strata(male) basech(H)
(output omitted)
```


The result of adding `basech(H)` is to create the new variable `H` containing the estimated baseline cumulative hazard functions. This one variable contains the baseline cumulative hazard function for females in those observations for which `male==0` and that for males in those observations for which `male==1`. We can thus graph the two functions using

```
. gen H0 = H if male==0
(100 missing values generated)
. gen H1 = H if male==1
(106 missing values generated)
. line H0 H1 _t, c(J J) sort
```

which produces Figure 9.8.

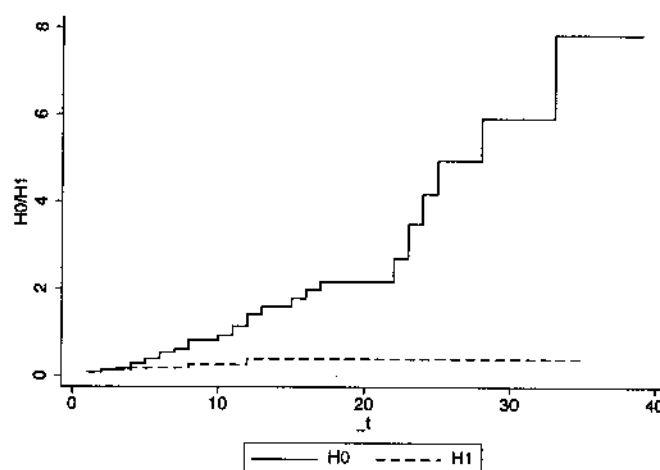


Figure 9.8: Estimated baseline cumulative hazard for males vs. females

Remember, these are the baseline cumulative hazard functions and not the baseline hazard functions themselves, and smoothed hazard function estimation is not available for stratified Cox models. Thus, if we think in terms of derivatives, we can compare the baseline hazards. Concerning females (H_0), there is a fairly constant slope until time $t=18$, and then, perhaps, a higher constant slope after $t=20$, suggesting an increasing hazard rate. Concerning males (H_1), there is a very gentle constant slope. In any case, one could argue equally well that the hazard functions are proportional, or not.

9.4 Cox models with shared frailty

The term *shared frailty* is used in survival analysis to describe regression models with random effects. A *frailty* is a latent random effect that enters multiplicatively on the

hazard function. In the context of a Cox model, the data are organized as $i = 1, \dots, n$ groups with $j = 1, \dots, n_i$ subjects in group i . For the j th subject in the i th group, then, 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 data.

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

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

and thus the log-frailties, ν_i , are analogous to random effects in standard linear models.

Before continuing, we must point out that a more detailed discussion of frailty models is given in Section 15.3 in the context of parametric models. Frailty is best first understood in terms of parametric models, since these models posit parametric forms of the baseline hazard function for which the effect of frailty can be easily described and displayed graphically. Shared frailty models are also best understood if compared and contrasted with unshared frailty models, which for reasons of identifiability do not exist with Cox regression. Therefore, for a more thorough discussion of frailty, we refer you to Section 15.3.

For our current discussion, it suffices to say that a Cox model with shared frailty is simply a random-effects Cox model. Shared frailty models are used to model within-group correlation; observations within a group are correlated because they share the same frailty, and the extent of the correlation is measured by θ . For example, we could have survival data on individuals within families, and we would expect (or at least be willing to allow) those subjects within each family to be correlated, since some families would inherently be more frail than others. When $\theta = 0$, the Cox shared frailty model simply reduces to the standard Cox model.

9.4.1 Parameter estimation

Consider the data from a study of kidney-dialysis patients, as described in McGilchrist and Aisbett (1991) and described in more detail in Section 15.3.2 on parametric frailty models. 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/cgg/kidney2, clear
(Kidney data, McGilchrist and Aisbett, Biometrics, 1991)
```

```
. list patient time fail age gender in 1/10, sepby(patient)
```

	patient	time	fail	age	gender
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 (`fail==1`) or right-censoring (`fail==0`). Among the covariates measured are `age` and `gender` (1 if female, 0 if male).

Note the use of the generic term “subjects”. In this example, the subjects are taken to be the individual catheter insertions and 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 the patient was admitted into the study. Thus, we have two subjects (insertions) for each group (patient). Since each observation represents one subject, we are not required to `stset` an id variable, although we would need to if we ever wished to `stsplit` the data at some later time.

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 parameters to account for the possible correlation. This is done by specifying option `cluster(patient)` to `stcox`. We do this below after first `stsetting` the data.

```
. stset time, failure(fail)
(output omitted)
. stcox age gender, nohr cluster(patient)
      failure _d: fail
      analysis time _t: time
Iteration 0:   log pseudo-likelihood = -188.44736
Iteration 1:   log pseudo-likelihood = -185.36881
Iteration 2:   log pseudo-likelihood = -185.11022
Iteration 3:   log pseudo-likelihood = -185.10993
Refining estimates:
Iteration 0:   log pseudo-likelihood = -185.10993
```

```

Cox regression -- Breslow method for ties
No. of subjects      =          76      Number of obs   =          76
No. of failures      =          58
Time at risk         =         7424
Log pseudo-likelihood = -185.10993      Wald chi2(2)    =          2.74
                                      Prob > chi2      =          0.2540
                                      (standard errors adjusted for clustering on patient)

```

_t	Coef.	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
age	.0022426	.0078139	0.29	0.774	-.0130724	.0175575
gender	-.7986869	.487274	-1.64	0.101	-1.753726	.1563526

Specifying `cluster(patient)` is equivalent to specifying both option `robust` and option `cluster(patient)`; in either case, you obtain the robust estimate of variance as described in the context of Cox regression by Lin and Wei (1989), with an added adjustment for clustering (see [P] `_robust`).

If there indeed exists within-patient correlation, the standard Cox model depicted above is misspecified. However, because we specified `cluster(patient)`, the standard errors of the estimated coefficients on `age` and `gender` are valid representations of the sample-to-sample variability of the obtained coefficients. We do not know exactly what the coefficients measure (for that we would need to know exactly how the correlation arises), but we can measure their variability, and in many instances we may still be able to test the null hypothesis that a coefficient is zero. That is, in many instances, testing that a covariate effect is zero under our misspecified model is equivalent to testing that the effect is zero under several other models that allow for correlated observations.

One such model is the shared frailty model, and more specifically the model where the shared frailty is gamma distributed (with mean one and variance θ). We fit this model in Stata by specifying option `shared(patient)` in place of `cluster(patient)`.

```

. stcox age gender, nohr shared(patient)
      failure _d: fail
      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

Log likelihood = -181.97453          Wald chi2(2)      =      11.66
                                      Prob > chi2        =      0.0029

```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
age	.0061825	.012022	0.51	0.607	-.0173801	.0297451
gender	-1.575675	.4626528	-3.41	0.001	-2.482458	-.6688924
theta	.4754497	.2673108				

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

Note: Standard errors of regression parameters are conditional on theta.

Given the estimated frailty variance, $\hat{\theta} = 0.475$, and the significance level of the likelihood-ratio test of $H_0: \theta = 0$, we conclude that under this model there is significant within-group correlation. To interpret the coefficients, let us begin by redisplaying them as hazard ratios.

```

. stcox
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

Log likelihood = -181.97453          Wald chi2(2)      =      11.66
                                      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
gender	.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.

The interpretation of the hazard ratios is the same as before, except that they are conditional on the frailty. For example, we interpret the hazard ratio for gender as indicating that, once we account for intragroup correlation via the shared frailty model, for a given level of frailty the hazard for females is about one-fifth that for males. Of

course, a subject's frailty would have a lot to say about the hazard, and when fitting a shared frailty model, we can use `stcox` to additionally estimate the frailties (or more precisely, the log-frailties ν_i) for us:

```
. stcox age gender, nohr shared(patient) effects(nu)
(output omitted)
. 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

By specifying option `effects(nu)`, we tell `stcox` to create a new variable `nu` containing the estimated random effects (the $\hat{\nu}_i$). After sorting, we find that the least frail (or strongest) patient is Patient 21 with $\hat{\nu}_{21} = -2.45$, and that the most-frail patient is Patient 7 with $\hat{\nu}_7 = 0.52$.

□ Technical Note

Estimation in the case of the Cox shared frailty model consists of two layers. In the outer layer, the optimization is in terms of θ only. For fixed θ , the inner layer consists of fitting a standard Cox model via penalized likelihood, with the ν_i introduced as estimable coefficients of dummy variables identifying the groups. The penalized likelihood is simply the standard Cox likelihood with an added penalty that is a function of θ . 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 fit. For this reason, 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. That is, when performing inference on these coefficients it is with the caveat that you are treating θ as known. For more details on estimation, see Chapter 9 of Therneau and Grambsch (2000).

□

9.4.2 Obtaining estimates of baseline functions

Baseline estimates for the Cox shared frailty model are obtained in the usual way, and the definition of baseline extends to include $\nu = 0$. For example, working with our kidney data, we can obtain an estimate of the baseline survivor function via the

`basesurv()` option to `stcox`, just as before. Before we do so, however, we first recenter age so that baseline corresponds to something more meaningful.

```
. generate age40 = age - 40
. stcox age40 gender, nohr shared(patient) basesurv(S0)
      failure _d: fail
      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	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
age40	.0061825	.012022	0.51	0.607	-.0173801 .0297451
gender	-1.575675	.4626528	-3.41	0.001	-2.482458 -.6688924
theta	.4754497	.2673108			

```
Likelihood-ratio test of theta=0: chibar2(01) =      6.27 Prob>=chibar2 = 0.006
Note: Standard errors of regression parameters are conditional on theta.
```

Note that recentering age has no effect on the parameter estimates, but it does produce a baseline survivor-function estimate (S0) that corresponds to a 40-year-old male patient with mean frailty, or equivalently a log-frailty of zero.

Because the estimation does not change, we know from the previous section that the estimated log-frailties still range from -2.45 to 0.52. We can use this information to produce a graph comparing the survival curves for 40-year-old males at the lowest level of frailty, mean value of frailty (baseline), and at the highest level of frailty.

```
. gen S_low = S0*exp(-2.45)
. gen S_high = S0*exp(0.52)
```

```
. line S_low S0 S_high _t if _t<200, c(J J J) sort ytitle("Survivor function")
```

This produces Figure 9.9. Note that for these data, the least frail patients have survival experiences that are far superior, even when compared to those with mean frailty. We have also restricted the plot to range from time 0 to time 200, where most of the failures occur.

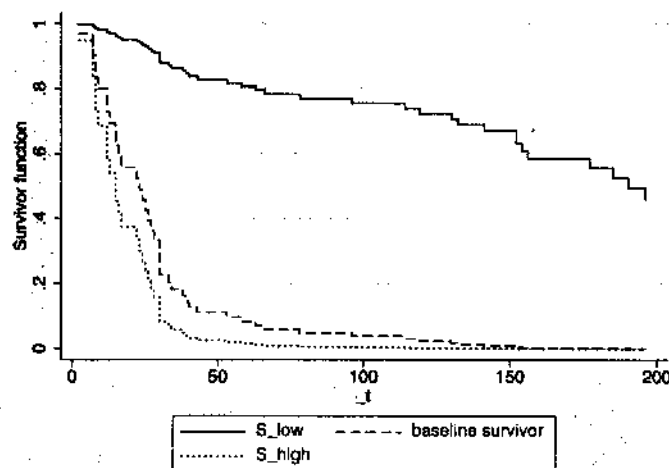


Figure 9.9: Comparison of survival curves for various frailty values

Currently, `stcurve` has no facility for setting the frailty. Instead, all graphs produced by `stcurve` are for $\nu = 0$. `stcurve` does, however, have an `outfile()` option for saving the coordinates used to produce the graph, and these coordinates can be transformed so that they correspond to other frailty values. For example, if we wanted smoothed baseline hazard plots for the same three frailty values used above, we could perform the following, remembering to first re-estimate in order to obtain the baseline hazard contributions:

```
. stcox age40 gender, nohr shared(patient) basehc(h)
      (output omitted)
. stcurve, hazard kernel(gaussian) outfile(basehaz)
note: all plots evaluated at frailty equal to one
. use basehaz, clear
```



```
. describe
Contains data from basehaz.dta
  obs:          101                      Kidney data, McGilchrist and
                                         Aisbett, Biometrics, 1991
  vars:          2                      27 Aug 2003 15:59
  size:          1,212 (99.9% of memory free)
```

variable name	storage type	display format	value label	variable label
haz1	float	%9.0g		Smoothed hazard function
_t	float	%9.0g		_t

Sorted by:

stcurve saved a dataset containing variable `_t` (time) and `haz1`, the estimated (after smoothing) hazard function. We have loaded these data into memory, and so now all we have left to do is to generate the other two hazards and then graph all three against `_t`.

```
. label variable haz1 "mean frailty hazard"
. gen haz_low = haz1*exp(-2.45)
. gen haz_high = haz1*exp(0.52)
. line haz_low haz1 haz_high _t if _t<200, yscale(log) sort
> ytitle("Smoothed hazard estimate")
```

This produces Figure 9.10. Note that the comparison of hazards for the frailty extremes matches that for the survivor function; the least frail individuals are quite immune to infection.

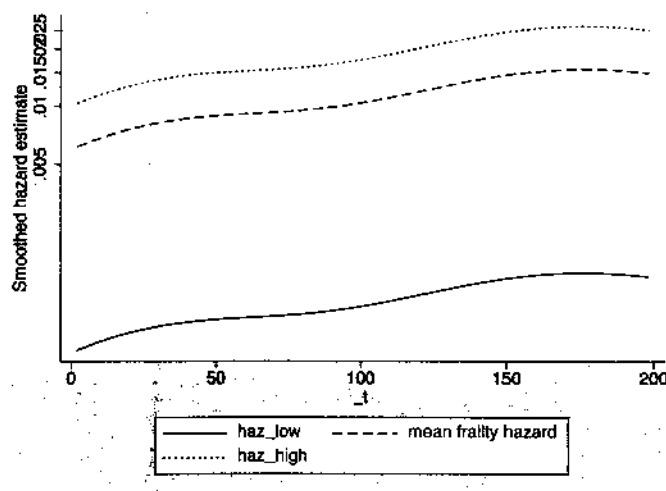


Figure 9.10: Comparison of hazards for various frailty values



10 Model building using stcox

`stcox` is extremely versatile, and you can fit many different models by controlling the covariate list that you supply to `stcox`. Recall the form of the Cox model,

$$h(t|\mathbf{x}) = h_0(t) \exp(\mathbf{x}\beta_x)$$

and note that all the “modeling” that takes place in the Cox model is inherent to the linear predictor $\mathbf{x}\beta_x = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k$, for k covariates.

10.1 Indicator variables

One type of variable used in modeling is the indicator variable; that is, a variable equal to 0 or 1, where a value of 1 indicates that a certain condition has been met. For example, in our hip-fracture data we parameterize the effect of wearing the hip-protection device using the indicator variable `protect`. We also measure the effect of gender by using the indicator variable `male`.

If, for example, we fit a standard Cox model with `protect` and `male` as covariates,

```
. use http://www.stata-press.com/data/cgg/hip3
(hip fracture study)
. stcox protect male
      failure _d: fracture
      analysis time _t: time1
      id: id
Iteration 0:  log likelihood = -150.85015
(output omitted)
Iteration 4:  log likelihood = -124.39469
Refining estimates:
Iteration 0:  log likelihood = -124.39469
Cox regression -- Breslow method for ties
No. of subjects =          148          Number of obs   =          206
No. of failures =           37
Time at risk   =          1703
Log likelihood =  -124.39469          LR chi2(2)       =          52.91
                                      Prob > chi2      =          0.0000
```

	<code>_t</code> <code>_d</code>	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
protect		.1315411	.051549	-5.18	0.000	.0610222	.2835534
male		.2523232	.1178974	-2.95	0.003	.10098	.6304912

we might wonder what would happen if we coded gender in the reverse way; i.e., 1 for females and 0 for males:

```
. gen female = !male
. stcox protect female
      failure _d: fracture
      analysis time _t: time1
      id: id

Iteration 0:  log likelihood = -150.85015
Iteration 1:  log likelihood = -129.05654
Iteration 2:  log likelihood = -125.53679
Iteration 3:  log likelihood = -124.3968
Iteration 4:  log likelihood = -124.39469
Refining estimates:
Iteration 0:  log likelihood = -124.39469
Cox regression -- Breslow method for ties
No. of subjects =      148                Number of obs =      206
No. of failures =       37
Time at risk   =     1703
Log likelihood =  -124.39469                LR chi2(2)      =     52.91
                                           Prob > chi2     =     0.0000
```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
protect	.1315411	.051549	-5.18	0.000	.0610222 .2835534
female	3.963172	1.851783	2.95	0.003	1.586065 9.902954

The estimated hazard ratio for **protect** remains unchanged (which is comforting). The hazard ratio for **male** in the first model was 0.2523, meaning that all else equal (the same value of **protect** in this case), males were estimated to face 0.2523 of the hazard of females. The estimated hazard ratio for **female** in the second model is 3.963, meaning that we estimate females to face 3.963 of the hazard of males. Since $1/0.2523 = 3.96$, the ultimate conclusion doesn't change.

If we instead look at the regression coefficients, we first obtained a coefficient for **male** of $\ln(0.2523) = -1.38$. For **female**, we obtained $\ln(3.963) = 1.38$, meaning that the magnitude of the shift in the linear predictor is the same regardless of how we code the variable.

10.2 Categorical variables

A categorical variable is a variable in which different values represent membership in different groups. Categorical variables usually take on values 1, 2, ..., but there is nothing special about those values, and a categorical variable could just as well take on 0, 1, ..., or 5, 9.2, 11.3, An indicator variable is just a special case of a categorical variable: it is a categorical variable that has two groups, and the two groups are labeled 0 and 1.

The important difference between indicator and categorical variables is that you do not include categorical variables directly in your model, or at least you should not if the categorical variable contains more than two unique values. Consider a categorical variable `race` where 1 = white, 2 = black, and 3 = hispanic. If you were to fit some model,

```
. stcox ... race ...
```

you would be constraining the effect of being hispanic to be twice the effect of being black. There would be no justifying that since the coding 1 = white, 2 = black, 3 = hispanic was arbitrary, and were you to use a different coding, you would be constraining some other relationship. A categorical variable that takes on m distinct values thus must be converted into $m - 1$ indicator variables:

```
. gen black = race==2
. gen hispanic = race==3
. stcox ... black hispanic ...
```

An alternative to creating the variables yourself is to use the `xi` command:

```
. xi: stcox ... i.race ...
```

Type `help xi` for more information on the use of `xi`.

In the example above, which uses the categorical variable `race`, we had three distinct groups (races) and included indicator variables for two of them (black and hispanic). Which two groups we choose makes no substantive difference. Let's pretend that in the above estimation we obtained the following results:

Variable	$\exp(\hat{\beta})$
black	1.11
hispanic	1.22

The interpretation would be that hispanics face a 22% greater hazard than whites and that blacks face an 11% greater hazard than whites. Had we instead fit a model on `white` and `hispanic`, we would have obtained

Variable	$\exp(\hat{\beta})$
white	0.607
hispanic	0.741

meaning that whites face 60.7% of the hazard that blacks (the omitted variable) face and that hispanics face 74.1% of that same hazard. These results are consistent. For instance, how do hispanics compare to whites? They face $.741/.607 = 1.22$ of the hazard faced by whites, and this corresponds exactly to what we estimated when we included the variables `black` and `hispanic` in the first analysis.

Which group you choose to omit makes no difference, but it does affect how results are reported. Comparisons will be made to the omitted group. Regardless of how you estimate, you can always obtain the other comparisons following the technique we demonstrated above.

In the example we have shown, the categorical variable has no inherent order; it was arbitrary that we coded 1 = white, 2 = black, and 3 = hispanic. Some categorical variables do have a natural order. For example, you may have a variable that is coded 1 = mild, 2 = average, and 3 = severe. Such variables are called *ordinal variables*. Even so, remember that the particular numbers assigned to the ordering are arbitrary (not in their ordinality, but in their relative magnitude), and it would be inappropriate to merely include the variable in the model. Is “severe” twice that of “average” or four times “average”? The coefficients you obtain on the corresponding indicator variables will answer that question.

10.3 Continuous variables

Continuous variables refer to variables that are measured on a well-defined, cardinal scale, such as age in years. What distinguishes continuous covariates from indicator and categorical variables is that continuous covariates are measured on an interval scale. That is, the distance between any two values of the variable is meaningful. This is not the case for indicator and categorical variables. We might have an indicator variable that assigns values 0 and 1 to each of the sexes, but the difference $1 - 0 = 1$ has no particular meaning. Or we might have a categorical variable coded 1 = mild, 2 = average, and 3 = severe, but “severe” – “mild” = $3 - 1 = 2$ has no particular meaning. On the other hand, the difference between two age values is meaningful.

Continuous variables can be directly included in your model; e.g.,

```
. stcox ... age ...
```

but doing so does not necessarily mean that you have correctly accounted for the effect of the variable. Including *age* by itself means that you are constraining each additional year of age to affect the hazard by a multiplicative constant that is independent of the level of age. It is not difficult to imagine failure processes where the effect of age increases with age (or diminishes with age). In such cases, it is popular to estimate the effect as quadratic in age:

```
. gen age2 = age^2
. stcox ... age age2 ...
```

This usually works fairly well, but caution is required. Quadratic terms have a minimum or a maximum, and if the extremum is in the range of the data, then you have just estimated that the effect of the variable, as it increases, increases and then decreases, or decreases and then increases. That might be reasonable for some problems, or it might simply be an artifact of using the quadratic to approximate what you simply believe to be a nonlinear but strictly increasing or decreasing effect.

This can be checked, and the procedure is relatively straightforward once you remember the quadratic formula. To demonstrate, we perform the following analysis, and display regression coefficients instead of hazard ratios (option *nohr*):

```

. use http://www.stata-press.com/data/cgg/hip2
(hip fracture study)
. gen age2 = age^2
. stcox protect age age2, nohr
      failure _d: fracture
      analysis time _t: time1
      id: id
Iteration 0:   log likelihood = -98.571254
Iteration 1:   log likelihood = -82.020427
Iteration 2:   log likelihood = -81.84397
Iteration 3:   log likelihood = -81.843652
Refining estimates:
Iteration 0:   log likelihood = -81.843652
Cox regression -- Breslow method for ties
No. of subjects =      48                Number of obs   =      106
No. of failures =      31
Time at risk   =      714
Log likelihood = -81.843652                LR chi2(3)       =      33.46
                                          Prob > chi2      =      0.0000

```

	_t	_d	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
protect			-2.336337	.4640283	-5.03	0.000	-3.245816 -1.426858
age			-.9093282	.8847323	-1.03	0.304	-2.643372 .8247153
age2			.0070519	.0061416	1.15	0.251	-.0049854 .0190891

Note that had we forgotten to specify `nohr`, we could just replay the results in this metric (without refitting the model) by typing `stcox, nohr`.

Define b to be the coefficient for `age`, a to be the coefficient for `age2`, and $x = \text{age}$. The quadratic function $ax^2 + bx$ has maximum or minimum at $x = -b/2a$, at which point the function changes direction. From the above model we estimate $\hat{a} = 0.0070519$ and $\hat{b} = -0.9093282$, and thus we estimate $-b/2a$ to be 64.5. Up until age 64.5, the relative hazard falls, and after that, it increases at an increasing rate. What is the range of age in our data?

```

. summarize age

```

Variable	Obs	Mean	Std. Dev.	Min	Max
age	106	70.46226	5.467087	62	82

We estimated a turning point at 64.5 within the range of `age`. Since it is not realistic to assume that the effect of age is ever to reduce the chances of a broken hip, our estimation results are problematic.

In the case we have just shown, we would just drop `age2` from the model, noting that the Wald test given in the output shows the effect of `age2` to be insignificant given the other covariates, and also noting that `age` and `age2` are jointly significant:

```
. test age age2
( 1)  age = 0
( 2)  age2 = 0
      chi2( 2) =    10.53
      Prob > chi2 =    0.0052
```

However, for purposes of this demonstration, let us put aside that easy solution and pretend that we really did have significant coefficients with an artifact that we do not believe.

There exist many solutions to this problem, and here we explore two of them. In our first solution, we note that it is only between ages 62 and 64.5 where we have a problem. It might be reasonable to say that there is no effect of age up until age 65, at which point age has a quadratic effect. We could try

```
. gen age65 = cond(age>65, age-65, 0)
. gen age65_2 = age65^2
. stcox protect age65 age65_2
```

and we invite you, the reader, to try that solution—it produces very reasonable results. What we did was to create the variable `age65`, which is defined as 0 for ages below 65 and as `age` minus 65 for ages above 65.

Our second solution would be to constrain age to have a linear effect up until age 65 and then a quadratic effect after that. In that case,

```
. gen age2gt65 = cond(age>65, age-65, 0)^2
. stcox protect age age2gt65
```

This produces even more reasonable results.

10.4 Interactions

The job of the analyst using Cox regression is to choose a parameterization of $\mathbf{x}\beta_x$ that fairly represents the process at work. It is not sufficient to say that the outcome is a function of variables A , B , and C , and then parameterize $\mathbf{x}\beta_x = \beta_A A + \beta_B B + \beta_C C$.

For instance, the effect of A may be constant (as we have it), or it may increase or diminish with the level of A , and in the latter, we must allow for that in our parameterization, perhaps by approximating the effect by including A^2 in the model. Similarly, the effect of A may increase or diminish with the level of another factor, say, B , and in such cases, it is common to approximate that by inclusion of the terms such as AB ; i.e., the *interaction* of A and B .

In a Cox model, the linear predictor $\mathbf{x}\beta_x$ is the logarithm of the relative hazard $\exp(\mathbf{x}\beta_x)$ since the baseline hazard function is multiplicatively shifted based on this value. As such, in what follows, we use the notation LRH (log relative hazard) to denote the linear predictor.

Returning to our example, if we wanted to entertain the possibility that the effect of wearing a hip-protection device might increase or decrease with age, we could include the term `protect*age` in the model:

```
. gen pXage = protect*age
. stcox protect age pXage
```

The sign of the coefficient for the variable `pXage` will tell whether the effect of `protect` decreases or increases with `age`, since the Cox model fit by the above command depends on the log relative hazard

$$\text{LRH} = \beta_1 \text{protect} + \beta_2 \text{age} + \beta_3 \text{protect} * \text{age}$$

which implies that

$$\frac{\partial \text{LRH}}{\partial \text{protect}} = \beta_1 + \beta_3 \text{age}$$

If $\beta_3 > 0$, the effect of `protect` increases with `age`; if $\beta_3 < 0$, it decreases; and if $\beta_3 = 0$, the effect of `protect` is constant. If you fit this model using `hip2.dta`, we would estimate $\hat{\beta}_3 = 0.036$ with 95% confidence interval $(-0.11, 0.18)$. Since this confidence interval includes zero, we cannot reject the hypothesis (at the 5% level) that there is no interaction effect.

Continuous variables can be interacted with continuous variables, continuous variables with indicator variables, and indicator variables with other indicator variables. Consider an indicator variable `one_if_female` and an indicator variable `one_if_over65`. The interaction `one_if_female*one_if_over65` takes on two values, 0 and 1, and 1 means that the subject is both female and over 65.

It is not uncommon to read that interaction effects should only be included in the model when the corresponding main effects are also included, but there is nothing wrong with including interaction effects by themselves. In some particular problem, being female *and* over 65 may be important, while either fact alone could be of little or no importance. The goal of the researcher is to parameterize what is reasonably likely to be true for the data for the problem at hand and not merely to follow a description.

In developing parameterizations and in interpreting parameterizations developed by others, we find it easiest to think in terms of derivatives—in terms of change. Consider the hypothetical parameterization

$$\text{LRH} = \beta_1 \text{weight} + \beta_2 \text{age} + \beta_3 \text{age}^2 + \beta_4 \text{protect} + \beta_5 \text{female} * \text{protect}$$

The easiest way to interpret this parameterization is to consider each of its component derivatives. What is the effect of `weight` in this model? The derivative of the above with respect to `weight` is

$$\frac{\partial \text{LRH}}{\partial \text{weight}} = \beta_1$$

and so the effect of `weight` is constant.

What is the effect of **age** in this model? It is

$$\frac{\partial \text{LRH}}{\partial \text{age}} = \beta_2 + 2\beta_3 \text{age}$$

meaning that the effect is not constant but changes with **age**, either decreasing ($\beta_3 < 0$) or increasing ($\beta_3 > 0$).

The effect of **protect** in this model is

$$\frac{\partial \text{LRH}}{\partial \text{protect}} = \beta_4 + \beta_5 \text{female}$$

meaning that **protect** has the effect β_4 for males and $\beta_4 + \beta_5$ for females. If $\beta_5 = 0$, the effect for males and females is the same.

The effect of **female** is

$$\frac{\partial \text{LRH}}{\partial \text{female}} = \beta_5 \text{protect}$$

meaning that there is only an effect if **protect**==1; males and females are otherwise considered to be identical. That is odd but perhaps not unreasonable for some problems.

One can build models by reversing this process. For instance, we start by thinking how the variables **weight**, **age**, **protect**, and **female** affect the outcome. From that, we can build a specification of LRH by

- (1) Thinking carefully about each variable separately and writing down how we think a change in the variable affects the outcome.
- (2) Multiplying each of the formulas developed in (1) by its corresponding variable.
- (3) Adding the resulting terms together and simplifying.

Let us demonstrate this process. We begin by stepping through each of the variables (**weight**, **age**, **protect**, **female**) one at a time and considering how changes in each affect the outcome:

- (1) What is the effect of **weight**? We think it is constant and call it α_1 .
- (2) What is the effect of **age**? We think the effect increases with **age** and express it as $\alpha_2 + \alpha_3 \text{age}$.
- (3) What is the effect of **protect**? We think **protect** has one constant effect for males and another for females, so the effect is $\alpha_4 + \alpha_5 \text{female}$.
- (4) What is the effect of **female**? We do not think there is one, other than what we have already considered. If we wanted to be consistent about it, an implication of (3) is that the effect must be $\alpha_5 \text{female}$, but we need not worry about consistency here. Each variable may be considered in isolation, and later, when we assemble the results, inconsistencies will work themselves out. The effect of **female**, therefore, is zero.

At this point, you must think carefully about each variable in isolation, but you can be sloppy in your thinking between variables. Do not worry about consistency—the next step will resolve that. You can write down that there is one effect at step (3) and then forget that you did that in step (4), as we did, or you can include the same effect more than once; it will not matter.

So, that is the first step. Write down all the derivatives without worrying about consistency. At this point we have

weight: α_1

age: $\alpha_2 + \alpha_3 \text{age}$

protect: $\alpha_4 + \alpha_5 \text{female}$

female: 0

Importantly, in addition to these derivatives, we have a story to go along with each because we have thought carefully about them. Each of the above are marginal effects, or said differently, they are derivatives, or they are the predicted effects of a change in the variable, so the next step is to integrate each with respect to its variable. Actually, multiplication by the variable will be good enough for our purposes. That is not quite equivalent to integration, yet the difference is consummate to the inclusion of the correct constant term, which the Cox model just subsumes into the baseline hazard anyway.

Therefore, multiplying each of our derivatives by the variable of interest and summing, we get

$$\text{LRH} = \alpha_1 \text{weight} + (\alpha_2 + \alpha_3 \text{age}) \text{age} + (\alpha_4 + \alpha_5 \text{female}) \text{protect} + 0$$

which simplifies to

$$\text{LRH} = \alpha_1 \text{weight} + \alpha_2 \text{age} + \alpha_3 \text{age}^2 + \alpha_4 \text{protect} + \alpha_5 \text{protect} * \text{female}$$

There is our model. If you go through this process with your own model, you might end up with a term such as $(\alpha_2 + \alpha_3) \text{weight}$, but if so, that does not matter. Just treat $(\alpha_2 + \alpha_3)$ as one separate coefficient when you fit the model. When you interpret the results, this one coefficient represents the cumulative effect of the two terms arose because you inconsistently wrote the same effect twice.

The important point of this is that models specified by LRH are not a random assortment of terms manufactured according to some predetermined set of rules, such as, include this if you include that, or do not include this unless you include that. Models are developed by carefully and substantively thinking about the problem, and thinking in terms of derivatives with respect to the component variables in your data is a useful way to proceed.

10.5 Time-varying variables

Time-varying covariates are handled automatically by Stata's *st* family of commands, including *stcox*. If you have data of the form

id	_t0	_t	age	calcium	protect	fracture
...						
12	0	5	64	11.77	0	0
12	5	8	64	11.34	0	1
...						

and you fit the model

```
. stcox protect age calcium
```

the covariate *calcium* will take on the value 11.77 for subject 12 during the period (0, 5] and the value 11.34 during (5, 8].

Thus, the only trick to using time-varying covariates is understanding what they mean and concocting the time-varying covariates that you want.

In terms of meaning, just understand that the hazard of the risk changes the instant the variable changes values; there is no anticipation of the change, nor is there any delay in the change taking effect. For instance, if you are an economist analyzing time to employment for the unemployed, you might theorize that whether the person is receiving unemployment payments would affect the chances of accepting employment. In that case, you might include a time-varying covariate *receiving_benefits*, and that covariate would change from 1 to 0 when the benefits ran out. Let us further assume that the estimated coefficient on *receiving_benefits* is negative. The instant benefits stop being paid, the hazard of employment increases. There would be nothing in the model that would say that the hazard of employment increases before the benefits ran out—in anticipation of that event—unless you included some other variable in your model.

Alternatively, imagine you are testing generators in overload situations. You run a generator under its approved load for a while and then increase the load. You have a time-varying covariate recording load in kVAs. In that case, you would be asserting that the instant the load increased, the hazard of failure increased, too. There would be no delay, even though failure might be caused by overheated bearings and heat takes time to accumulate.

In most situations, researchers ignore anticipation or delay effects, assuming they do not amount to much. That is fine as long as care is taken to ensure that any possible delay in the effect is insignificant.

Relatedly, it is useful conceptually to distinguish between time-varying covariates and time-varying coefficients, even though the latter can be estimated using the former. With time-varying covariates (also known as time-dependent covariates), the marginal effect remains the same but the variable changes, so the aggregate effect changes. With time-varying coefficients (also known as time-dependent coefficients), the variable remains the same but the marginal effect changes, so the aggregate effect still changes.

For example, suppose that you hypothesize that the effect of some covariate \mathbf{x} is β_1 up to analysis time $t = 5$ and $\beta_1 + \beta_2$ thereafter. You can fit such a model by specifying

$$\text{LRH} = \beta_1 + \beta_2 I(t > 5)\mathbf{x}$$

where $I()$ is the indicator function equal to one if the condition is true ($t > 5$ in this case) and zero otherwise. If your data looked like

id	_t0	_t	x	xtgt5	_d
1	0	3	22	0	1
2	0	4	17	0	0
3	0	5	23	0	0
3	5	12	23	23	1
4	0	5	11	0	0
4	5	21	11	11	0
...					

you could type

```
. stcox x xtgt5
```

and be done with it. That is, each subject who is known to have survived beyond $t = 5$ has a record for $t \leq 5$ and a record thereafter, so the values of the variable `xtgt5` can be recorded. Probably, however, your data look like

id	_t0	_t	x	_d
1	0	3	22	1
2	0	4	17	0
3	0	12	23	1
4	0	21	11	0
...				

in which case, fitting a Cox model with time-varying covariates would require a little work, which we discuss next.

10.5.1 Using `stcox`, `tvc()` `teexp()`

Consider the dataset in its more common form above. It is fairly easy to fit the Cox model with the time-varying covariate

$$\text{LRH} = \beta_1 \mathbf{x} + \beta_2 I(t > 5)\mathbf{x}$$

without having to concoct a new (expanded) dataset with the generated interaction variable `xtgt5`. The interaction term $I(t > 5)\mathbf{x}$ is of the form

$$\text{variable_in_my_data} \times \text{some_function_of_analysis_time}$$

where *variable_in_my_data* is \mathbf{x} and *some_function_of_analysis_time* is $I(t > 5)$. Since our interaction takes this specific form, we can just type

```
. stcox x, tvc(x) teexp(_t>5)
```

The first *x* in the variable list for *stcox* represents the main effect term $\beta_1 x$. The option *tvx(x)* specifies that I want *x* to be interacted with some function of analysis time, and the option *tevp(.t>5)* specifies the function of time to be used (note that *.t* represents analysis time here, since this is how *stset* defines it).

That the function *.t > 5* is an indicator with only one change point masks a lot of what can be done using *tvx()* and *tevp()*. Let us consider a more complex example:

Consider a more complete version of our hip-fracture data with an additional variable, *init_drug_level*, which gives the initial dosage level for a new experimental bone-fortifying drug.

```
. use http://www.stata-press.com/data/cgg/hip4, clear
(hip fracture study)
. list id _t0 _t _d init_drug_level in 1/10
```

	id	_t0	_t	_d	init_d-1
1.	1	0	1	1	50
2.	2	0	1	1	50
3.	3	0	2	1	50
4.	4	0	3	1	50
5.	5	0	4	1	100
6.	6	0	4	1	50
7.	7	0	5	1	100
8.	8	0	5	1	50
9.	9	0	5	0	50
10.	9	5	8	1	50

Note that the initial dosage of the drug comes in one of two levels, either 50mg or 100mg. To analyze the effect of this drug, we could easily fit the Cox model with

$$\text{LRH} = \beta_1 \text{protect} + \beta_2 \text{init_drug_level}$$

using

```
. stcox protect init_drug_level
```

but we would be assuming that the relative hazard between those with an initial dosage of 50mg and those with an initial dosage of 100mg would be constant over time. This would be fine if we were willing to assume that the hazard function for the duration of the whole study were completely determined by the initial dosage, but it is perhaps more reasonable to think that the hazard function changes with the *current* level of the drug in the patient's bloodstream, which we know to decay (nonlinearly) as time passes.

Let's instead pretend that the drug is administered only once upon entry to the study, and that its level in the bloodstream decays at an exponential rate such that the current level of the drug is equal to $\text{init_drug_level} \times \exp(-0.35t)$. For example, at $t = 2$, the current drug level would be $\exp(-0.7) = 0.5$ times its original level (in other words, 2 days is the half-life of the drug). We wish to fit the Cox model with *protect* and the current (time-varying) drug level as covariates, or equivalently,

$$\text{LRH} = \beta_1 \text{protect} + \beta_2 \text{init_drug_level} \exp(-0.35t)$$

This is accomplished by

```
. stcox protect, tvc(init_drug_level) texp(exp(-0.35*_t))
      failure _d: fracture
      analysis time _t: time1
      id: id
Iteration 0:  log likelihood = -98.571254
Iteration 1:  log likelihood = -83.895138
Iteration 2:  log likelihood = -83.241951
Iteration 3:  log likelihood = -83.214617
Iteration 4:  log likelihood = -83.214437
Refining estimates:
Iteration 0:  log likelihood = -83.214437
Cox regression -- Breslow method for ties
No. of subjects =          48          Number of obs =          106
No. of failures =          31
Time at risk   =          714
Log likelihood =  -83.214437          LR chi2(2)      =          30.71
                                      Prob > chi2     =          0.0000
```

	_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
rh							
	protect	.1183196	.0518521	-4.87	0.000	.050122	.2793091
t							
	init_drug_1	.8848298	.0601786	-1.80	0.072	.7744052	1.011

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

Note that now our estimated hazard ratios are now split into two categories: those for constant-with-time variables (rh) and those for time-varying variables (t), and we are reminded at the bottom of the output which function of analysis time was used to create the time-varying covariate. The hazard ratio 0.8848 is now interpreted as those with higher drug levels in *their bloodstreams* have a lower risk of having a hip fracture.

10.5.2 Using stsplot

Recall our simpler example from the beginning of Section 10.5, where our data look like

```
id  _t0  _t  x  _d
1   0   3  22  1
2   0   4  17  0
3   0  12  23  1
4   0  21  11  0
...
```

and we were fitting a Cox model with

$$\text{LRH} = \beta_1 x + \beta_2 I(t > 5)x$$

In the previous section, we mentioned that using *stcox*, *tvc(x) texp(_t>5)* avoided having to concoct the expanded dataset

id	_t0	_t	x	xtgt5	_d
1	0	3	22	0	1
2	0	4	17	0	0
3	0	5	23	0	0
3	5	12	23	23	1
4	0	5	11	0	0
4	5	21	11	11	0
...					

and fitting the same model with *stcox x xtgt5*.

Although not having to expand the data proves convenient, it is worth learning how to expand a dataset so that time-varying covariates may be introduced. Suppose that we instead wished to fit a Cox model using

$$\text{LRH} = \beta_1 x + \beta_2 I(t > 5)x + \beta_3 I(t > 10)x$$

so that the effect of *x* is β_1 over $(0, 5]$, $\beta_1 + \beta_2$ over $(5, 10]$, and $\beta_1 + \beta_2 + \beta_3$ over $(10, \infty)$. One limitation of the *stcox*, *tvc()* *texp()* is that only one function of analysis time may be specified, and here we have two, $I(t > 5)$ and $I(t > 10)$.

Another limitation of *stcox*, *tvc()* *texp()* is that it is unavailable if the exact marginal (*exactm*) or exact partial method (*exactp*) is used for treating ties. For these reasons, it is important to learn how to manually replicate the calculations performed by the convenience command *stcox*, *tvc()* *texp()* so that they may be generalized to cases where the convenience command is not available.

Toward this goal, Stata has the powerful *stsplsplit* command. For a simple illustration of *stsplsplit*, let's return to our model,

$$\text{LRH} = \beta_1 x + \beta_2 I(t > 5)x$$

and pretend that estimation was not possible via *stcox*, *tvc()* *texp()*. *stsplsplit* takes data such as

id	_t0	_t	x	_d
1	0	3	22	1
2	0	4	17	0
3	0	12	23	1
4	0	21	11	0
...				

and if you type

```
. stsplsplit new, at(5)
```

the data will be divided at analysis time $t = 5$, and a new variable *new* will be created so that *new*==0 for $t \leq 5$ and *new*==5 for $t > 5$:


```
. list id _t0 _t x _d
```

	id	_t0	_t	x	_d
1.	1	0	3	22	1
2.	2	0	4	17	0
3.	3	0	12	23	1
4.	4	0	21	11	0

```
. stsplrit new, at(5)
(2 observations (episodes) created)
. list id _t0 _t x _d new, sep(0)
```

	id	_t0	_t	x	_d	new
1.	1	0	3	22	1	0
2.	2	0	4	17	0	0
3.	3	0	5	23	0	0
4.	3	5	12	23	1	5
5.	4	0	5	11	0	0
6.	4	5	21	11	0	5

stsplrit automatically fills in the time and failure variables appropriately on the new records. Once the data have been split on $t = 5$, creating the *xtgt5* variable and fitting the model is easy,

```
. gen xtgt5 = x*(new==5)
. stcox x xtgt5
```

or we could type

```
. gen xtgt5 = x*(_t>5)
. stcox x xtgt5
```

Whether we use the *new* variable *stsplrit* created for us or the original *_t* makes no difference.

Although we illustrate *stsplrit* on a simple dataset, the original data with which we start could be inordinately complicated. The original might already have multiple records per subject, and some subjects might already have records split at $t = 5$ and some not. That will not cause *stsplrit* any difficulty.

stsplrit can split at multiple times simultaneously. You could type

```
. stsplrit cat, at(5 10 25 100)
```

and *stsplrit* would split the appropriate records at the indicated times. Note that the split points do not have to be equally spaced. For each subject, the new variable *cat* will contain 0 over the interval $(0, 5]$, 5 over $(5, 10]$, 10 over $(10, 25]$, 25 over $(25, 100]$, and 100 for records over the interval $(100, \infty)$. Any valid numlist may be used with *stsplrit*'s *at()* option; see *help numlist*.

□ Technical Note

`stjoin` is the inverse of `stsplit`, and it will safely rejoin split records; “safely” means that `stjoin` will verify that, in the rejoining, there will be no loss of information. `stjoin` does this by making sure no variables differ (other than those specifying the time span) on any records it considers joining. For instance, looking through the data, `stjoin` would join these two records,

id	_t0	_t	_d	x1	x2
57	5	7	0	3	8
57	7	9	0	3	8

to produce

id	_t0	_t	_d	x1	x2
57	5	9	0	3	8

but would not have joined the records if either `x1` or `x2` varied over the two records.

Thus, to use `stjoin` after `stsplit`, you must drop the new variable that `stsplit` created because it will differ over the now split records for a particular subject.

You can use `stjoin` anytime, not just after `stsplit`. It is always safe to type “`stjoin`”.

□

When the time-varying covariates vary continuously with time, `stsplit`, in an alternative syntax, proves even more useful. Returning to our hip-fracture example with time-varying covariates model,

$$\text{LRH} = \beta_1 \text{protect} + \beta_2 \text{init_drug_level} \exp(-0.35t)$$

we can use `stsplit` to manually replicate what was done by

```
. stcox protect, tvc(init_drug_level) texp(exp(-0.35*_t))
```

Recall that Cox regression only operates on times when failures actually occur. `stsplit` has an option that will split your records at all observed failure times (note that for this particular use of `stsplit`, no new variable is specified).

```
. use http://www.stata-press.com/courses/nc631/hip4, clear
(hip fracture study)
. stsplit, at(failures)
(21 failure times)
(452 observations (episodes) created)
. gen current_drug_level = init_drug_level*exp(-0.35*_t)
. stcox protect current_drug_level

      failure _d: fracture
analysis time _t: time1
              id: id

Iteration 0:  log likelihood = -98.571254
Iteration 1:  log likelihood = -83.895138
Iteration 2:  log likelihood = -83.241951
Iteration 3:  log likelihood = -83.214617
Iteration 4:  log likelihood = -83.214437
```

```

Refining estimates:
Iteration 0:  log likelihood = -83.214437
Cox regression -- Breslow method for ties
No. of subjects =          48          Number of obs   =          558
No. of failures =          31
Time at risk   =          714
Log likelihood =  -83.214437          LR chi2(2)       =          30.71
                                      Prob > chi2      =          0.0000

```

	<i>_t</i> <i>_d</i>	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
	protect	.1183196	.0518521	-4.87	0.000	.050122	.2793091
	current_dr-1	.8848298	.0601786	-1.80	0.072	.7744052	1.011

Except for some stylistic changes in the output, we have the same results as previously reported. The disadvantages to performing the analysis this way are (a) it took more effort on our part and (b) we had to expand our dataset from 106 observations to 558 observations. Such expansion is not always feasible (think of very large datasets with lots of failure times). The advantages of expanding the data are (a) we are not limited to how many functions of analysis time we can use in our Cox models and (b) we can estimate using any method to handle ties and are not limited to the *breslow* or *efron* methods.

stsplit, *at(failures)* is indeed a powerful tool since it uses the minimum amount of record splitting possible while still enabling your Cox model to fully capture the continuously changing nature of your time-varying covariate. When you have large datasets, this minimalistic approach to record splitting may prove invaluable.

In any case, just remember that when *stcox*, *tvc()* *teexp()* does not apply, use *stsplit*, *at(failures)*.

□ Technical Note

stsplit has another neat feature. Typing

```
. stsplit, at(failures) riskset(newvarname)
```

will not only split the records at every observed failure time but will also create the new variable *newvarname*, which uniquely identifies the risk sets. For every failure time, there is a group of subjects who were at risk of failure, and *newvarname* describes this grouping.

Once we have the risk sets organized into groups, we can (for example) use this to verify something we hinted at in Section 1.2, namely, that Cox regression is really just a collection of conditional logistic regressions, one taking place at each failure time.

We invite you to try the following:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
. stsplot, at(failures) riskset(riskid)
. stcox age protect, exactp nohr
. clogit _d age protect, group(riskid)
```

and verify that you get equivalent regression results. Note that since we do have ties, we had to tell *stcox* to handle them the discrete way--the same way a conditional logistic model would--using option *exactp*.

□

11 The Cox model: Diagnostics

Despite its semiparametric nature, the Cox model is no different from an ordinary least squares model in that there exist a multitude of diagnostics that will check for model misspecification, outliers, influential points, etc. All the usual notions of what constitutes a “well-fitting” model are also true for Cox models; the difference is in the details.

11.1 Testing the proportional hazards assumption

Despite the suggestive name “testing the proportional hazards assumption”, these tests are really just model specification tests that verify that you have adequately parameterized the model and that you have chosen a good specification for $\mathbf{x}\beta_x$. It just so happens that with Cox models, specification tests often go under the name “tests of the proportional hazards assumption”.

As with specification tests for other models, there exist many ways to test the proportional hazards assumption, and passing one test does not necessarily mean that others would be passed also. There are many ways that you can misspecify a model.

11.1.1 Tests based on re-estimation

The way specification tests generally work is that one searches for variables to add to the model. Under the assumption that the model is correctly specified, these added variables will add little or no explanatory power, and so one tests that these variables are “insignificant”. Even tests of the proportional hazards assumption follow that scheme.

The first test that we strongly recommend—and we recommend this test for all models, not just Cox models and not just survival models,—is called a link test. This test is easy to do and is remarkably powerful. What you do is type

```
. stcox ...                /* fit your Cox model */  
. linktest
```

which is really just shorthand for

```
. stcox ...  
. predict s, xb            /* obtain linear predictor */  
. gen s2 = s^2  
. stcox s s2
```

The point of the link test is to verify that the coefficient on the squared linear predictor (`s2`) is insignificant. The basis for this test is that one first estimates β_x from the standard Cox model and then estimates β_1 and β_2 from a second round model

$$\text{LRH} = \beta_1(\mathbf{x}\hat{\beta}_x) + \beta_2(\mathbf{x}\hat{\beta}_x)^2$$

Under the assumption that $\mathbf{x}\beta_x$ is the correct specification, $\beta_1 = 1$ and $\beta_2 = 0$. Thus, one tests that $\beta_2 = 0$.

This test is reasonably powerful for errors in specifying $\mathbf{x}\beta_x$ under the assumption that \mathbf{x} at least has the right variables in it, or more correctly, that the test is weak in terms of detecting the presence of omitted variables. If you fail this test, you need to go back and reconsider the specifications of the effects (derivatives) for each of the variables.

In terms of directly going after the proportional hazards assumption, a popular way is to interact analysis time with the covariates and verify that the effects of these interacted variables are not different from zero because the proportional hazards assumption states that the effects do not change with time except in ways that you have already parameterized. Or, more generally, if you have the correct specification, the effect will not change with time, and moreover, any variable you can think to add will have no explanatory power.

There are many ways to proceed. You could, as with the link test, start with the predicted $\mathbf{x}\beta_x$ from a first round model and estimate

$$\text{LRH} = \beta_1(\mathbf{x}\hat{\beta}_x) + \beta_2(\mathbf{x}\hat{\beta}_x)t$$

and test that $\beta_2 = 0$, or you could refit the entire model in one swoop and include many interactions,

$$\text{LRH} = \mathbf{x}\beta_{x1} + \mathbf{x}\beta_{x2}t$$

and test $\beta_{x2} = 0$. However, by far the most popular method is to fit separately one model per covariate and then perform separate tests of each,

- Fit the model $\text{LRH} = \mathbf{x}\beta_x + \beta_1(x_1t)$ and test $\beta_1 = 0$
- Fit the model $\text{LRH} = \mathbf{x}\beta_x + \beta_2(x_2t)$ and test $\beta_2 = 0$

and so on. If you are going to follow this approach, and if you have many covariates, we recommend that you worry about the effect of performing so many tests as if each were the only test you were performing. Some sort of adjustment, such as a Bonferroni adjustment, is called for. In the Bonferroni adjustment, if you are testing at the 0.05 level and performing k tests, you use $0.05/k$ as the critical level for each.

Alternatively, you could instead simply include all the covariates, interacted with `.t`, at once. We have been fitting a model on `age` and `protect`, and so we could type

```
. use http://www.stata-press.com/data/cgg/hip2, clear
. stcox protect age, tvc(age protect) texp(t)
. test [t]
```

If you instead want to follow the individual-regression approach, you would type

```
. stcox protect age, tvc(age) texp(t)
. stcox protect age, tvc(protect) texp(t)
```

Doing that, you will discover that neither term significantly interacts with time (even without a Bonferroni adjustment.)

There are two things to note at this point: (1) We could have omitted the `texp(t)` option in the above since this is the default function of analysis time used by `stcox`, `tvc()`, and (2) we handled tied failures using the default Breslow approximation. Had we instead wanted to use the marginal or partial methods for ties, we would need to `stsplit` our records at all observed failure times and generate the interactions ourselves; see Section 10.5.2.

In the previous discussion, we stated that a popular test of the proportional hazards assumption is to interact analysis time with the covariates (individually) and to verify that the interacted variables have no significant effect. Actually, any function of analysis time may be interacted with the covariates, and not all variables need to be interacted with the same function of time. It is all a matter of personal taste, and researchers often use $\ln(t)$ instead of t in the interactions. Doing this in Stata is easy enough—just change the specification of `texp()`. For example,

```
. use http://www.stata-press.com/data/hip2, clear
. stcox age protect, tvc(age) texp(ln(t))
```

In any case, such tests have been found to be sensitive to the choice of function, and there are no clear guidelines for choosing functions. A more complete approach would be to try t , $\ln(t)$ and then any other functions of time that occur to you.

A more flexible method, however, is to let the function of time be a step function (of a few steps) and let the magnitudes of the steps be determined by the data:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. stsplit cat, at(5 8 15)
(30 observations (episodes) created)
. gen age2 = age*(cat==5)
. gen age3 = age*(cat==8)
. gen age4 = age*(cat==15)
. stcox protect age age2-age4, noshow
Iteration 0:   log likelihood = -98.571254
Iteration 1:   log likelihood = -81.803706
Iteration 2:   log likelihood = -81.511169
Iteration 3:   log likelihood = -81.5101
Refining estimates:
Iteration 0:   log likelihood = -81.5101
```

Cox regression -- Breslow method for ties

```

No. of subjects =      48                Number of obs =      136
No. of failures =      31
Time at risk   =      714

Log likelihood =    -81.5101                LR chi2(6) =      34.12
                                           Prob > chi2 =      0.0000

```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
protect	.1096836	.0496909	-4.88	0.000	.0451351	.266544
age	1.17704	.0789269	2.43	0.015	1.032081	1.34236
age2	.937107	.1024538	-0.59	0.552	.7563579	1.16105
age3	.8677277	.0911832	-1.35	0.177	.7062149	1.066179
age4	.9477089	.0948454	-0.54	0.592	.7789111	1.153087

```
. testparm age2-age4
```

```

( 1) age2 = 0
( 2) age3 = 0
( 3) age4 = 0

      chi2( 3) =    1.83
      Prob > chi2 =    0.6079

```

This, we argue, is a better test. Time is allowed to have whatever effect it wants, although the cost is that the time function is only crudely defined over broad ranges of the data, meaning we misfit those broad ranges. In the context of what we are testing, however, that cost should be minimal. We do not need to find the best model, we just need to know if there is evidence as to the existence of another model that is better than our current one.

To summarize, the basis of specification tests is the consideration of models of the form

$$LRH = \mathbf{x}_j \beta_x + \beta_2 q_j$$

where q_j is something else. Under the assumption that $\mathbf{x}_j \beta_x$ is the correct specification, β_2 will be zero, and you can test for that. Since the choice of q_j is entirely up to the user, this forms a very large class of tests.

11.1.2 Test based on Schoenfeld residuals

Another way of checking the proportional hazards assumption (which is to say, specification) is based on analysis of residuals. This is not unlike re-estimation. The idea is to retrieve the residuals, fit a smooth function of time to them, and then test whether there is a relationship. Stata's `stphtest` command does this and is based on the generalization by Grambsch and Therneau (1994). They showed that many of the popular tests for proportional hazards are in fact a test of nonzero slope in a generalized linear regression of the scaled Schoenfeld (1982) residuals on functions of time.

In its simplest form, when there are no tied failure times, the Schoenfeld residual for covariate x_u , $u = 1, \dots, p$, and for observation j observed to fail is

$$r_{uj} = x_{uj} - \frac{\sum_{i \in R_j} x_{ui} \exp(\mathbf{x}_i \hat{\boldsymbol{\beta}}_x)}{\sum_{i \in R_j} \exp(\mathbf{x}_i \hat{\boldsymbol{\beta}}_x)}$$

That is, r_{uj} is the difference between the covariate value for the failed observation and the weighted average of the covariate values (weighted according to the estimated relative hazard from a Cox model) over all those subjects at risk of failure when subject j failed.

Let us momentarily allow that the coefficient on x_u does vary with time (in contradiction of the proportional hazards assumption), so that β_u is actually

$$\beta_u(t) = \beta_u + q_j g(t)$$

where q_j is just some coefficient and $g(t)$ is some function of time. Under the proportional hazards assumption, $q_j = 0$. Grambsch and Therneau (1994) provide a method of scaling the Schoenfeld residual to form r_{uj}^* , and the scaling provides that

$$E(r_{uj}^* + \beta_u) = \beta_u(t)$$

Consequently, a graph of r_{uj}^* versus t_j (or some function of t_j) provides an assessment of the proportional hazards assumption. Under the null hypothesis of proportional hazards, we expect the curve to have zero slope. In fact, r_{uj}^* versus t_j is fairly rough, so it is common to add to the graph some sort of smooth curve through the points and to base one's judgment on that. You can base your judgment on the graph or on a formal statistical test of $H_0 : q_j = 0$, the common test being based on a linear regression estimate of r_{uj}^* on t_j or $g(t_j)$. In the testing framework, the choice of $g(\cdot)$ is important. When you look at a graph, checking for a nonzero slope by inspection is sufficient. If you plan on performing the test after looking at a graph, choose a form of $g(\cdot)$ that transforms the relationship to a linear one, and then the test based on linear regression can detect a nonzero slope, given that the relationship is linear.

Stata's `stphtest` automates this process, and graphs and tests for individual covariates and globally the null hypothesis of zero slope. This test is equivalent to testing that the log hazard ratio function is constant over time. Thus, rejection of the null hypothesis indicates a deviation from the proportional hazards assumption. By default, `stphtest` sets $g(t) = t$, but you can change this by specifying the `time()` option; see `help stphtest` for details.

`stphtest` works after you have fit your model using `stcox`, but to use `stphtest`, you must specify both the `schoenfeld()` and `scaledsch()` options at the time you estimate using `stcox`. These options save the Schoenfeld and scaled Schoenfeld residuals (respectively), which are quantities required by `stphtest`. Actually, you need to specify `schoenfeld()` to perform the global test and `scaledsch()` if you want the variable-by-variable tests.

```
. use http://www.stata-press.com/data/cgg/hip2
. stcox protect age, schoenfeld(sch*) scaledsch(sca*)
(output omitted)
```

The effect of the above, in addition to fitting the model, is to create four new variables in the dataset: `sch1` and `sch2` are created by `schoenfeld(sch*)` and store the Schoenfeld residuals for variables `protect` and `age`, respectively; the variables `sca1` and `sca2` are created by `scaledsch(sca*)`, and store the scaled Schoenfeld residuals. You could have given any new variable names you wanted, but it is convenient to provide these options with a stub (`sch*`, for example) and to let Stata number the variables for you.

Having estimated the Schoenfeld and scaled Schoenfeld residuals, we may now use `stphtest`. The `detail` option reports the variable-by-variables tests along with the overall test:

```
. stphtest, detail
Test of proportional hazards assumption
Time: Time
```

	rho	chi2	df	Prob>chi2
protect	0.00889	0.00	1	0.9627
age	-0.11519	0.43	1	0.5140
global test		0.44	2	0.8043

We find no evidence that our specification violates the proportional hazards assumption.

Had we typed just `stphtest`, only the global (combined) test is reported:

```
. stphtest
Test of proportional hazards assumption
Time: Time
```

	chi2	df	Prob>chi2
global test	0.44	2	0.8043

Note that for this test we would only need the `schoenfeld()` option when we fit the model.

In addition, we can see each of the graphs by specifying `stphtest`'s `plot()` option,

```
. stphtest, plot(age)
```

which produces Figure 11.1. We could also see the graph for the other covariate in our model using `stphtest, plot(protect)`. Examining these graphs will help you to choose a transform of t that makes the curve roughly linear. This transform may then be used by `stphtest` to perform a formal statistical test.

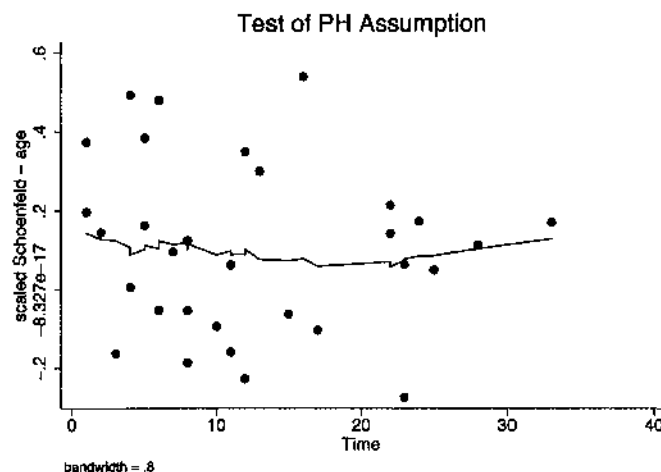


Figure 11.1: Test of the proportional hazards assumption for age

□ Technical Note

The Grambsch and Therneau tests of the proportional hazards assumption as implemented in Stata assume homogeneity of variance across risk sets. This allows the use of the estimated overall (pooled) variance–covariance matrix in the equations. Although these tests have been shown to be fairly robust to departures from this assumption, care must be exercised where this assumption may not hold, particularly when used with a stratified Cox model. In such cases, we recommend that the proportional hazards assumption be checked separately for each stratum.

□

11.1.3 Graphical methods

Of the many graphical methods proposed throughout the literature for assessing the proportionality of hazards, two are available in Stata: the commands `stphplot` and `stcoxkm`. Both are intended for use with discrete covariates. For a good introduction and review of graphical methods, see Hess (1995) and Garrett (1997).

`stphplot` plots an estimate of $-\ln[-\ln\{\hat{S}(t)\}]$ versus $\ln(t)$ for each level of the covariate in question, where $\hat{S}(t)$ is the Kaplan–Meier estimate of the survivor function. For proportional hazards, $h(t|\mathbf{x}) = h_0(t) \exp(\mathbf{x}\beta_x)$, and thus

$$S(t|\mathbf{x}) = S_0(t)^{\exp(\mathbf{x}\beta_x)}$$

which implies that

$$-\ln[-\ln\{S(t|\mathbf{x})\}] = -\ln[-\ln\{S_0(t)\}] - \mathbf{x}\beta_x$$

Under the proportional hazards assumption, the plotted curves should thus be parallel. For example,

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. stphplot, by(protect)
      failure _d: fracture
      analysis time _t: time1
      id: id
```

which produces Figure 11.2. We see that the curves are roughly parallel, providing evidence in favor of the proportional hazards assumption for the effect of wearing the hip-protection device.

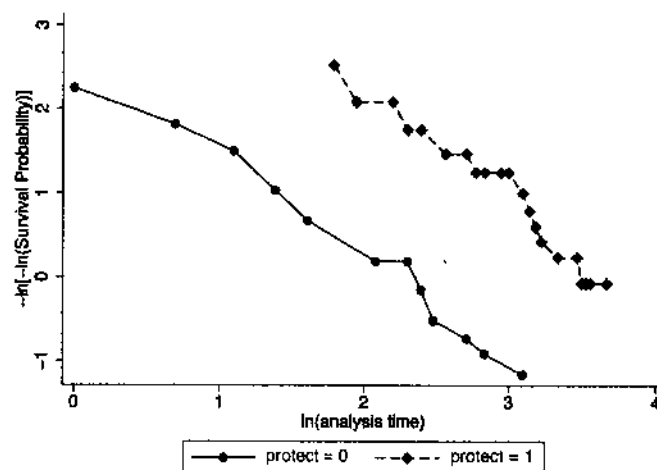


Figure 11.2: Test of proportional hazards assumption for protect

There is nothing magical being done by `stphplot` that we could not do ourselves. Basically, `stphplot` calculates a Kaplan-Meier curve $\hat{S}(t)$ for each level of the variable specified in `by()`, calculates for each curve the transformation $-\ln[-\ln\{\hat{S}(t)\}]$, and plots these curves with $\ln(t)$ on the x -axis.

The model for the hip-protection device we have been considering is

$$\text{LRH} = \beta_1 \text{protect} + \beta_2 \text{age}$$

Figure 11.2 does not address the question of whether the effect of `protect` is constant *conditional* on `age` being in the model. `stphplot` works differently from `stphtest` in that it does not make use of the `stcox` model that we might have previously fit. We can obtain the graph for `protect` conditional on `age` being in the model by

```
. stphplot, strata(protect) adjust(age)
```

which produces Figure 11.3.

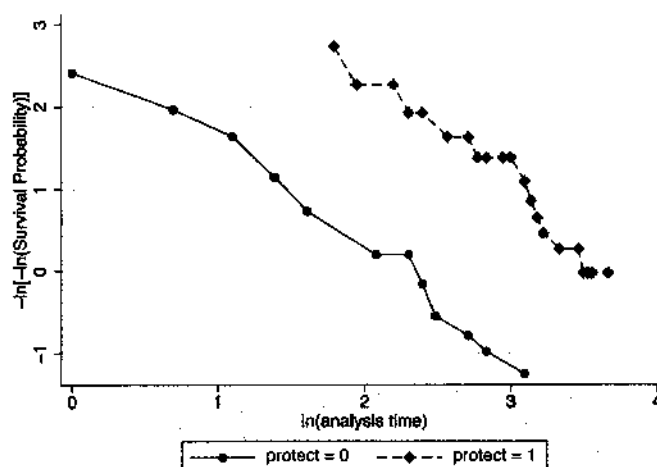


Figure 11.3: Test of proportional hazards assumption for `protect`, given `age`

In general, you should specify whatever other variables are in your model in the `adjust()` option. Specifying `adjust()` tells Stata not to estimate the survivor function using Kaplan–Meier, but instead to estimate it using `stcox`, and this is done so that we can introduce other variables into the model. In any case, the procedure followed by `stphplot`, `adjust()` `strata()` is as follows:

1. Fit a Cox model on the `adjust()` variables as covariates and on the `strata()` variable as a stratification variable. In this example, the command `stcox age, strata(protect)` is executed. Thus, the baseline hazard is allowed to be different for different values of `protect`, and it is left to the user to decide whether the hazards are really proportional.
2. Obtain the baseline survivor functions for each of the strata directly from `stcox`, and then calculate $-\ln[-\ln\{\hat{S}(t)\}]$ for each.
3. Plot the transformed (stratified) survivor curves on the same graph, with $\ln(t)$ on the x -axis.

Another method of graphically assessing proportional hazards is to compare separately estimated Kaplan–Meier curves (which are model agnostic) with estimates of $S_0(t)^{\exp(\mathbf{x}\beta_x)}$ from a Cox model, which does impose the model assumption of proportional hazards. `stcoxkm` does this,

```
. stcoxkm, by(protect)
```

which produces Figure 11.4.

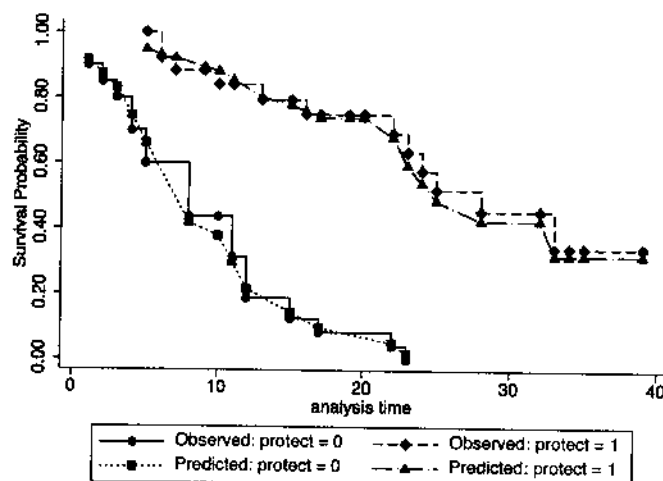


Figure 11.4: Comparison of Kaplan-Meier and Cox survivor functions

The way this graph is implemented in Stata, however, does not allow its use with other covariates in the model. Actually, the way this assessment has been developed in the literature does not allow its use when other covariates are included in the model, but generalizing it to that would be easy enough. Rather than using the Kaplan-Meier estimates to compare against, you could use the baseline survivor curve from a stratified Cox model.

11.2 Residuals

Although the uses of residuals vary and depend on the data and user preferences, some suggested traditional uses are as follows:

1. Cox-Snell residuals are useful in assessing overall model fit.
2. Martingale residuals are useful in determining the functional form of the covariates to be included in the model.
3. Schoenfeld residuals (both scaled and unscaled), score residuals, and efficient score residuals are useful for checking and testing the proportional hazard assumption, examining leverage points, and identifying outliers.
4. Deviance residuals are useful in examining model accuracy and identifying outliers.

Efficient score residuals, martingale residuals, Schoenfeld residuals, and scaled Schoenfeld residuals are obtained when we fit the Cox proportional hazard model via options to `stcox`. Cox-Snell and deviance residuals are obtained after estimation using `predict`.

When there are multiple observations per subject, cumulative martingale and cumulative Cox-Snell residuals are obtained after estimation using `predict`, and accumulation takes place over each subject's set of records.

Reye's syndrome data

We illustrate some of the possible applications of residuals using a study of 150 children diagnosed with Reye's syndrome.

Reye's syndrome is a rare disease, usually affecting children under the age of fifteen who are recovering from an upper respiratory illness, chicken pox, or flu. The condition causes severe brain swelling and inflammation of the liver. This acute illness requires immediate and aggressive medical attention. The earlier the disease is diagnosed, the better the chances of a successful recovery. Treatment protocols include drugs to control the brain swelling and intravenous fluids to restore normal blood chemistry.

For this study of a new medication to control the brain swelling, and thus to prevent death, 150 Reye's syndrome patients were randomly allocated at the time of hospital presentation to either the standard high-dose barbiturate treatment protocol or to a treatment protocol that included the new experimental drug. The time from treatment allocation to death or the end of follow-up was recorded in days. Here are a few of the records.

```
. use http://www.stata-press.com/data/cgg/reyes, clear
(Reye's syndrome data)
. list id days dead treat age sex ftliver ammonia sgot in 1/6, noobs sep(0)
```

id	days	dead	treat	age	sex	ftliver	ammonia	sgot
1	8	1	1	13	1	1	14.3	300
2	82	1	0	13	1	0	.9	287
3	19	1	0	16	1	1	1.2	298
6	46	0	1	15	1	0	.6	286
7	33	0	1	13	1	0	.8	270
9	44	1	0	10	1	0	3	280

The variable `treat` equals 0 for patients on the standard protocol and 1 for patients on the experimental protocol. The variable `days` indicates the number of follow-up days from treatment allocation to death (`death==1`) or to censoring (`death==0`). In addition to `age` and `sex`, several laboratory measurements including blood `ammonia` level and serum `sgot` were performed at the time of initial hospitalization. All patients involved also underwent a liver biopsy to assess the presence of fatty liver disease (`ftliver`) within 24 hours of treatment allocation. Note that all covariates in this study are fixed.

We are interested in determining whether the new experimental treatment is more effective in prolonging the life expectancy of these children. We begin addressing this question by `stset`ing our dataset and fitting a Cox proportional hazards model with the treatment variable only.

```

. stset days, id(id) failure(dead)
(output omitted)
. stcox treat, efron
      failure _d:  dead
      analysis time _t:  days
              id:  id
Iteration 0:   log likelihood = -253.90166
Iteration 1:   log likelihood = -252.86308
Iteration 2:   log likelihood = -252.86284
Refining estimates:
Iteration 0:   log likelihood = -252.86284
Cox regression -- Efron method for ties
No. of subjects =          150          Number of obs   =          150
No. of failures =           58
Time at risk   =          4971
Log likelihood =  -252.86284          LR chi2(1)       =          2.08
                                      Prob > chi2      =          0.1495

```

	_t _d	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
treat		.6814905	.1831707	-1.43	0.154	.4024151 1.154105

We observe that although there is some indication that the new treatment reduces the risk of dying when compared to the standard treatment (hazard ratio=0.68), there is not strong evidence suggesting that the new therapy is superior to the standard treatment. We could have arrived at a similar conclusion had we performed a log-rank test by typing `sts test treat`.

11.2.1 Determining functional form

Previous studies on children with Reye's syndrome have identified elevated ammonia and sgot levels along with the presence of fatty liver disease (`ftliver`) as important predictors of final outcome. Thus, in our analysis we would like to compare the two treatment protocols while adjusting for the effects of these covariates.

What is the best functional form to use for each covariate? We can use martingale residuals to help answer this question. Martingale residuals are obtained by specifying the `mgale()` option when fitting the Cox model. These residuals can be interpreted simply as the difference between the observed number of failures in the data and the number of failures predicted by the model.

Martingale residuals are also helpful in determining the proper functional form of the covariates to be included in the model. That is, should we include in the model the variable `age` directly, or should we perhaps include $\ln(\text{age})$ or some of other function of `age`?

Let M_i be the martingale residual of the i th observation obtained when no covariates are included in the model. Assume a true relative hazard model of the form

$$h(t|\mathbf{x}_i) = h_0(t) \exp\{f(\mathbf{x}_i)\}$$

where \mathbf{x}_i is the covariate vector for the i th subject and $f()$ is some function. It may then be shown that M_i is approximately $kf(\mathbf{x}_i)$, where k is a constant that depends on the number of censored observations. The above implies the existence of a linear relationship between $f(\mathbf{x}_i)$ and M_i . When used with a single predictor x , consequently a smooth plot of martingale residuals versus x may provide a visual indication of the transformation needed, if any, to obtain a proper functional form for x . The goal is to determine a transformation that will result in an approximately straight curve.

Returning to our data, we apply this method to determine the functional form for the `ammonia` variable. We begin by fitting a Cox model without covariates and obtaining the martingale residuals using the `mgale(newvarname)` option. We handle ties using Efron's method (for no real reason other than variety.) Recall that it is necessary to use the `estimate` option when fitting null models.

```
. stcox, mgale(mg) efron estimate
      failure _d:  dead
      analysis time _t:  days
              id:  id

Iteration 0:  log likelihood = -253.90166
Refining estimates:
Iteration 0:  log likelihood = -253.90166
Cox regression -- Efron method for ties

No. of subjects =          150          Number of obs =          150
No. of failures =           58
Time at risk   =          4971

Log likelihood = -253.90166          LR chi2(0) =          0.00
                                Prob > chi2 =          .
```

	<code>_t</code>				
	<code>_d</code>	Haz. Ratio	Std. Err.	z	P> z [95% Conf. Interval]

The newly created variable, `mg`, contains the values of the martingale residuals from this null model. We can now plot each variable whose functional form we are interested in determining against `mg`. We use the `lowess` command to obtain the graph along with a running mean smoother in order to ease interpretation. Alternatively, a `lowess` or other smoother could be used. We type

```
. lowess mg ammonia
```

which produces Figure 11.5.

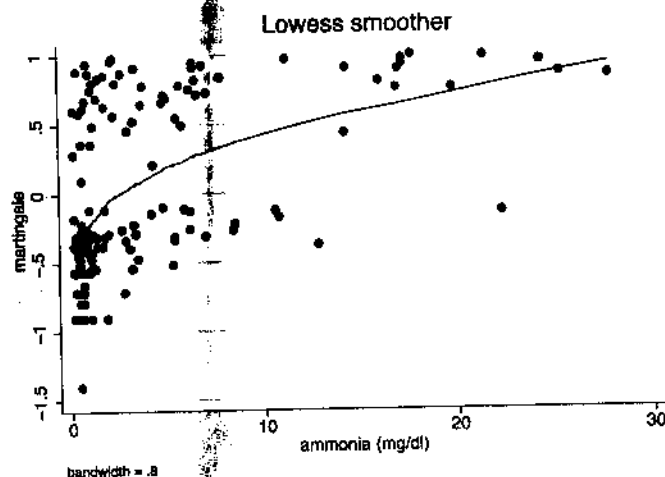


Figure 11.5: Finding the functional form for ammonia

We see that the resulting smooth plot is not linear at lower values of ammonia. In this case, a simple log transformation of the variable will yield a more linear result. We type

```
. gen lamm=ln(ammonia)
. lowess mg lamm
```

which produces Figure 11.6.

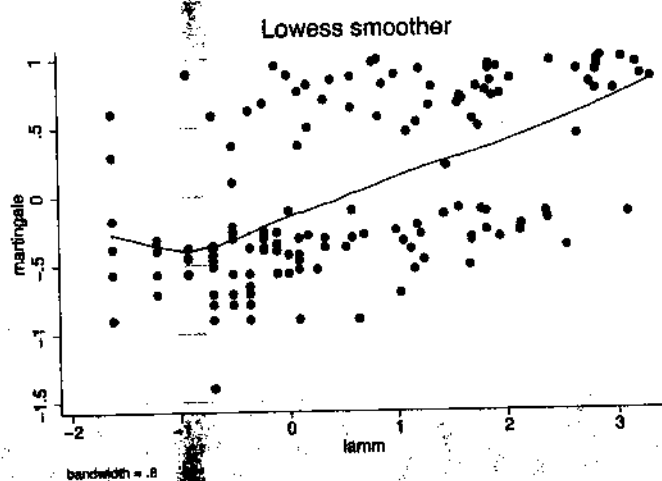


Figure 11.6: Using the log transformation

This yields an approximately linear smooth and leads us to believe that the log transformed ammonia variable is better for inclusion in the model.

We now fit the Cox model with the treatment and lamm covariates.

```
. stcox treat lamm, efron noshow
Iteration 0:  log likelihood = -253.90166
Iteration 1:  log likelihood = -222.27086
Iteration 2:  log likelihood = -221.77893
Iteration 3:  log likelihood = -221.77885
Refining estimates:
Iteration 0:  log likelihood = -221.77885
Cox regression -- Efron method for ties
No. of subjects =      150          Number of obs   =      150
No. of failures =       58
Time at risk   =     4971
Log likelihood = -221.77885          LR chi2(2)      =     64.25
                                      Prob > chi2     =     0.0000
```

	_t _d	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
treat		.5943764	.1656545	-1.87	0.062	.3442144 1.026347
lamm		2.546743	.3201795	7.44	0.000	1.99054 3.25836

We see that (as shown by other studies) increased levels of blood ammonia are strongly associated with increases in the hazard of dying. More importantly, however, we see that if we adjust by this important covariate, the treatment effect becomes significant at the 10% level. This indicates a lack of homogeneity of patients assigned to the treatment modalities. We can observe this heterogeneity by tabulating the mean ammonia level by treatment level and performing a *t*-test.

```
. ttest ammonia, by(treat) uneq
Two-sample t test with unequal variances
```

Group	Obs	Mean	Std. Err.	Std. Dev.	[95% Conf. Interval]
0	71	2.683099	.392177	3.304542	1.900926 3.465271
1	79	5.143038	.7920279	7.039698	3.566231 6.719844
combined	150	3.978667	.4661303	5.708907	3.057587 4.899746
diff		-2.459939	.8838049		-4.210858 -.7090201

Satterthwaite's degrees of freedom: 113.345

Ho: mean(0) - mean(1) = diff = 0

Ha: diff < 0	Ha: diff != 0	Ha: diff > 0
t = -2.7834	t = -2.7834	t = -2.7834
P < t = 0.0032	P > t = 0.0063	P > t = 0.9968

We find that on average, patients with lower ammonia levels ended up on the standard treatment, while those with higher ammonia levels, and therefore with a worse prognosis, were assigned to the experimental group. The fact that this imbalance exists can be significant in assessing the validity of this study and of the treatment allocation protocol.

We used this same method for examining the functional forms of the other covariates, and found it unnecessary to transform the other covariates. We chose, however, to center `sgot` at its mean so that the baseline hazard corresponds to `treat==0`, `lamm==0`, `ftliver==0`, and `sgot` equal to its mean value. This last part we did by typing

```
. quietly summarize sgot
. replace sgot = sgot - r(mean)
(150 real changes made)
```

Finally, we fit a Cox model using `treat`, `lamm`, `sgot` (now with 0 corresponding to its mean level), and `ftliver`.

```
. stcox treat lamm sgot ftliver, efron
      failure _d:  dead
      analysis time _t:  days
      id:  id

Iteration 0:  log likelihood = -253.90166
Iteration 1:  log likelihood = -219.18093
Iteration 2:  log likelihood = -215.1553
Iteration 3:  log likelihood = -215.14935
Refining estimates:
Iteration 0:  log likelihood = -215.14935

Cox regression -- Efron method for ties

No. of subjects =          150          Number of obs =          150
No. of failures =           58
Time at risk   =          4971

Log likelihood =  -215.14935          LR chi2(4) =          77.50
                                      Prob > chi2 =          0.0000
```

	_t _d	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
	treat	.4899586	.1439631	-2.43	0.015	.2754567	.8714962
	lamm	2.194971	.2831686	6.09	0.000	1.704678	2.826444
	sgot	1.049024	.0183381	2.74	0.006	1.01369	1.085589
	ftliver	2.14807	.6931377	2.37	0.018	1.141257	4.043091

We observe that after adjusting for ammonia, serum concentration, and the presence of fatty liver disease, the new therapy reduces the risk of death by 51%.

11.2.2 Goodness of fit

We now turn our attention to the evaluation of the overall model fit using Cox-Snell (Cox and Snell 1968) residuals. It has been shown that if the Cox regression model fits the data, then the true cumulative hazard function conditional on the covariate vector has an exponential distribution with a hazard rate of 1.

The Cox-Snell residual for the j th observation is defined as

$$CSr_j = \hat{H}_0(t_j) \exp(\mathbf{x}_j \hat{\boldsymbol{\beta}}_x) \quad (11.1)$$

where both $\hat{H}_0(t)$ and $\hat{\beta}_x$ are obtained from the Cox model fit. If the model fits the data well, then the Cox–Snell residuals should have a standard exponential distribution with hazard function equal to one for all t , and thus the cumulative hazard of the Cox–Snell residuals should be a straight 45° line. We can verify the model fit by estimating the empirical Nelson–Aalen cumulative hazard function, with the Cox–Snell residuals as the time variable along with the data’s original censoring variable.

We will fit two models. In the first model, we include the covariates `treat` and `ammonia`, and in the second model, we will substitute `lamm` for `ammonia`.

In Stata, Cox–Snell residuals are computed from the martingale residuals. Thus, when we fit the model using `stcox`, we save the martingale residuals and then use `predict` to obtain the Cox–Snell residuals. If we forget to save the martingale residuals at estimation, `predict` will refuse to calculate Cox–Snell residuals.

```
. use http://www.stata-press.com/data/cgg/reyes, clear
(Reye's syndrome data)
. stcox treat ammonia, efron mgale(mg)
      failure _d:  dead
analysis time _t:  days
              id:  id

Iteration 0:  log likelihood = -253.90166
Iteration 1:  log likelihood = -230.10661
Iteration 2:  log likelihood = -227.58788
Iteration 3:  log likelihood = -227.58783
Refining estimates:
Iteration 0:  log likelihood = -227.58783

Cox regression -- Efron method for ties

No. of subjects =          150          Number of obs   =          150
No. of failures =           58
Time at risk   =         4971

Log likelihood =  -227.58783          LR chi2(2)       =          62.63
                                      Prob > chi2      =          0.0000
```

	_t _d	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
treat		.4295976	.1300496	-2.79	0.005	.2373463 .777573
ammonia		1.171981	.0233525	7.96	0.000	1.127094 1.218657

Now we use `predict` to compute the Cox–Snell residuals and then `stset` the data using the Cox–Snell residuals as the time variable and our original failure variable, `dead`, as the failure indicator.

(Continued on next page)

```

. predict cs, csnell
. stset cs, fail(dead)
      failure event:  dead != 0 & dead < .
obs. time interval:  (0, cs]
exit on or before:  failure

```

```

150 total obs.
  0 exclusions

```

```

150 obs. remaining, representing
58 failures in single record/single failure data
58 total analysis time at risk, at risk from t =      0
      earliest observed entry t =      0
      last observed exit t = 2.902519

```

We can now use `sts generate` to generate the Nelson–Aalen cumulative hazard function and plot it against the Cox–Snell residuals,

```

. sts gen H = na
. line H cs cs, sort ytitle("") legend(cols(1))

```

which produces Figure 11.7.

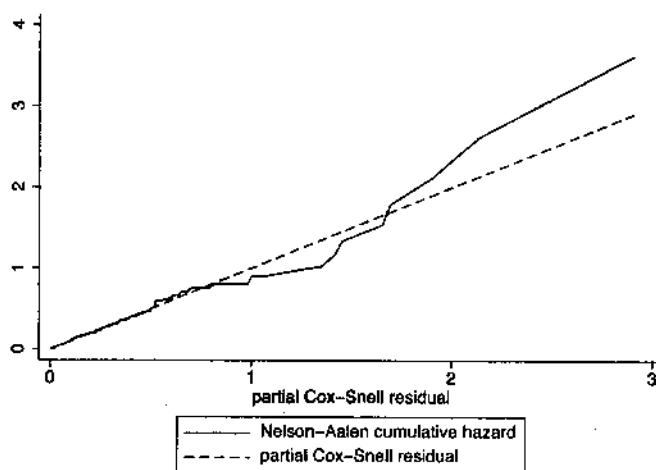


Figure 11.7: Cumulative hazard of Cox–Snell residuals (*ammonia*)

Note that in the `line` command we specified `cs` twice so as to get a 45° reference line. Comparing the jagged line to the reference line, we observe what could be considered a concerning lack of fit.

We repeat the above steps for the second model,

```

. use http://www.stata-press.com/data/cgg/reyes, clear
. gen lamm = ln(ammonia)

```

```
. stcox treat lamm, efron mgale(mg)
. predict cs, csnell
. stset cs, fail(dead)
. sts gen H = na
. line H cs cs, sort ytitle("") legend(cols(1))
```

which produces Figure 11.8.

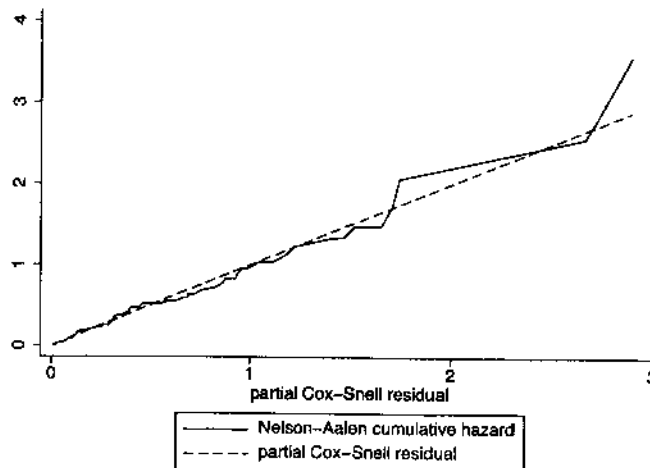


Figure 11.8: Cumulative hazard of Cox-Snell residuals (*lamm*)

When we use `lamm=ln(ammonia)` instead of `ammonia` in our model, the model does not fit the data too badly.

□ Technical Note

Note that when plotting the Nelson-Aalen cumulative hazard estimator for Cox-Snell residuals, even if we have a well-fitting Cox model, some variability about the 45° line is still expected, particularly in the right-hand tail. This is due to the reduced effective sample caused by prior failures and censoring.

□

11.2.3 Outliers and influential points

In evaluating the adequacy of the fitted model, it is important to determine if any one or any group of observations has a disproportionate influence on the estimated parameters. This is known as influence or leverage analysis. The preferred method of performing influence or leverage analysis is to compare the estimated parameter $\hat{\beta}_x$ obtained from the full data with the estimated parameters $\hat{\beta}_x^{(i)}$ obtained by fitting the model to the $n - 1$ observations remaining after the i th observation is removed. If $\hat{\beta}_x - \hat{\beta}_x^{(i)}$ is close

to zero, then the i th observation has little influence on the estimate. The process is repeated for all observations included in the original model.

To compute these differences for a dataset with n observations, we would have to execute `stcox` $n + 1$ times, which could be impractical for large datasets. In such cases, an approximation to the difference $\hat{\beta}_x - \hat{\beta}_x^{(i)}$ based on the efficient score residuals can be calculated as

$$DV(\hat{\beta}_x)$$

where $V(\hat{\beta}_x)$ is the variance-covariance matrix of $\hat{\beta}_x$ and D is the matrix of efficient score residuals. The difference $\hat{\beta}_x - \hat{\beta}_x^{(i)}$ is commonly referred to as “dfbeta” in the literature.

Stata saves efficient score residuals in variables generated at the time of estimation when you specify option `esr(newvarnames)`. 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. Therefore, if you want all the efficient score residuals, then you must either explicitly specify one new variable name for each regressor in your model or specify `esr(stub*)`, where `stub` is a name of your choosing. Stata then creates the variables `stub1`, `stub2`, etc.

□ Technical Note

Be careful when asking for efficient score residuals. Note that `stcox` may drop variables from the model due to collinearity. This is a desirable feature, yet a side-effect is that the score residual variables (and also the Schoenfeld residual variables and the scaled Schoenfeld residual variables) 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`, and `r3` the residual associated with `x3`. But assume that `x2` is dropped due to collinearity. In that case, `r1` will correspond to `x1`, as before, but `r2` will correspond to `x3` (and `r3` will contain 0). This happens because, after omitting the collinear variables, there are only two variables in the model: `x1` and `x3`.

□

Returning to the Reye’s syndrome data, we can obtain the efficient score residuals by typing

```
. use http://www.stata-press.com/data/cgg/reyes, clear
. gen lamm = ln(ammonia)
. quietly summarize sgot
. replace sgot = sgot - r(mean)
. stcox treat lamm sgot ftliver, efron esr(esr*)
```

This command creates four new variables: `esr1`, `esr2`, `esr3`, and `esr4`, corresponding to the four covariates in the model. We next use the matrix command `mkmat` to create a

matrix of the efficient score residuals, obtain the variance-covariance matrix, and then multiply the two matrices:

```
. mkmat esr1 esr2 esr3 esr4, matrix(esr)
. mat V = e(V)
. mat Inf = esr*V
. svmat Inf, names(s)
```

The last command saves the estimates of $\hat{\beta}_x - \hat{\beta}_x^{(i)}$ in the variables s1, s2, s3, and s4. We then label these new variables, just so that we do not forget their contents:

```
. label var s1 "dfbeta - treat"
. label var s2 "dfbeta - lamm"
. label var s3 "dfbeta - sgot"
. label var s4 "dfbeta - ftliver"
```

In any case, a graph of any of these variables versus time or observation numbers can be used to identify observations with disproportionate influence; for example,

```
. scatter s3 _t, yline(0) mlab(id) s(1)
```

which produces Figure 11.9.

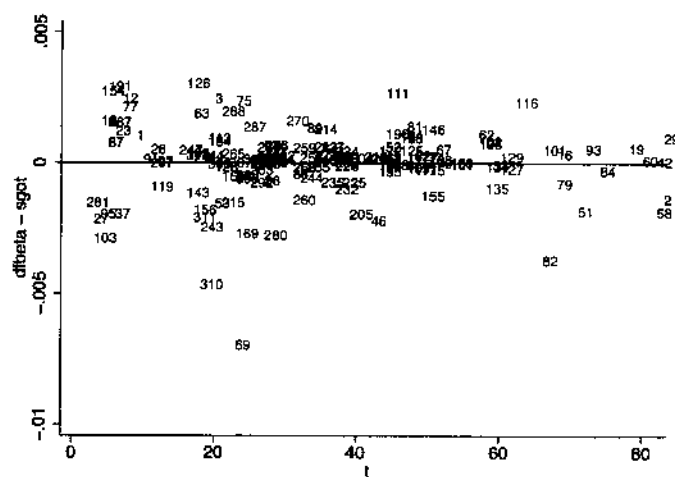


Figure 11.9: Dfbeta(sgot) for Reye's syndrome data



12 Parametric models

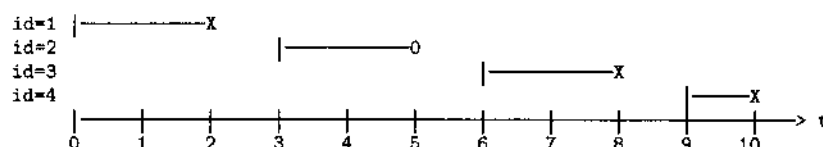
12.1 Motivation

Compared to the Cox model, parametric models are very different in the way they exploit the information contained in the data. As an extreme case, consider the following data:

```
. use http://www.stata-press.com/data/cgg/odd, clear
. list id _t0 _t female _d
```

	id	_t0	_t	female	_d
1.	1	0	2	0	1
2.	2	3	5	1	0
3.	3	6	8	0	1
4.	4	9	10	1	1

In these data, there are no overlaps in our observations of the subjects:



In the above diagram, t is analysis time, not calendar time. We observed subject 1 from $(0, 2]$, at which point the subject fails; we observe subject 2 from $(3, 5]$, at which point the subject is censored; and so on. What makes these data so odd is the fact that at any particular analysis time we observe at most one subject because most of our subjects arrived late to our study.

In any case, given these very odd data, the risk groups so important to semiparametric analysis are

1. Failure at time $t = 2$: Risk group contains only subject $id==1$.
 $\Pr(id == 1 \text{ fails} | \text{one failure}) = 1$.
2. Failure at time $t = 8$: Risk group contains only subject $id==3$.
 $\Pr(id == 3 \text{ fails} | \text{one failure}) = 1$.
3. Failure at time $t = 10$: Risk group contains only subject $id==4$.
 $\Pr(id == 4 \text{ fails} | \text{one failure}) = 1$.

and the conditional probability of observing what we did is one, regardless of any parameter we might want to estimate, because the probability that “our” subject fails given one subject fails is 1 if “our” subject is the only one at risk at that time. We could not, for instance, fit a Cox model on `female`:

```
. stcox female
      failure _d:  dead
      analysis time _t:  time1
Iteration 0:  log likelihood =          0
Refining estimates:
Iteration 0:  log likelihood =          0
Cox regression -- no ties
No. of subjects =          4          Number of obs =          4
No. of failures =          3
Time at risk =          7
Log likelihood =          0          LR chi2(1) =          0.00
                                Prob > chi2 =          1.0000
```

	_t _d	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
	female	1

Moreover, we could not even obtain a Kaplan–Meier estimate of the overall survivor function:

```
. sts list
      failure _d:  dead
      analysis time _t:  time1
```

Time	Beg. Total	Fail	Net Lost	Survivor Function	Std. Error	[95% Conf. Int.]
2	1	1	0	0.0000	.	.
3	0	0	-1	0.0000	.	.
5	1	0	1	0.0000	.	.
6	0	0	-1	0.0000	.	.
8	1	1	0	0.0000	.	.
9	0	0	-1	0.0000	.	.
10	1	1	0	0.0000	.	.

This extreme dataset has no information that nonparametric and semiparametric methods can exploit, namely, there is never more than one subject at risk at any particular time, which makes comparing subjects within risk sets difficult. Yet, parametric methods have no difficulty with these data:

```

. streg female, dist(exponential)
      failure _d:  dead
      analysis time _t:  time1

Iteration 0:  log likelihood = -.46671977
Iteration 1:  log likelihood = -.41031745
Iteration 2:  log likelihood = -.40973293
Iteration 3:  log likelihood = -.40973283

Exponential regression -- log relative-hazard form
No. of subjects =          4          Number of obs   =          4
No. of failures =          3
Time at risk    =          7

Log likelihood =  -.40973283          LR chi2(1)      =          0.11
                                      Prob > chi2     =          0.7357

```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
female	.6666667	.8164966	-0.33	0.741	.0604511 7.352135

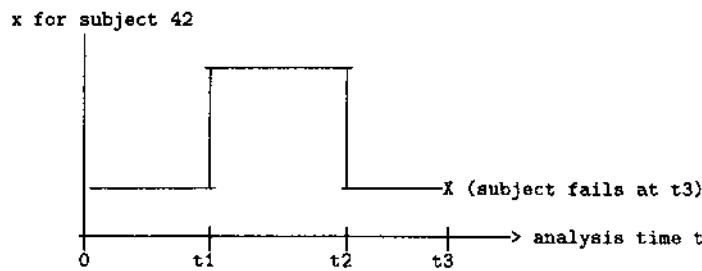
We began with this odd example not because it demonstrates the superiority of parametric methods; in fact, parametric methods do have their disadvantages. We used this example only to emphasize the very different way parametric methods exploit the information in the data to obtain estimates of the parameters.

Nonparametric and semiparametric methods compare subjects at the times when failures happen to occur, and to emphasize that, we have carefully concocted an example where no such comparison is possible.

Parametric methods, on the other hand, do not base their results on such comparisons. Rather, for each record in the data spanning $(t_{0j}, t_j]$, parametric estimation schemes use probabilities that depict what occurs over the whole interval, given what is known about the subject during this time (\mathbf{x}_j). For example, if it is observed that the subject was censored at time t_j , then the contribution to the likelihood for this record is

$$L_j = S(t_j|t_{0j}, \mathbf{x}_j) = \frac{S(t_j|\mathbf{x}_j)}{S(t_{0j}|\mathbf{x}_j)}$$

The implications of this difference in procedure are dramatic, but they are only matters of efficiency. For example, consider a survivor model where failure depends on a (possibly time-varying) covariate x . Pretend that one of the subjects in the dataset has the following x profile:



In semiparametric analysis (Cox regression), if no other subject fails between t_1 and t_2 , it simply does not matter that x blipped up for this subject because no comparisons will be made in that interval using the temporarily higher value of x . In other words, in this case, we would obtain the same Cox regression results if the blip in the time profile for this subject did not exist; i.e., if x remained at its initial value throughout.

The blip in x , however, would be of importance in a parametric model, regardless of whether other failures occurred in the interval because the parametric model would exploit all the information.

Cox regression is not making an error by ignoring the blip—it is merely being inefficient. Suppose that higher values of x increase failure rates. Conditional on having survived beyond time t_2 , the fact that the blip occurred becomes irrelevant in terms of subsequent survival. The information in the blip is that it indeed occurred and the subject managed to survive it, which means that this subject provides evidence that higher values of x really are not so bad. Cox regression would ignore that unless other failures occurred in the interval, in which case some amount of the information contained in the interval would be exploited in improving the estimate of the effect of x . Parametric methods would ignore no part of that information.

12.2 Classes of parametric models

Parametric models are written in variety of ways. As we stated in Chapter 1, linear regression is an example of a survival model,

$$t_j = \mathbf{x}_j \beta_x + \epsilon_j \quad \epsilon_j \sim N(0, \sigma^2)$$

Other models are written in the log-time metric (also known as the accelerated time metric),

$$\ln(t_j) = \mathbf{x}_j \beta_x + \epsilon_j \quad \epsilon_j \sim \text{oddlly, but not odd given the context}$$

and others in the hazard metric,

$$h(t|\mathbf{x}_j) = h_0(t) \exp(\mathbf{x}_j \beta_x), \quad h_0(t) = \text{some functional form}$$

and, in fact, you can imagine parametric models written in other ways. Parametric models are written in the way natural to the parameterization. Some models may be written in more than one way because they fit equally well into more than one style of thinking. It is important to understand that, in that case, it is still the same model and it is just being expressed differently. Stata can fit the following parametric models:

1. Time parameterization:
 - a. linear regression
`regress, cnreg, intreg`
2. Log-time parameterization:
 - a. exponential
`streg, dist(exponential) time`
 - b. Weibull
`streg, dist(weibull) time`
 - c. log-normal
`streg, dist(lognormal)`
 - d. log-logistic
`streg, dist(loglogistic)`
 - e. gamma
`streg, dist(gamma)`
3. hazard parameterization:
 - a. exponential
`streg, dist(exponential)`
 - b. Weibull
`streg, dist(weibull)`
 - c. Gompertz
`streg, dist(gompertz)`

With the exception of the linear-regression time-parameterization models (which we do not discuss further), parametric models in Stata are fit using the `streg` command, and options control the particular parameterization used. As with `stcox`, you must `stset` your data prior to using `streg`. As stated previously, some models fit into more than one parameterization, but even so, they are still the same model.

12.2.1 Parametric proportional hazards models

Proportional hazards models are written

$$h(t|\mathbf{x}_j) = h_0(t) \exp(\mathbf{x}_j\boldsymbol{\beta}_x)$$

and, in the Cox model, $h_0(t)$ was simply left unparameterized, and through conditioning on failure times, estimates of β_x were obtained anyway. In the parametric approach, a functional form for $h_0(t)$ is specified. For example, if we assume $h_0(t) = \exp(a)$, for some a , then we have the exponential model. The baseline hazard is assumed constant over time, and there is an extra parameter (a) to estimate. When we fit this model, we are just estimating (a, β_x') .

If we assume

$$h_0(t) = pt^{p-1} \exp(a)$$

then we have the Weibull model. This model has two ancillary parameters, a and p . When we fit this model, we are estimating (a, p, β_x') .

If we assume

$$h_0(t) = \exp(a) \exp(\gamma t)$$

then we obtain the Gompertz model, another model with two ancillary parameters, this time a and γ . Estimated from that data is (a, γ, β_x') .

There are other possibilities, and, of course, you could make up your own. Whatever function is chosen for $h_0(t)$ is supposed to parameterize adequately what the baseline hazard really is.

In any case, all of these models produce results that are directly comparable to those produced by Cox regression. In all these models, $x\beta_x$ is the log relative hazard and the elements of β_x have the standard interpretation, meaning that $\exp(\beta_i)$ is the hazard ratio for the i th coefficient. In addition, parametric models produce estimates of the ancillary parameters, and from that you can obtain the predicted baseline hazard function $h_0(t)$ and any of the other related functions, such as the cumulative hazard, survivor function, etc.

The (direct) comparability to Cox regression is probably the most appealing feature of the parametric proportional hazards model. When you engage in this kind of parametric estimation, it is prudent to compare the estimated coefficients $\hat{\beta}_x$ to those from a Cox model fit, in order to verify that they are roughly similar. If they prove not to be similar, then this is evidence of a misparameterized underlying baseline hazard.

Stata makes this comparison easy because, by default, `streg` reports hazard ratios (exponentiated coefficients) when fitting models in this metric. For example, in our discussion of Cox regression we repeatedly examined the following model:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
```



```
. stcox protect age
      failure _d: fracture
      analysis time _t: time1
      id: id
Iteration 0:   log likelihood = -98.571254
Iteration 1:   log likelihood = -82.735029
Iteration 2:   log likelihood = -82.471037
Iteration 3:   log likelihood = -82.470259
Refining estimates:
Iteration 0:   log likelihood = -82.470259
Cox regression -- Breslow method for ties
No. of subjects =          48          Number of obs   =          106
No. of failures =          31
Time at risk    =          714
Log likelihood  = -82.470259          LR chi2(2)       =          32.20
                                          Prob > chi2      =          0.0000
```

_t _d	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
protect	.1046812	.0475109	-4.97	0.000	.043007	.2547989
age	1.110972	.0420079	2.78	0.005	1.031615	1.196434

If we now fit this same model using exponential regression (meaning that we assume the baseline hazard $h_0(t)$ is constant), we would obtain

```
. streg protect age, dist(exponential)
      failure _d: fracture
      analysis time _t: time1
      id: id
Iteration 0:   log likelihood = -60.067085
Iteration 1:   log likelihood = -54.034598
Iteration 2:   log likelihood = -47.563588
Iteration 3:   log likelihood = -47.534671
Iteration 4:   log likelihood = -47.534656
Iteration 5:   log likelihood = -47.534656
Exponential regression -- log relative-hazard form
No. of subjects =          48          Number of obs   =          106
No. of failures =          31
Time at risk    =          714
Log likelihood  = -47.534656          LR chi2(2)       =          25.06
                                          Prob > chi2      =          0.0000
```

_t _d	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
protect	.1847118	.0684054	-4.56	0.000	.0893849	.3817025
age	1.084334	.0371696	2.36	0.018	1.013877	1.159688

Note how different is the estimated hazard ratio for protect in the two models: `stcox` reports 0.105, `streg` reports 0.185. This inconsistency points out the inadequacy of a model that assumes a constant baseline hazard, and in fact, when we previously

considered this Cox model, we noted that the estimated cumulative hazard seemed to be increasing at an increasing rate. All of which means that the assumption of constant hazard is probably incorrect in this case, and we would obtain better estimates by choosing a parameterization for $h_0(t)$ that would allow it to grow.

The Weibull model will do that, and using that distribution, we would obtain

```
. streg protect age, dist(weibull)
      failure _d: fracture
      analysis time _t: time1
      id: id

Fitting constant-only model:
Iteration 0:  log likelihood = -60.067085
Iteration 1:  log likelihood = -59.30148
Iteration 2:  log likelihood = -59.298481
Iteration 3:  log likelihood = -59.298481

Fitting full model:
Iteration 0:  log likelihood = -59.298481
Iteration 1:  log likelihood = -54.887563
Iteration 2:  log likelihood = -42.123875
Iteration 3:  log likelihood = -41.993012
Iteration 4:  log likelihood = -41.992704
Iteration 5:  log likelihood = -41.992704

Weibull regression -- log relative-hazard form
No. of subjects =      48      Number of obs   =      106
No. of failures =      31
Time at risk    =      714
Log likelihood   = -41.992704      LR chi2(2)      =      34.61
                                      Prob > chi2      =      0.0000
```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
protect	.1099611	.0448214	-5.42	0.000	.0494629	.2444548
age	1.117186	.0423116	2.93	0.003	1.03726	1.203271
/ln_p	.5186694	.1376486	3.77	0.000	.2490831	.7886556
p	1.680127	.2312671			1.282849	2.200436
1/p	.5951931	.0819275			.4544553	.7795152

and we now obtain an estimate of the hazard ratio for protect of 0.110, which is more in line with the Cox regression result.

If you are looking for a parameterization of $h_0(t)$ that has considerable flexibility and has no restrictions on the shape of the hazard that you want to impose upon the model, we suggest you stay with Cox regression. That is what Cox regression does, and it does that very well. Parametric estimation is appropriate when you do have an idea of what the baseline hazard looks like and you want to impose that idea in order to (1) obtain the most efficient estimates of β_x possible and (2) to obtain an estimate of $h_0(t)$ subject to that constraint.

We mentioned above that Stata makes comparing Cox to parametric estimation easy because `streg` reports hazard ratios when fitting models that have a natural proportional hazards parameterization (exponential, Weibull, Gompertz). Parametric models, as we have suggested, sometimes have more than one parameterization. The Weibull model is an example of this, and a popular parameterization of the Weibull is that which carries the accelerated time interpretation. Stata can also report results in this alternate metric (obtained by specifying the `time` option):

```
. streg protect age, dist(weibull) time
      failure _d: fracture
      analysis time _t: time1
      id: id

Fitting constant-only model:
Iteration 0:  log likelihood = -60.067085
Iteration 1:  log likelihood = -59.30148
Iteration 2:  log likelihood = -59.298481
Iteration 3:  log likelihood = -59.298481

Fitting full model:
Iteration 0:  log likelihood = -59.298481
Iteration 1:  log likelihood = -54.887563
Iteration 2:  log likelihood = -42.123875
Iteration 3:  log likelihood = -41.993012
Iteration 4:  log likelihood = -41.992704
Iteration 5:  log likelihood = -41.992704

Weibull regression -- accelerated failure-time form
No. of subjects =      48                Number of obs =      106
No. of failures =      31
Time at risk   =     714

Log likelihood = -41.992704                LR chi2(2)      =     34.61
                                           Prob > chi2      =     0.0000
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
protect	1.313965	.2366229	5.55	0.000	.8501928	1.777737
age	-.0659554	.0221171	-2.98	0.003	-.1093041	-.0226067
_cons	6.946524	1.575708	4.41	0.000	3.858192	10.03486
/ln_p	.5188694	.1376486	3.77	0.000	.2490831	.7886556
p	1.680127	.2312671			1.282849	2.200436
1/p	.5951931	.0819275			.4544553	.7795152

These results look nothing like the results reported by `stcox`, and there is no reason that they should. These results are reported in a different metric, but it is merely a different way of reporting the same information. In fact, the above coefficients may be transformed back into the proportional hazards metric using the relationship $HR = \exp(-\hat{p}\hat{\beta}_{AFT})$, where HR is a single hazard ratio reported in the proportional hazards metric and $\hat{\beta}_{AFT}$ is the corresponding regression coefficient from an accelerated time model.

However, we do not expect you to remember that relationship, which is unique to the Weibull. If a model can be cast in the proportional hazards metric, then `streg` can report results in that metric, and those results are directly comparable to those produced by `stcox`. We mention this to emphasize that, even if you find it otherwise desirable to view and think about these results in the accelerated time metric, you can still compare the results reported in the hazard metric to those of `stcox`, and then, after convincing yourself that these are reasonable results, return to the accelerated time metric. The models are the same—they just look different.

□ Technical Note

When fitting models that have both a proportional hazards and an accelerated time parameterization, switching from one parameterization to the other requires refitting the entire model. In other words, using `streg` in replay mode will not suffice. Although both models are equivalent and transformation from one form to the other is possible post-estimation, there are many aspects of Stata's post-estimation commands (such as `test`, `predict`, etc.) that are tailored to one particular parameterization. As such, re-estimation is required. View this as a shortcoming of Stata, not that the models are different in any substantive way.

□

Returning to our discussion of the proportional hazards metric, recall that `stcox` by default reported hazard ratios (exponentiated coefficients) but that, if you specified the `nohr` option, it would report the coefficients themselves. `streg` works the same way, and you can specify `nohr` when fitting the model or when you redisplay results. For example, when we fit our exponential model in the proportional hazards metric, by default we displayed hazard ratios. If we specify the `nohr` upon replay, we get the coefficients themselves:

```
. streg protect age, dist(exponential)
(output omitted)
. streg, nohr
Exponential regression -- log relative-hazard form
No. of subjects =      48          Number of obs =      106
No. of failures =      31
Time at risk   =      714
Log likelihood = -47.534656          LR chi2(2)      =      25.06
                                      Prob > chi2     =      0.0000
```

_t	Coef.	Std. Err.	z	P> z	{95% Conf. Interval}
protect	-1.688958	.3703357	-4.56	0.000	-2.414803 - .9631137
age	.0809663	.0342787	2.36	0.018	.0137813 .1481514
_cons	-7.892737	2.458841	-3.21	0.001	-12.71198 -3.073498

Note that when comparing this output to the previous one from exponential regression, $\exp(-1.688958) = 0.1847$, the reported hazard ratio for `protect`.

We see one thing, however, that is new: `streg` reported a coefficient for `_cons`, something to which `stcox` has no counterpart. The coefficient on `_cons` has to do with the estimation of $h_0(t)$. In particular, the exponential model is

$$h(t|\mathbf{x}_j) = h_0(t) \exp(\mathbf{x}_j \boldsymbol{\beta}_x)$$

where $h_0(t) = \exp(a)$, meaning that

$$\begin{aligned} h(t|\mathbf{x}_j) &= \exp(a + \mathbf{x}_j \boldsymbol{\beta}_x) \\ &\equiv \exp(\beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x) \end{aligned}$$

and that is where we get our intercept term `_cons`. The intercept in the exponential model has to do with the level of the baseline hazard function, and in this case, we estimate $h_0(t) = \exp(-7.892737)$. This is a very small hazard, but remember, just as with `stcox`, the baseline hazard is a reflection of the hazard when all covariates equal zero. In particular, for our model `age==0` is an absurd value. Just as with `stcox`, if you want to normalize the baseline hazard to reflect the hazard for reasonable values of the covariates, adjust the covariates so that the zeros are themselves reasonable:

```
. gen age60 = age-60
. streg protect age60, dist(exponential) nohr noshow

Iteration 0:  log likelihood = -50.067085
Iteration 1:  log likelihood = -54.034598
Iteration 2:  log likelihood = -47.553588
Iteration 3:  log likelihood = -47.534671
Iteration 4:  log likelihood = -47.534656
Iteration 5:  log likelihood = -47.534656

Exponential regression -- log relative-hazard form
No. of subjects =          48          Number of obs   =          106
No. of failures =           31
Time at risk    =          714

Log likelihood = -47.534656          LR chi2(2)        =          25.06
                                      Prob > chi2         =          0.0000
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
protect	-1.688958	.3703357	-4.56	0.000	-2.414803 - .9631137
age60	.0809663	.0342787	2.36	0.018	.0137813 .1481514
_cons	-3.034758	.4536734	-6.69	0.000	-3.923942 -2.145675

The baseline hazard is now a more reasonable $\exp(-3.034758) = 0.048$ per month.

12.2.2 Accelerated failure-time models

Accelerated failure-time models, also known as accelerated time models or $\ln(\text{time})$ models, follow the parameterization

$$\ln(t_j) = \mathbf{x}_j \boldsymbol{\beta}_x + \epsilon_j \quad \epsilon_j \sim \text{oddlly, but not odd given the context}$$

The word “accelerated” is used in describing these models because, rather than assuming that failure time t_j is exponential, Weibull, or some other form, a distribution is instead assumed for

$$\tau_j = \exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j$$

and $\exp(-\mathbf{x}_j\boldsymbol{\beta}_x)$ is called the acceleration parameter.

- if $\exp(-\mathbf{x}_j\boldsymbol{\beta}_x) = 1$, then $\tau_j = t_j$, and time passes at its “normal” rate.
- if $\exp(-\mathbf{x}_j\boldsymbol{\beta}_x) > 1$, then time passes more quickly for the subject (time is accelerated), and so failure would be expected to occur sooner.
- if $\exp(-\mathbf{x}_j\boldsymbol{\beta}_x) < 1$, then time passes more slowly for the subject (time is decelerated), and so failure would be expected to occur later.

The derivation of these models is straightforward. If $\tau_j = \exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j$, then $t_j = \exp(\mathbf{x}_j\boldsymbol{\beta}_x)\tau_j$, and

$$\ln(t_j) = \mathbf{x}_j\boldsymbol{\beta}_x + \ln(\tau_j) \quad (12.1)$$

The random quantity $\ln(\tau_j)$ has a distribution determined by what is assumed about the distribution of τ_j , and in the usual nomenclature of these models, it is the distribution of τ_j that is specified. For example, in a log-normal model, τ_j follows a log-normal distribution, which implies that $\ln(\tau_j)$ follows a normal distribution, which makes (12.1) analogous to linear regression.

□ Technical Note

In comparing our derivation of this model to that found elsewhere, you may find that other sources report the result

$$\ln(t_j) = -\mathbf{x}_j\boldsymbol{\beta}_x + \ln(\tau_j)$$

rather than (12.1), but that is only because they started with the assumption that $\tau_j = \exp(\mathbf{x}_j\boldsymbol{\beta}_x)t_j$ rather than $\tau_j = \exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j$. The sign flip is of no importance. If one really thought in the accelerated time metric, it would indeed be more natural to follow this alternate specification because in our developments, it is $-\boldsymbol{\beta}_x$ that corresponds to how time is accelerated.

However, the use of (12.1) is justified on the grounds of predicted time. A positive coefficient in $\boldsymbol{\beta}_x$ serves to increase the expected value of $\ln(\text{time to failure})$. This is the view we chose to accept at the outset, and thus we use $\tau_j = \exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j$. □

There is a second sense in which these models are “accelerated”. The effect of a change in one of the \mathbf{x} variables, measured in time units, increases with t . For example,

pretend that $\beta_1 = 0.5$, meaning that a one-unit increase in x_1 increases the expected value of $\ln(t)$ by 0.5. For a subject predicted to fail at $t = 1$, this one unit increase would delay the predicted time of failure to $\exp\{\ln(1) + 0.5\} = 1.65$. For a subject predicted to fail at $t = 5$, this one-unit increase would delay the predicted time of failure to 8.24. That is, the marginal effect of x_1 accelerates. For larger t , we expect a longer delay in failure due to a one-unit increase in x_1 .

In the accelerated time metric, exponentiated coefficients have the interpretation of time ratios for a one-unit change in the corresponding covariate. For a subject with covariate values $\mathbf{x} = (x_1, x_2, \dots, x_k)$,

$$t_j = \exp(\beta_1 x_1 + \beta_2 x_2 + \dots + \beta_k x_k) \tau_j$$

If the subject had x_1 increased by 1, then

$$t_j^* = \exp\{\beta_1(x_1 + 1) + \beta_2 x_2 + \dots + \beta_k x_k\} \tau_j$$

and the ratio of t_j^* to t_j is $\exp(\beta_1)$.

By default, **streg**, when used to fit AFT (accelerated failure time) models, reports coefficients and not exponentiated coefficients, but you can specify the **tr** option either when you estimate or when you redisplay results to see results in terms of time ratios. The default is surprising given that, by default, **streg**, when used to fit PH (proportional hazards) models, reports exponentiated coefficients, but that is because most authors do not even mention the time-ratio interpretation of exponentiated coefficients. However, due to the ease of interpretation, specifying option **tr** and reporting time ratios is recommended.

12.2.3 Comparing the two parameterizations

The PH (proportional hazards) metric is used mainly as an analog to Cox regression, when the researcher wishes to gain insight into the actual risk process (the hazard function) that causes failure and to gain insight into how the risk changes with the values of covariates in the model. As with Cox regression, little attention is paid to the actual failure times, and predictions of these failure times are seldom desired.

The AFT (accelerated failure time) metric, however, gives a more prominent role to analysis time. Recall that the typical AFT model is of the form

$$\ln(t_j) = \mathbf{x}_j \boldsymbol{\beta}_x + \ln(\tau_j)$$

and by specifying a model in this form, one basically is asserting an interest in what happens to $E\{\ln(t_j) | \mathbf{x}_j\}$ for different values of \mathbf{x}_j . With such an interest usually comes a desire to predict either failure time or the logarithm of failure time, and there are instances when such predictions can prove problematic.

One difficulty with predicting time to failure has to do with time-varying covariates. Time-varying covariates cause no theoretical difficulty in accelerated time models, nor

are they difficult to fit (in fact, Stata does all the work). The problem occurs when one goes back to construct predictions.

Recall that in our hip-fracture data we have a time-varying covariate calcium. It turned out not be significant in a Cox model, and the same turns out to be true if we estimate using a Weibull model. However, if we omit age from the model, we can get calcium to be significant in order to allow us to make our point about prediction with time-varying covariates. We begin by fitting the model:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect calcium, time dist(weibull)
      failure _d: fracture
      analysis time _t: time1
                  id: id

Fitting constant-only model:
Iteration 0:  log likelihood = -60.067085
Iteration 1:  log likelihood = -59.30148
Iteration 2:  log likelihood = -59.298481
Iteration 3:  log likelihood = -59.298481

Fitting full model:
Iteration 0:  log likelihood = -59.298481
Iteration 1:  log likelihood = -54.764667
Iteration 2:  log likelihood = -42.840111
Iteration 3:  log likelihood = -42.728013
Iteration 4:  log likelihood = -42.727796
Iteration 5:  log likelihood = -42.727796

Weibull regression -- accelerated failure-time form
No. of subjects =          48          Number of obs   =          106
No. of failures =          31
Time at risk    =          714
Log likelihood  = -42.727796          LR chi2(2)       =          33.14
                                      Prob > chi2      =          0.0000
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
protect	1.300919	.241717	5.38	0.000	.8271626 1.774676
calcium	.238073	.0904338	2.63	0.008	.060826 .4153199
_cons	-.0700465	.9016456	-0.08	0.938	-1.837239 1.697146
/ln_p	.4849596	.1363349	3.56	0.000	.2177481 .752171
p	1.624109	.2214228			1.243274 2.121601
1/p	.6157221	.0839444			.4713421 .804328

This model has been fit in the AFT metric. Thus, the model states that

$$\ln(t_j) = -0.07 + 1.30\text{protect}_j + 0.24\text{calcium}_j + \ln(\tau_j)$$

The estimate $\hat{p} = 1.624$ has to do with the distribution of $\ln(\tau)$ and need not concern us right now.

We now obtain the predicted times to failure (which we choose to be the conditional mean given the covariates), something easy enough to do in Stata

```
. predict t_hat, mean time
```

Let's look at the predictions for subject `id==10`, one of our subjects who has time-varying covariate `calcium`.

```
. list id _t0 _t protect calcium _d t_hat if id==10
```

	id	_t0	_t	protect	calcium	_d	t_hat
11.	10	0	5	0	9.69	0	8.384427
12.	10	5	8	0	9.47	0	7.956587

Subject 10 was observed over the period $(0, 8]$ and then was censored; at time 0 and 5, we had measurements of `calcium` on her. The variable `t_hat` contains the predicted (mean) failure times.

The interpretation of `t_hat` in the first observation is that if we had a subject who had `protect==0` and `calcium==9.69`, and those covariates were fixed, the predicted time of failure would be 8.38. In the second observation, the prediction is that if we had a subject who was under continual observation from time 0 forward, and if the subject had fixed covariates `protect==0` and `calcium==9.47`, then the predicted time of failure would be 7.96.

Our point is that neither of those predictions really has to do with subject 10. To obtain the predicted time to failure for a subject under continual observation whose value of `calcium` starts at 9.69 and then changes to 9.47 at $t = 5$ is a truly miserable calculation, and Stata has no automated way to calculate this for you. If you really wanted it, here is what you would have to do:

1. Start by translating results back to the hazard metric.
2. Write down the hazard function $h(t)$ for subject 10. It will be a function of the estimated parameters, and it will have a discontinuity at time 5.
3. Integrate $h(t)$ to obtain the cumulative hazard $H(t)$.
4. Obtain the density function of t , $f(t) = h(t) \exp\{-H(t)\}$.
5. Having $f(t)$, now obtain its expected value.

As we stated, it would be a lot of work. Even after you had that, you would have to think carefully if that is what you want. Do you really want to assume that `calcium` changes from 9.69 to 9.47 at $t = 5$, or would you also like to add that it continues to decline, perhaps declining to 9.25 at time 10, and so on? Changing the future profile of the variable will change the expected value.

Thus, time can be a difficult metric in which to interpret regression results. Note, however, that the above difficulties are not the fault of the AFT metric—we would

have the same problem with a PH model—it is just that AFT models tend to place the emphasis on time, and with that emphasis, the above predictions become more desirable.

For this reason, it is often preferable to think of AFT models not as linear models on $\ln(\text{time})$, but instead to use something analogous to the PH interpretation. Namely, for an AFT model we can think in terms of survivor functions

$$S(t|\mathbf{x}) = S_0\{\exp(-\mathbf{x}\beta_x)t\}$$

That is, the probability of survival past time t for an individual with covariates \mathbf{x} is equivalent to the probability of survival past time $\exp(-\mathbf{x}\beta_x)t$ for an individual at the baseline (one with covariate values all equal to zero).

As stated earlier, some AFT models, namely the exponential and Weibull, have both a hazard interpretation and an accelerated time interpretation. The other AFT models that Stata can fit—log-normal, log-logistic, and gamma—have no natural PH interpretation. This is not to deny that you could work out the hazard function corresponding to these models, but that function would not be a simple function of the regression coefficients and in no sense would be easy to interpret.

13 A survey of parametric regression models in Stata

The parametric regression models in Stata work like the Cox model in that they can handle all the same problems, which is to say, time-varying covariates, delayed entry, gaps, and right censoring. If your dataset is appropriate for use with `stcox`, it is ready for use with `streg`.

Datawise, there is one important difference that you will need to consider. In Cox regression, the definition of `origin()`—the definition of when analysis time $t = 0$ —plays no real role, whereas in parametric models, it can be (and usually is) of vital importance.

The only role `origin()` plays in `stcox` is that Stata applies the rule that subjects are not and cannot be at risk prior to $t = 0$. Putting that aside, the definition of the origin really does not matter. That is, if you took a dataset, ran `stcox`, and then added 50 to all time variables and reran the analysis, nothing would be changed. The only role played by time in the Cox model is to determine who is to be compared to whom, and the magnitude of the time variable matters not at all.

Try the same experiment with a parametric model—add 50 to the time variables— and results will change, unless you are fitting an exponential regression model. In parametric models, time plays a real role, and the point at which analysis time $t = 0$ determines when risk begins accumulating. Adding 50 to all the time variables changes that accumulated risk in a nonproportional way. You need to think about when the onset of risk really is.

Both the Cox model and parametric models are invariant to multiplicative transforms of time, which is to say, it makes no difference whether you measure time in minutes, hours, days, or years. It actually is not a theoretical constraint on parametric models that they be scale invariant, but all the standard models are because, were they not, they would hardly be reasonable or useful.

The likelihood functions of the parametric models—regardless of the particular one under consideration—all follow the same general form:

$$L_j(\beta_x, \Theta) = \frac{\{S(t_j | \mathbf{x}_j \beta_x, \Theta)\}^{1-d_j} \{f(t_j | \mathbf{x}_j \beta_x, \Theta)\}^{d_j}}{S(t_{0j} | \mathbf{x}_j \beta_x, \Theta)} \quad (13.1)$$

where $f()$ is the density function of the assumed distribution, $S()$ is the corresponding survivor function, and $(t_{0j}, t_j, d_j, \mathbf{x}_j)$ is the information on the j th observation. The

parameters β_x and Θ are estimated from the data: β_x are the coefficients on \mathbf{x} , and Θ are ancillary parameters, if any, required by the assumed distribution. For instance, for the Weibull distribution $\Theta = (\beta_0, p)$, where β_0 is the scale parameter and p is the shape parameter, and we choose β_0 to denote the scale because, as we show later, for this model the scale can also be thought of as an intercept term for the linear predictor $\mathbf{x}\beta_x$.

The triple (t_{0j}, t_j, d_j) summarizes the survival experience for the observation: the subject was observed and known not to fail during the period $t_{0j} < t < t_j$, and then at $t = t_j$, the subject either failed ($d_j = 1$) or was censored ($d_j = 0$). Thus, the powers $(1 - d_j)$ and d_j in (13.1) serve to select either $S()$ or $f()$ as the numerator of the ratio. If censored, $S()$ is chosen, and that is the probability that the subject survives from 0 to t_j without failure. If $d_j = 1$, if the subject fails, $f()$ is chosen, and that is the “probability” of failure at time t_j . Either way, the numerator is divided by $S(t_{0j}|\mathbf{x}_j\beta_x, \Theta)$, which is the probability of surviving up to time t_{0j} , and thus whichever is the numerator is converted to a conditional probability or probability density for the time span under consideration. When $t_{0j} = 0$, $S(t_{0j}|\mathbf{x}_j\beta_x, \Theta) = 1$.

Note that the terms of the likelihood function are stated in terms of observations and not subjects, which is to say that there may be more than one observation on a subject. In simple survival data, there is a one-to-one correspondence between observations and subjects and, in addition, $t_{0j} = 0$. But in more complex cases, a subject may have multiple observations, as we have seen previously:

id	_t0	_t	x	_d
101	0	5	4	0
101	5	7	3	0
101	7	8	4	1

In this way, parametric models are generalized to allow time-varying covariates.

Equation (13.1) may be equivalently written as

$$L_j(\beta_x, \Theta) = \frac{S(t_j|\mathbf{x}_j\beta_x, \Theta)}{S(t_{0j}|\mathbf{x}_j\beta_x, \Theta)} \{h(t_j|\mathbf{x}_j\beta_x, \Theta)\}^{d_j}$$

which you can obtain by substitution using the formulas given in Chapter 2. This variation is as easily thought about. The first part, $S(t_j|\dots)/S(t_{0j}|\dots)$, is the probability of survival from t_{0j} until t_j . The last part, $h(t|\dots)^{d_j}$, becomes $h(t_j|\dots)$ if the span ends in failure (which is the corresponding risk of that event at time t_j), or 1 if the span ends in censoring.

All parametric likelihoods are of the above form, and the only difference among the models is how $S()$ (and therefore $f()$ and $h()$) is chosen.

streg can fit any of six parametric models: exponential, Weibull, Gompertz, log-normal, log-logistic, and generalized gamma.

13.1 The exponential model

13.1.1 Exponential regression in the PH metric

The exponential model is the simplest of parametric survival models because it assumes that the baseline hazard is constant,

$$\begin{aligned} h(t|\mathbf{x}_j) &= h_0(t) \exp(\mathbf{x}_j \boldsymbol{\beta}_x) \\ &= \exp(\beta_0) \exp(\mathbf{x}_j \boldsymbol{\beta}_x) \\ &= \exp(\beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x) \end{aligned}$$

for some constant β_0 . We use the notation β_0 to emphasize the fact that the constant may also be thought of as an intercept term from the linear predictor. Using the well-known relationships for the exponential model,

$$\begin{aligned} H(t|\mathbf{x}_j) &= \exp(\beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x) t \\ S(t|\mathbf{x}_j) &= \exp\{-\exp(\beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x) t\} \end{aligned} \quad (13.2)$$

If you fit a model in this metric and display the regression coefficients instead of hazard ratios, the intercept term `_b[_cons]` is the estimate of β_0 . To do this, specify the `nohr` option either during estimation or upon replay:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg age protect, dist(exp) nohr
      failure _d: fracture
  analysis time _t: timel
             id: id

Iteration 0:  log likelihood = -60.067085
Iteration 1:  log likelihood = -54.034598
Iteration 2:  log likelihood = -47.553588
Iteration 3:  log likelihood = -47.534671
Iteration 4:  log likelihood = -47.534656
Iteration 5:  log likelihood = -47.534656

Exponential regression -- log relative-hazard form
No. of subjects =          48          Number of obs =          106
No. of failures =          31
Time at risk   =          714

Log likelihood = -47.534656          LR chi2(2)      =          25.06
                                Prob > chi2       =          0.0000
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
age	.0809663	.0342787	2.36	0.018	.0137813 .1481514
protect	-1.688958	.3703357	-4.56	0.000	-2.414803 -.9631137
_cons	-7.892737	2.458841	-3.21	0.001	-12.71198 -3.073498

Translating the above to our mathematical notation,

$$\mathbf{x}_j \hat{\boldsymbol{\beta}}_x = -1.69 \text{protect}_j + 0.08 \text{age}_j$$

and $\hat{\beta}_0 = -7.89$. Thus, our estimate of the baseline hazard is $\hat{h}_0(t) = \exp(-7.89) = 0.00037$, and our estimate of the overall hazard is

$$h(t|\mathbf{x}_j) = 0.00037 \exp(-1.69\text{protect}_j + 0.08\text{age}_j)$$

In the exponential model, $h_0(t)$ being constant means that the failure rate is independent of time, and thus the failure process is said to lack memory. As such, you may be tempted to view exponential regression as suitable for use only in the simplest of cases.

That would be unfair. There is another sense in which the exponential model is the basis of all other models. The baseline hazard $h_0(t)$ is constant, and given the proportional hazards model,

$$h(t|\mathbf{x}_j) = h_0(t) \exp(\mathbf{x}_j\boldsymbol{\beta}_x)$$

the way in which the overall hazard varies with time is purely a function of how $\mathbf{x}_j\boldsymbol{\beta}_x$ varies with time, if at all. The overall hazard need not be constant with time; it is just that every little bit of how the hazard varies must be specified in $\mathbf{x}_j\boldsymbol{\beta}_x$. If you fully understand a process, you should be able to do that. When you do not fully understand a process, you are forced to assign a role to time (the hazard increases with time, the hazard decreases with time, etc.), and in that way, you hope, put to the side your ignorance and still describe the part of the process you do understand.

It is a rare process that is fully understood, but your goal should be to understand the process well enough that, some day, you could fit an exponential model. And it is worth appreciating that, once you do fully understand a process, time plays no role. Rather, you have a model in which it is the accumulation of this toxin or of a particular kind of information or of something else that accounts for the apparent role of time, and you can describe exactly how the toxin, knowledge, or whatever accumulates with time.

In addition, exponential models can be used to model the overall hazard as a function of time, if they include t or functions of t as covariates.

For example, for our hip-fracture data we could fit a model in which we simply claim that the hazard is constant with time,

$$h(t|\mathbf{x}_j) = h_0 \exp(\beta_1\text{protect}_j + \beta_2\text{age}_j)$$

for some constant h_0 , or we could directly include t in the model,

$$h(t|\mathbf{x}_j) = h_0 \exp(\beta_1\text{protect}_j + \beta_2\text{age}_j + \beta_3t) \quad (13.3)$$

As a computer issue, we will have to construct the time-varying covariate for t , but that is not difficult (although it may require considerable memory depending on your dataset. Note: `streg` has no `tvc()` option like `stcox` and so you must `stsplot` the data):

```
. summarize _t
      Variable |      Obs      Mean   Std. Dev.      Min      Max
-----+-----+-----+-----+-----+-----+-----
      _t      |      106     11.5283    8.481024         1       39

. stsplitt myt, at(1(1)39)
(608 observations (episodes) created)

. streg protect age myt, dist(exp) nohr
      failure _d: fracture
      analysis time _t: time1
      id: id

Iteration 0:  log likelihood = -60.067085
Iteration 1:  log likelihood = -51.705495
Iteration 2:  log likelihood = -43.731761
Iteration 3:  log likelihood = -43.704655
Iteration 4:  log likelihood = -43.704635
Iteration 5:  log likelihood = -43.704635

Exponential regression -- log relative-hazard form
No. of subjects =          48          Number of obs =          714
No. of failures =          31
Time at risk    =          714
Log likelihood   = -43.704635          LR chi2(3)      =          32.72
                                          Prob > chi2     =          0.0000
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
protect	-2.225445	.4381355	-5.08	0.000	-3.084175 -1.366715
age	.1023765	.036895	2.77	0.006	.0300636 .1746895
myt	.066876	.0234889	2.85	0.004	.0208387 .1129133
_cons	-9.862588	2.694014	-3.66	0.000	-15.14276 -4.582417

Note how we used `stsplitt` to split the records at each time in $\{1, 2, 3, \dots, 39\}$ and create the time varying covariate `myt` = t .

Given the above estimation results, we can rewrite the model in (13.3) as

$$h(t|\mathbf{x}_j) = \{h_0 \exp(\beta_3 t)\} \exp(\beta_1 \text{protect}_j + \beta_2 \text{age}_j)$$

and, in a logical sense, $\{h_0 \exp(\beta_3 t)\}$ is our baseline hazard. Since $h_0 = \exp(\beta_0)$ in an exponential model, in terms of our estimates we can calculate and graph the baseline hazard

```
. gen hazard = exp(_b[_cons]) * exp(_b[myt] * myt)
. line hazard myt, c(J) sort ltitle("baseline hazard")
```

which produces Figure 13.1.

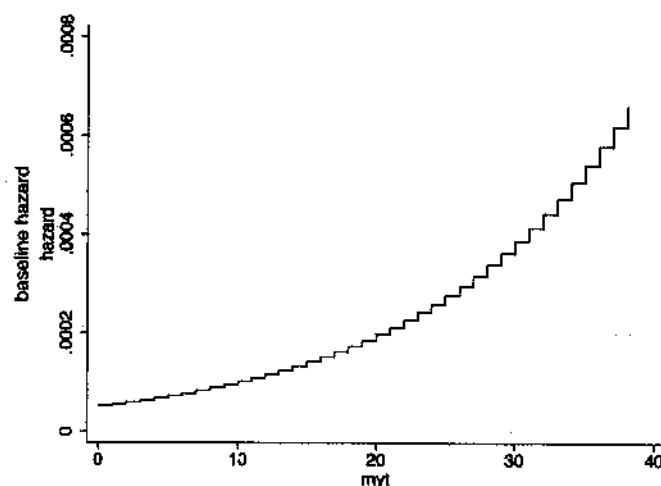


Figure 13.1: Estimated baseline hazard function

We did not really fit a smooth function of t for the baseline hazard in `streg` because that would require `stsplitting` our records infinitely finely. However, the “grid” size we chose (every integer) seems to be adequate for our purposes.

In this case, including t in the model resulted in a hazard function that not only increases with t , but increases at an increasing rate. That, however, is due to the functional form we chose and may not really reflect what is going on in the data. In our parameterization, $h_0 \exp(\beta_3 t)$, the only way the hazard function can increase is at an increasing rate.

Continuing this development, let’s try $\ln(\text{myt} + 1)$ instead of `myt` as a covariate so that the model becomes

$$\begin{aligned} h(t|\mathbf{x}_j) &= h_0 \exp\{\beta_1 \text{protect}_j + \beta_2 \text{age}_j + \beta_3 \ln(\text{myt} + 1)\} \\ &= [h_0 \exp\{\beta_3 \ln(\text{myt} + 1)\}] \exp(\beta_1 \text{protect}_j + \beta_2 \text{age}_j) \end{aligned}$$

where we now treat $h_0 \exp\{\beta_3 \ln(\text{myt} + 1)\}$ as the *de facto* baseline hazard,

```
. gen lmyt = ln(myt+1)
. streg protect age lmyt, dist(exp)
(output omitted)
. gen hazard2 = exp(_b[_cons] + _b[lmyt]*lmyt)
. line hazard2 myt, c(J) sort l1title("baseline hazard")
```

which produces Figure 13.2.

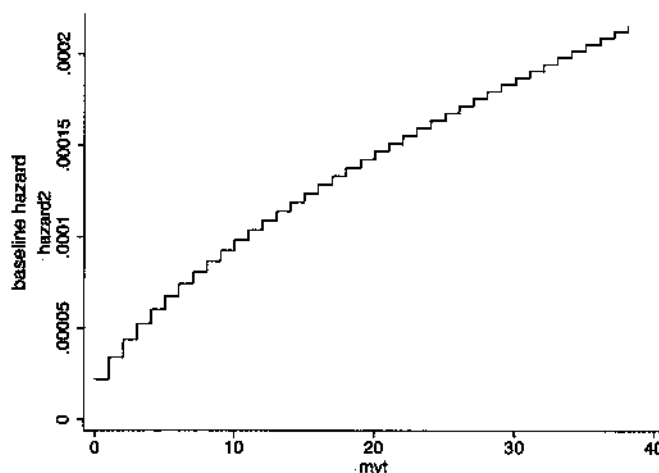


Figure 13.2: Estimated baseline hazard function using $\ln(\text{myt} + 1)$

In this case, we get a hazard that increases at a (gently) decreasing rate, but once again, this could be an artifact of our parameterization.

Rather than specifying a particular functional form of `myt` to include in our model, we can get a better idea of the hazard function preferred by the data by including a crude step function of `myt` instead, in which the height of each step is its own model parameter,

```
. gen in_t1 = 0<=myt & myt<10
. gen in_t2 = 10<=myt & myt<20
. gen in_t3 = 20<=myt & myt<30
. gen in_t4 = 30<=myt
. streg protect age in_t2 in_t3 in_t4, dist(exp)
  (output omitted)
. gen hazard3 = exp(_b[_cons] + _b[in_t2]*in_t2 + /*
>               */ _b[in_t3]*in_t3 + _b[in_t4]*in_t4)
. line hazard3 myt, c(J) sort ltitle("baseline hazard")
```

which produces Figure 13.3.

(Continued on next page)

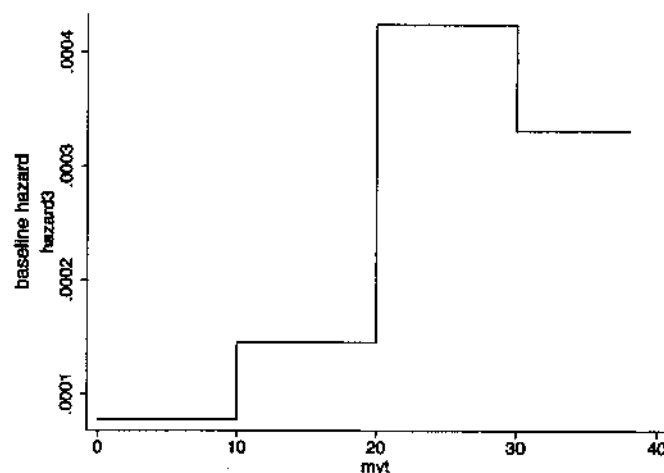


Figure 13.3: Estimated baseline hazard function using a step function

Note that for reasons of collinearity, we do not include the first step indicator variable `in_t1` in our model—the intercept term takes care of that step for us.

From Figure 13.3, it appears that the hazard is increasing at an increasing rate but then jumps down at the end. This graph, however, can be misleading because there are very different numbers of observations in each of the steps. A better graph would define the steps so as to put 25% of the data in each step,

```
. summarize myt, detail
```

myt					
	Percentiles	Smallest			
1%	0	0			
5%	0	0			
10%	1	0	Obs		714
25%	3	0	Sum of Wgt.		714
50%	8		Mean		10.61765
		Largest	Std. Dev.		8.678827
75%	16	35			
90%	23	36	Variance		75.32204
95%	28	37	Skewness		.7923223
99%	33	38	Kurtosis		2.79787

```
. gen in_t1 = 0<=myt & myt<3
. gen in_t2 = 3<=myt & myt<8
. gen in_t3 = 8<=myt & myt<16
. gen in_t4 = 16<=myt
. streg protect age in_t2 in_t3 in_t4, dist(exp)
(output omitted)
. gen hazard4 = exp(_b[_cons] + _b[in_t2]*in_t2 + /*
>                */ _b[in_t3]*in_t3 + _b[in_t4]*in_t4)
```

```
. line hazard4 myt, c(J) sort l1title("baseline hazard")
```

which produces Figure 13.4.

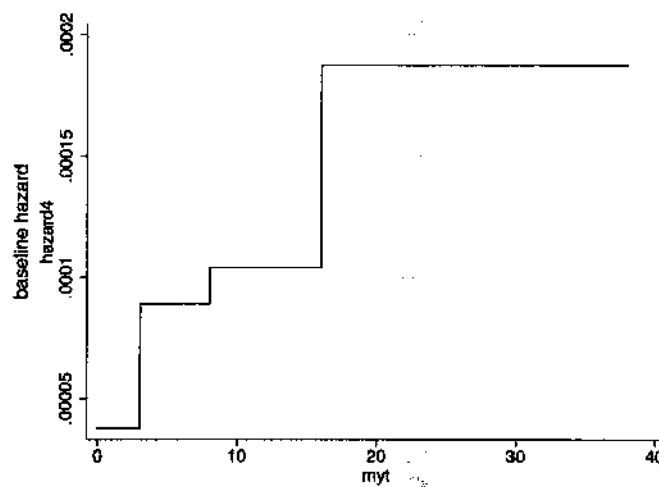


Figure 13.4: Estimated baseline hazard function using a better step function

Looking at this graph, we now know what we believe: the hazard increases at a decreasing rate. Actually, a better statement would be the following: up until time $t = 16$, we believe it certainly is not increasing faster than linearly (probably increasing at a decreasing rate), and after that, we really do not know. In any case, we compare this hazard to the result we obtained when we used $\ln(\text{myt} + 1)$,

```
. line hazard2 hazard4 myt, c(J J) sort l1title("baseline hazard")
```

which produces Figure 13.5.

(Continued on next page)

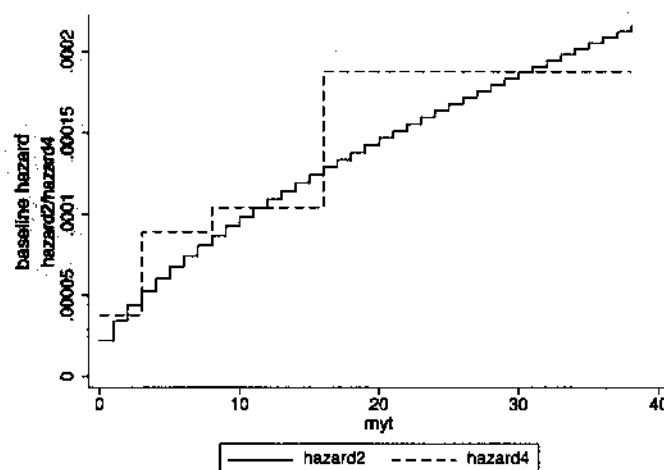


Figure 13.5: Comparison of estimated baseline hazards

It is a fair fit, except perhaps in the second time interval.

The point of the above exercise is not that you should use exponential regression when you have no idea of what the baseline hazard looks like—use Cox regression instead. The point is that exponential regression can be used to fit models in which the hazard varies with time, and that may be a reasonable thing to do, especially if you want to verify the fit of another parametric model. For instance, pretend that you have strong reason to believe that the formulation ought to be Weibull. Even after fitting a Weibull model, you could use the exponential model with dummy variables for time intervals to verify that the Weibull fit was reasonable.

The method we have used—splitting the data on each integer time point—uses considerable memory. Depending on the size of your dataset, that may not matter, but note that the piecewise-constant hazard with steps at the 25th, 50th, and 75th percentiles of time would have required only splitting the data on three time points.

13.1.2 Exponential regression in the AFT metric

In the accelerated failure-time formulation of Section 12.2.2, we have $\tau_j = \exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j$, and in the exponential model, it is assumed that

$$\tau_j \sim \text{Exponential}\{\exp(\beta_0)\}$$

That is, τ_j is distributed as exponential with mean $\exp(\beta_0)$. This implies then that

$$\begin{aligned} \ln(t_j) &= \mathbf{x}_j\boldsymbol{\beta}_x + \ln(\tau_j) \\ &= \beta_0 + \mathbf{x}_j\boldsymbol{\beta}_x + u_j \end{aligned}$$

where u_j follows the extreme-value (Gumbel) distribution, the result of which is that

$$E\{\ln(t_j)|\mathbf{x}_j\} = \beta_0 + \mathbf{x}_j\boldsymbol{\beta}_x + \Gamma'(1)$$

where $\Gamma'(1)$ is Euler's constant.

□ Technical Note

The Gumbel distribution gets the name extreme-value distribution because it can be shown to be the limiting distribution (as $n \rightarrow \infty$) of the maximum value from a sample of n identically distributed random deviates with a continuous probability distribution supported on the real line.

□

Alternatively, we can derive the AFT formulation by accelerating the effect of time on survival experience. At baseline values of \mathbf{x} , $\tau_j = t_j$ since all covariates are equal to zero. Thus, the baseline survivor function of t_j is that from an exponential distribution with mean $\exp(\beta_0)$; i.e.,

$$S_0(t_j) = \exp\{-\exp(-\beta_0)t_j\}$$

In an AFT model, the effect of the covariates is to accelerate time by a factor of $\exp(-\mathbf{x}_j\boldsymbol{\beta}_x)$. Thus, for the AFT model,

$$\begin{aligned} S(t_j|\mathbf{x}_j) &= S_0\{\exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j\} \\ &= \exp\{-\exp(-\beta_0)\exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j\} \\ &= \exp\{-\exp(-\beta_0 - \mathbf{x}_j\boldsymbol{\beta}_x)t_j\} \end{aligned}$$

and when we compare with (13.2), we find that the transformation from the PH to the AFT metric (for the exponential model) is simply one of flipping the signs of the regression coefficients.

Using our hip-fracture data, we can obtain exponential regression estimates in the AFT metric,

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(exp) time
      failure _d: fracture
      analysis time _t: time1
              id: id

Iteration 0:  log likelihood = -60.067085
Iteration 1:  log likelihood = -54.034598
Iteration 2:  log likelihood = -47.553588
Iteration 3:  log likelihood = -47.534671
Iteration 4:  log likelihood = -47.534656
Iteration 5:  log likelihood = -47.534656
```

Exponential regression -- accelerated failure-time form					
No. of subjects =	48	Number of obs =	106		
No. of failures =	31				
Time at risk =	714				
Log likelihood =	-47.534656	LR chi2(2) =	25.06		
		Prob > chi2 =	0.0000		
_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
protect	1.688958	.3703357	4.56	0.000	.9631137 2.414803
age	-.0809663	.0342787	-2.36	0.018	-.1481514 -.0137813
_cons	7.892737	2.458841	3.21	0.001	3.073498 12.71198

from which it follows that

$$\ln(t_j) = 7.89 + 1.69\text{protect}_j - 0.08\text{age}_j + u_j$$

or, if you prefer,

$$\tau_j = \exp(-1.69\text{protect}_j + 0.08\text{age}_j)t_j$$

The effect of **protect** is to slow down time, and the effect of **age** is to accelerate it, or equivalently, the effect of **protect** is to delay failure, and that of **age** is to hasten it.

As we have previously discussed in Section 13.1.1, we do not believe the assumption of constant hazard is appropriate for these data, and our previous concerns apply as much to the exponential model estimated in the AFT metric as they do to the model fit in the PH metric.

The approach we suggested for relaxing the assumption of constant hazard in the previous section, while applicable in this metric, is nearly impossible to interpret, and so we do not recommend it. You could, however, **stsplit** the data and introduce *t* as a variable in the model. If you do that, you will then discover that time speeds up with time, which you will have to think about carefully. When you go to evaluate the fit—to find out whether time should accelerate with time or, say, $\ln(\text{time})$ —you find that nearly impossible to do because **predict** will not help you, or at least it will not help you if you stick with the time metric and predict failure times. In time-varying data, **predict, time** calculates predictions for each observation as if the covariates are constant, and that is not what you need. All of this is to say that if you want to use the exponential model to explore the hazard, you will find that much easier to do in the proportional hazards metric.

13.2 Weibull regression

13.2.1 Weibull regression in the PH metric

The Weibull model assumes a baseline hazard of the form $h_0(t) = pt^{p-1} \exp(\beta_0)$, where *p* is some ancillary shape parameter estimated from the data and the scale parameter is

parameterized as $\exp(\beta_0)$. Given a set of covariates \mathbf{x}_j , under the proportional hazards model,

$$\begin{aligned} h(t|\mathbf{x}_j) &= h_0(t) \exp(\mathbf{x}_j \boldsymbol{\beta}_x) \\ &= pt^{p-1} \exp(\beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x) \end{aligned}$$

and this yields

$$\begin{aligned} H(t|\mathbf{x}_j) &= \exp(\beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x) t^p \\ S(t|\mathbf{x}_j) &= \exp\{-\exp(\beta_0 + \mathbf{x}_j \boldsymbol{\beta}_x) t^p\} \end{aligned} \quad (13.4)$$

The estimated scale parameter, thus, is obtained by exponentiating the estimated intercept coefficient.

The Weibull distribution can provide a variety of monotonically increasing or decreasing shapes of the hazard function, and their shape is determined by the estimated parameter p . Figure 13.6 gives a few examples.

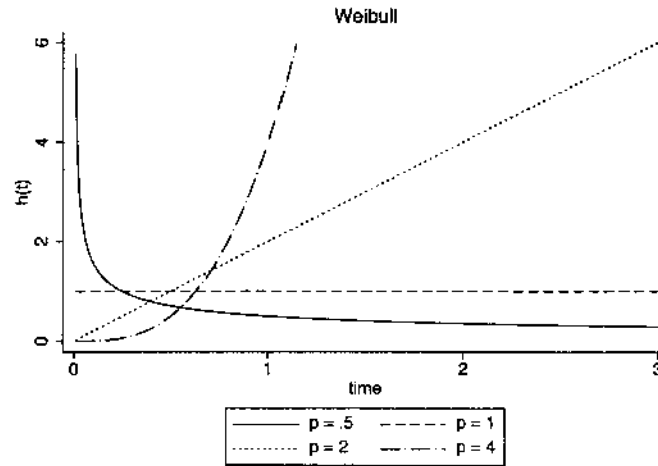


Figure 13.6: Weibull hazard function for various p

Note that when $p = 1$, the hazard is constant, and thus, the Weibull model reduces to the exponential model. For other values of p , the Weibull hazard is not constant; it is monotone decreasing when $p < 1$ and monotone increasing when $p > 1$. The Weibull is suitable for modeling data that exhibits monotone hazard rates.

Using `streg, dist(weibull)` with our hip-fracture data,

```

. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(weibull)
      failure _d: fracture
      analysis time _t: time1
      id: id

Fitting constant-only model:
Iteration 0:  log likelihood = -60.067085
Iteration 1:  log likelihood = -59.30148
Iteration 2:  log likelihood = -59.298481
Iteration 3:  log likelihood = -59.298481

Fitting full model:
Iteration 0:  log likelihood = -59.298481
Iteration 1:  log likelihood = -54.887563
Iteration 2:  log likelihood = -42.123875
Iteration 3:  log likelihood = -41.993012
Iteration 4:  log likelihood = -41.992704
Iteration 5:  log likelihood = -41.992704

Weibull regression -- log relative-hazard form
No. of subjects =          48          Number of obs =          106
No. of failures =          31
Time at risk   =          714

Log likelihood =  -41.992704          LR chi2(2)      =          34.61
                                      Prob > chi2     =          0.0000

```

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
protect	.1099611	.0448214	-5.42	0.000	.0494629	.2444548
age	1.117186	.0423116	2.93	0.003	1.03726	1.203271
/ln_p	.5188694	.1376486	3.77	0.000	.2490831	.7886556
p	1.680127	.2312671			1.282849	2.200436
1/p	.5951931	.0819275			.4544553	.7795152

Note that in the output above Stata reports a Wald test for $H_0: \ln(p) = 0$ for which the test statistic is 3.77 and that we can reject the null hypothesis. This is equivalent to testing $H_0: p = 1$, and thus we can reject that the hazard is a constant.

□ Technical Note

From the above output, we see that the results for three parameterizations of p are given: $\ln(p)$, p itself, and $1/p$. The first parameterization, $\ln(p)$, represents the metric in which the model is actually fit, and by estimating in this metric we are assured of obtaining an estimate of p that is positive, and the estimate of p is obtained by transforming $\ln(p)$ post-estimation. The third parameterization, $1/p$, is given so that one may compare these results with those of other researchers who commonly choose to parameterize the shape in this manner.

□

By default, **streg** reports hazard ratios (exponentiated coefficients) when estimating in the PH metric. We see that wearing the hip-protection device reduces the hazard of hip fracture to almost one-tenth of what it would be otherwise.

While results reported in this way make interpreting the effects of variables on the relative hazard easy, these results do now show all the parameters of the baseline hazard function because one of those parameters is β_0 . We can, however, redisplay the results and ask for the coefficients:

```
. streg, nohr
Weibull regression -- log relative-hazard form
No. of subjects =          48          Number of obs   =          106
No. of failures =          31
Time at risk   =          714
Log likelihood = -41.992704          LR chi2(2)       =          34.61
                                      Prob > chi2      =          0.0000
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
protect	-2.207628	.4076113	-5.42	0.000	-3.006532 -1.408725
age	.1108134	.0378734	2.93	0.003	.036583 .1850439
_cons	-11.67104	2.90919	-4.01	0.000	-17.37295 -5.969135
/ln_p	.5188694	.1376486	3.77	0.000	.2490831 .7886556
p	1.680127	.2312671			1.282849 2.200436
1/p	.5951931	.0819275			.4544553 .7795152

From these results, we see that

$$\begin{aligned}
 h_0(t) &= pt^{p-1} \exp(\beta_0) \approx 1.68t^{0.68} \exp(-11.67) \\
 &= 0.0000144t^{0.68}
 \end{aligned}$$

The baseline hazard is so small because it is being evaluated at `protect==0` and `age==0` and, as done previously, we could move the baseline to reflect a more reasonable group by estimating on, say, `age60 = age-60`. In any case, as demonstrated in Section 13.1.1, we could graph the baseline hazard by typing

```
. gen h = 0.0000144 * _t^0.68
. line h _t, c(1) sort
```

This time, however, we will follow a different approach to estimating the baseline hazard, and type

```
. replace protect=0
. replace age=0
. predict h, hazard
```

With this method, we let **predict** do all the work. **predict, hazard** will calculate the hazard at the recorded values of the covariates after any parametric estimation using **streg**. If we set all the covariates to zero, we obtain the baseline hazard. The full capabilities of **predict** when used after **streg** are covered in Chapter 14.

If we perform the above and estimate the baseline hazard for our fitted model, we can graph it using

```
. line h _t, c(1) sort ltitle("baseline hazard")
```

which produces Figure 13.7.

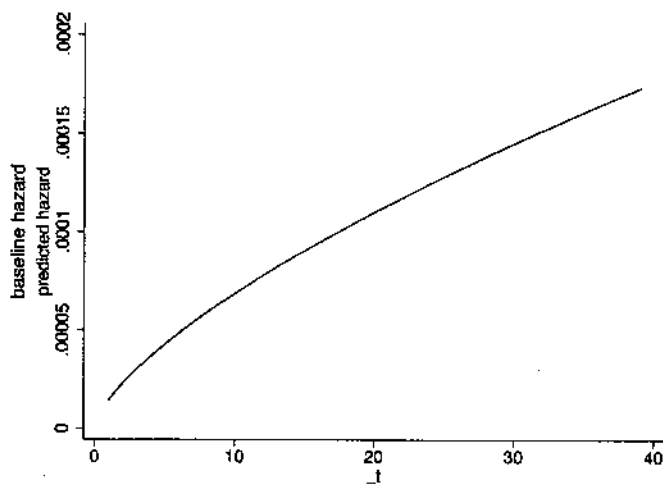


Figure 13.7: Estimated baseline hazard function for Weibull model

In Section 13.1.1, we fit a step function for the hazard. We can compare that to the baseline hazard produced by the Weibull model. In doing these kinds of comparisons between classes of parametric models, it is important to make them at reasonable values of the covariates, and thus we will make the comparison at `protect==0` and `age==70`. In replicating the calculations for the step function hazard, we let `predict` do the work:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
. stsplot myt, at(1(1)39)
. gen in_t1 = 0<=myt & myt<3
. gen in_t2 = 3<=myt & myt<8
. gen in_t3 = 8<=myt & myt<16
. gen in_t4 = 16<=myt
. streg protect age in_t2 in_t3 in_t4, dist(exp)
. gen hage = age
. gen hprotect = protect
. replace age = 70
. replace protect = 0
. predict hstep, hazard
. replace age = hage
. replace protect = hprotect
. streg protect age, dist(weib)
. replace age=70
. replace protect=0
. predict h, hazard
. line hstep h _t, c(1 1) sort ltitle("baseline hazard")
```

The result is shown in Figure 13.8.

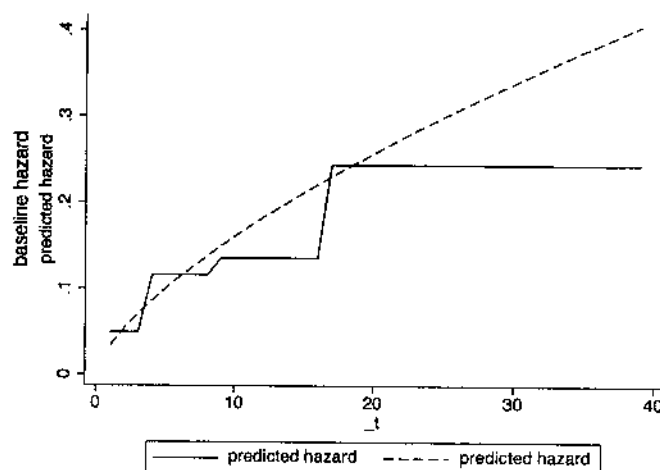


Figure 13.8: Comparison of exponential (step) and Weibull hazards

As we said in Section 13.1.1, when commenting on Figure 13.4, “Up until $t = 16$, we believe [the hazard] is certainly not increasing faster than linearly [...] and, after that, we really do not know.” We do not know because of the small amount of data after $t = 16$, and so, in Figure 13.8, we are not much bothered by the difference on the right between the two hazard functions.

Fitting null models

You can use `streg` to obtain estimates of the parameters of the parametric distribution when there are no covariates in the model. For example, if we wanted to find the Weibull distribution that fit our data by treating all subjects as alike (that is, ignoring age and protect), we could type

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg, dist(weibull) time
      failure _d: fracture
      analysis time _t: time1
                  id: id

Fitting constant-only model:
Iteration 0:   log likelihood = -60.067085
Iteration 1:   log likelihood = -59.30148
Iteration 2:   log likelihood = -59.298481
Iteration 3:   log likelihood = -59.298481
Fitting full model:
Iteration 0:   log likelihood = -59.298481
```

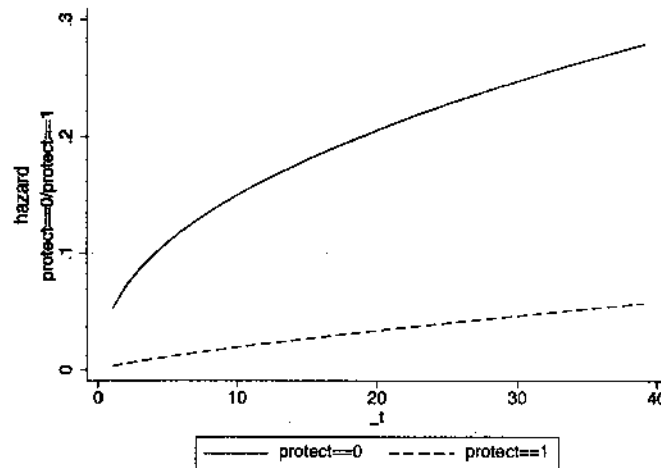



Figure 13.9: Estimated Weibull hazard functions over values of `protect`

Let us take a moment to reflect. In Figure 8.2 of Section 8.2.7, when we discussed nonparametric analysis, we parameterized neither the effect nor the underlying hazard function. Later, in Section 8.5, we illustrated nonparametric tests, which similarly made no assumptions about the functional forms of the effect or the hazards.

In Section 9.1, we used Cox regression. In that analysis, we parameterized the effect but not the underlying hazard.

Above, we just did a visual comparison, leaving the effect unparameterized but parameterizing the hazard function. And, of course, we could fit a Weibull model on `protect` and thus parameterize both.

All our tests and inspections have yielded the same result: there is a difference associated with wearing the hip-protection device. That results are the same need not be the case, and the question then arises, which should you use?

There is no simple answer to that question, so let us instead understand how to determine the answer in particular cases.

The advantage of the modeling-the-effect approaches is that you can control for the effects of other variables. For instance, in our hip data, we know that patients vary in age, and we know age also affects outcome. In a carefully controlled experiment, we could ignore that because the average ages (and the distribution of age as well) of the control and experimental groups would be the same, but in our data that is not so.

The disadvantage of the modeling-the-effect approaches is that you could model the effect incorrectly, and there are two ways you could do that. You could model incorrectly the effect of other variables, or you could mismodel the effect itself, for example, by stating its functional form incorrectly.

Effects of the form “apply the treatment and get an overall improvement” are often not simple. Effects can vary with other covariates (being perhaps larger for males than for females), and effects can vary with time, which is to say, aspects that change over time and that are not measured. For instance, a treatment might involve surgery, and in such cases, there may be a greater risk to be followed by a lesser risk in the future.

It is because of these concerns that looking at graphs such as the one above is useful, whether you are engaging in parametric or semiparametric modeling (although, when doing semiparametric modeling, you can only indirectly look at the hazard function by looking at the cumulative hazard or survivor function).

In most real circumstances, you will be forced into parametric or semiparametric analysis. Nonparametric analysis is useful when the experiment has been carefully controlled, although even controlled experiments are sometimes not controlled adequately. In all cases, nonparametric analysis is a useful starting point. In nonexperimental situations in the presence of covariates, you do this more as a data description technique rather than in hopes of producing any final analysis that you can believe. You, as a researcher, should be able to describe the survival experience, say, as reflected in a graph of the survivor function or cumulative hazard function for your data, ignoring the complications of confounding variables and the like. Before disentangling reality, you need to be able to describe what the reality is that you are starting with.

So, our position is that you will likely be forced into parameterizing the effect. This is perhaps due more to our past analysis experiences. In a well-designed controlled experiment, there is nothing wrong with stopping at nonparametric analysis.

If you do need to continue, should you parameterize the hazard function? On this issue, different researchers feel differently. We are very favorably disposed to parametric analysis when you have good reason to believe that the hazard function ought to follow a certain shape. Imposing a hazard function is an excellent way of improving the efficiency of your estimates and helping to avoid being misled by the fortuity of chance. On the other hand, when you do not have a good a priori reason to know the shape of the hazard, you should use semiparametric analysis.

When choosing between a semiparametric and parametric analysis, you must also take into consideration what information you are trying to obtain. If all you care about are hazard ratios (parameter effects) in a PH model, then you are probably better off with a semiparametric analysis. If you are interested in predicting the time to failure, however, some sort of parametric assumption as to the hazard is necessary. In this case, even if you do not have a priori knowledge as to the shape of the hazard, you can use `streg`, in all its implementations, to compare various functional forms of the hazard, and you can use the piecewise-exponential model to “nonparametrically” check the validity of any parametric form you wish to posit.

13.2.2 Weibull regression in the AFT metric

In the accelerated failure-time formulation of the Weibull, we have $\tau_j = \exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j$, and in the Weibull regression model it is assumed that

$$\tau_j \sim \text{Weibull}(\beta_0, p)$$

That is, τ_j is distributed as Weibull with parameters (β_0, p) with the cumulative distribution function

$$F(\tau) = 1 - \exp[-\{\exp(-\beta_0)\tau\}^p] \quad (13.5)$$

This implies then that

$$\begin{aligned} \ln(t_j) &= \mathbf{x}_j\boldsymbol{\beta}_x + \ln(\tau_j) \\ &= \beta_0 + \mathbf{x}_j\boldsymbol{\beta}_x + u_j \end{aligned}$$

where u_j follows the extreme-value (Gumbel) distribution with shape parameter p , the result of which is that

$$E\{\ln(t_j)|\mathbf{x}_j\} = \beta_0 + \mathbf{x}_j\boldsymbol{\beta}_x + \frac{\Gamma'(1)}{p}$$

where $\Gamma'(1)$ is Euler's constant. Alternatively, we can derive the AFT formulation by accelerating the effect of time on survival experience. At baseline values of \mathbf{x} , $\tau_j = t_j$ since all covariates are equal to zero. Thus, the baseline survivor function of t_j is obtained from (13.5) to be

$$S_0(t_j) = \exp[-\{\exp(-\beta_0)t_j\}^p]$$

In an AFT model, the effect of the covariates is to accelerate time by a factor of $\exp(-\boldsymbol{\beta}_x\mathbf{x}_j)$. Thus, for the AFT model,

$$\begin{aligned} S(t_j|\mathbf{x}_j) &= S_0\{\exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j\} \\ &= \exp[-\{\exp(-\beta_0)\exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j\}^p] \\ &= \exp[-\{\exp(-\beta_0 - \mathbf{x}_j\boldsymbol{\beta}_x)t_j\}^p] \end{aligned} \quad (13.6)$$

Comparison with (13.4) shows that one may transform the regression coefficients from one metric to the other using the following relationship:

$$\boldsymbol{\beta}_{AFT} = \frac{-\boldsymbol{\beta}_{PH}}{p} \quad (13.7)$$

In addition, some authors cast the Weibull shape parameter p (which is common to both parameterizations) in terms of $s = 1/p$, where s is known as the dispersion parameter. For convenience, Stata reports both the estimates and standard errors of p and $1/p$ in the output from Weibull regression. You fit Weibull models in the AFT metric by specifying the `time` option to `streg`:

```

. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(weib) time
      failure _d: fracture
      analysis time _t: time1
      id: id

Fitting constant-only model:
Iteration 0:  log likelihood = -60.067086
Iteration 1:  log likelihood = -59.30148
Iteration 2:  log likelihood = -59.298481
Iteration 3:  log likelihood = -59.298481

Fitting full model:
Iteration 0:  log likelihood = -59.298481
Iteration 1:  log likelihood = -54.887563
Iteration 2:  log likelihood = -42.123875
Iteration 3:  log likelihood = -41.993012
Iteration 4:  log likelihood = -41.992704
Iteration 5:  log likelihood = -41.992704

Weibull regression -- accelerated failure-time form
No. of subjects =          48          Number of obs   =          106
No. of failures =          31
Time at risk   =          714
Log likelihood = -41.992704          LR chi2(2)       =          34.61
                                      Prob > chi2      =          0.0000

      +-----+-----+-----+-----+-----+-----+
      | _t      | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
      +-----+-----+-----+-----+-----+-----+
      | protect | 1.313965 | .2366229 | 5.55 | 0.000 | .8501928 | 1.777737 |
      | age     | -.0659554 | .0221171 | -2.98 | 0.003 | -.1093041 | -.0226067 |
      | _cons    | 6.946524 | 1.575708 | 4.41 | 0.000 | 3.858192 | 10.03486 |
      +-----+-----+-----+-----+-----+-----+
      | /ln_p    | .5188694 | .1376486 | 3.77 | 0.000 | .2490831 | .7886556 |
      +-----+-----+-----+-----+-----+-----+
      | p        | 1.680127 | .2312671 |      |      | 1.282849 | 2.200436 |
      | 1/p      | .5951931 | .0819275 |      |      | .4544553 | .7795152 |
      +-----+-----+-----+-----+-----+-----+

```

We note that the estimate of p is identical to that from when we fit this Weibull model in the PH metric and that the regression coefficients obey the transformation given in (13.7).

13.3 Gompertz regression (PH metric)

The Gompertz model is available only in PH metric and assumes a baseline hazard

$$h_0(t) = \exp(\gamma t) \exp(\beta_0)$$

so that in the proportional hazards model

$$\begin{aligned}
 h(t|\mathbf{x}_j) &= h_0(t) \exp(\mathbf{x}_j \beta_x) \\
 &= \exp(\gamma t) \exp(\beta_0 + \mathbf{x}_j \beta_x)
 \end{aligned}$$

and thus,

$$\begin{aligned} H(t|\mathbf{x}_j) &= \gamma^{-1} \exp(\beta_0 + \mathbf{x}_j \beta_x) \{\exp(\gamma t) - 1\} \\ S(t|\mathbf{x}_j) &= \exp[-\gamma^{-1} \exp(\beta_0 + \mathbf{x}_j \beta_x) \{\exp(\gamma t) - 1\}] \end{aligned}$$

The Gompertz distribution is an old distribution that has been extensively used by medical researchers and biologists modeling mortality data. This distribution is suitable for modeling data with monotone hazard rates that either increase or decrease exponentially with time, and the ancillary parameter γ controls the shape of the baseline hazard.

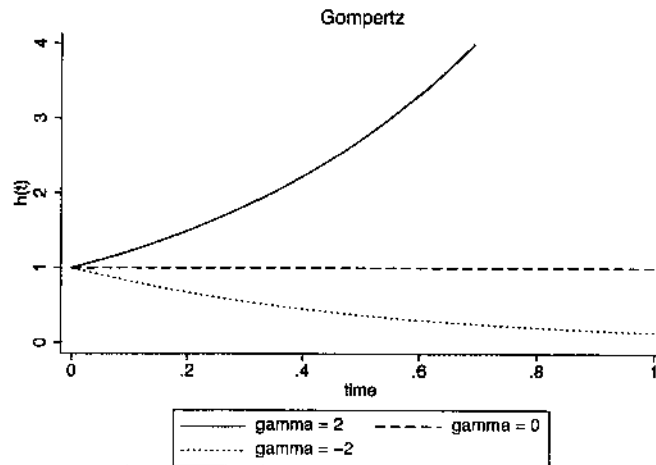


Figure 13.10: Gompertz hazard functions

As depicted in Figure 13.10, if γ is positive, the hazard function increases with time; if γ is negative, the hazard decreases with time; if $\gamma = 0$, the hazard function is $\exp(\beta_0)$ for all t , which is to say that it reduces to the hazard from the exponential model.

Some recent survival analysis texts such as Klein and Moeschberger (2003) restrict γ to be strictly positive. Note that when $\gamma < 0$, $S(t|\mathbf{x})$ decreases to a nonzero constant as $t \rightarrow \infty$, implying that there is a nonzero probability of never failing (living forever). That is, the hazard remains positive but decreases to zero at an exponential rate, which is too rapid a decline to guarantee eventual failure. By restricting γ to be positive, one is assured that $S(t|\mathbf{x})$ tends to zero as $t \rightarrow \infty$.

Although the above argument may be desirable from a mathematical perspective, Stata's implementation takes the more traditional approach of not restricting γ . In survival studies, subjects are not followed forever—there is a date when the study ends—and in many investigations (specifically in medical research), an exponentially decreasing hazard rate is clinically appealing.

Using our hip-fracture data, we fit

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(gompertz) nohr
      failure _d: fracture
      analysis time _t: time1
      id: id

Fitting constant-only model:
Iteration 0: log likelihood = -60.067085
Iteration 1: log likelihood = -59.734027
Iteration 2: log likelihood = -59.730841
Iteration 3: log likelihood = -59.73084

Fitting full model:
Iteration 0: log likelihood = -59.73084
Iteration 1: log likelihood = -53.743223
Iteration 2: log likelihood = -42.693409
Iteration 3: log likelihood = -42.609467
Iteration 4: log likelihood = -42.609352
Iteration 5: log likelihood = -42.609352

Gompertz regression -- log relative-hazard form
No. of subjects =      48          Number of obs   =      106
No. of failures =      31
Time at risk   =      714
Log likelihood = -42.609352          LR chi2(2)      =      34.24
                                      Prob > chi2      =      0.0000
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
protect	-2.311232	.4435779	-5.21	0.000	-3.180629	-1.441835
age	.1055445	.0371977	2.84	0.005	.0326383	.1784508
_cons	-10.19274	2.721731	-3.74	0.000	-15.52723	-4.858245
gamma	.0752767	.0233602	3.22	0.001	.0294915	.1210618

and find that $\hat{\gamma} > 0$, meaning that we estimate a hazard that is increasing exponentially. We can plot the baseline hazard using

```
. replace protect = 0
(72 real changes made)
. replace age = 0
(106 real changes made)
. predict h0, hazard
. line h0 _t, c(1) sort ltitle("baseline hazard")
```

which produces Figure 13.11.

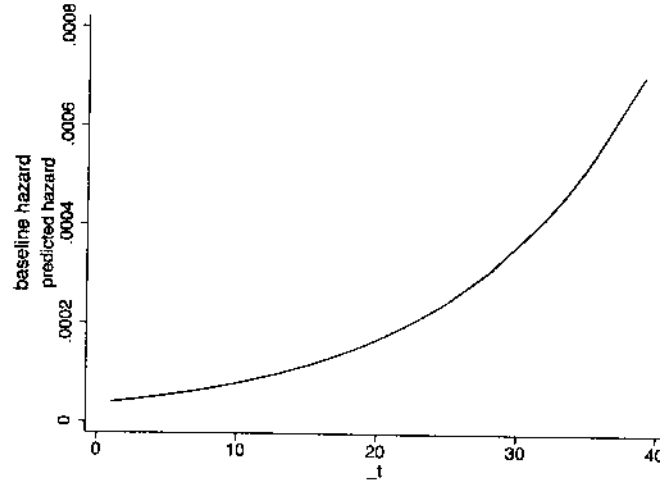


Figure 13.11: Estimated baseline hazard for the Gompertz model

Given our previous discussions on the exponential and Weibull models, this is a poor model for our data. We know that the hazard is increasing, and we have argued that it is probably increasing at a *decreasing* rate, something not allowed under a Gompertz specification where the hazard increases or decreases exponentially.

13.4 Log-normal regression (AFT metric)

In the accelerated failure-time formulation, we have $\tau_j = \exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j$, and for the log-normal regression model, it is assumed that

$$\tau_j \sim \text{Lognormal}(\beta_0, \sigma)$$

That is, τ_j is distributed as log-normal with parameters (β_0, σ) with cumulative distribution function

$$F(\tau) = \Phi\left(\frac{\ln \tau - \beta_0}{\sigma}\right) \quad (13.8)$$

where $\Phi()$ is the cumulative distribution function for the standard Gaussian (normal) distribution. Thus,

$$\begin{aligned} \ln(t_j) &= \mathbf{x}_j\boldsymbol{\beta}_x + \ln(\tau_j) \\ &= \beta_0 + \mathbf{x}_j\boldsymbol{\beta}_x + u_j \end{aligned}$$

where u_j follows a standard normal distribution with mean 0 and standard deviation σ . That is, for the log-normal model, transforming time into $\ln(\text{time})$ converts the problem into one of simple linear regression (with possible censoring). As a result,

$$E\{\ln(t_j)|\mathbf{x}_j\} = \beta_0 + \mathbf{x}_j\beta_x$$

Alternatively, we can derive the AFT formulation by accelerating the effect of time on survival experience at baseline, where all covariates are equal to zero. Thus, the baseline survivor function of t_j is obtained from (13.8) to be

$$S_0(t_j) = 1 - \Phi\left(\frac{\ln t_j - \beta_0}{\sigma}\right)$$

In an AFT model, the effect of the covariates is to accelerate time by a factor of $\exp(-\mathbf{x}_j\beta_x)$. Thus, for the AFT model,

$$\begin{aligned} S(t_j|\mathbf{x}_j) &= S_0\{\exp(-\mathbf{x}_j\beta_x)t_j\} \\ &= 1 - \Phi\left[\frac{\ln\{\exp(-\mathbf{x}_j\beta_x)t_j\} - \beta_0}{\sigma}\right] \\ &= 1 - \Phi\left\{\frac{\ln t_j - (\beta_0 + \mathbf{x}_j\beta_x)}{\sigma}\right\} \end{aligned} \quad (13.9)$$

The attractive feature (for some problems) of this distribution is that the hazard function is nonmonotonic—it increases and then decreases; see Figure 13.12.

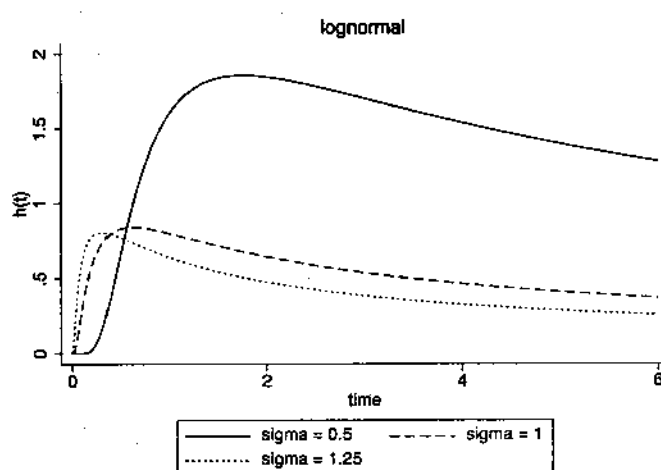


Figure 13.12: Examples of log-normal hazard functions ($\beta_0 = 0$)

This model has no natural PH interpretation. That is, from (13.9), we could certainly derive the hazard as a function of \mathbf{x}_j as

$$h(t|\mathbf{x}_j) = \frac{-\frac{d}{dt}S(t|\mathbf{x}_j)}{S(t|\mathbf{x}_j)}$$

but there is no choice of $h_0(t)$ for which this reduces to the form

$$h(t|\mathbf{x}_j) = h_0(t) \exp(\mathbf{x}_j \boldsymbol{\beta}_x)$$

and so this model is parameterized in the AFT metric only.

Although we do not think the log-normal model is an appropriate model for our hip-fracture data, by way of illustration, we fit a log-normal model to it:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(lognormal)
      failure _d: fracture
      analysis time _t: time1
                   id: id

Fitting constant-only model:
Iteration 0:  log likelihood = -69.784524
Iteration 1:  log likelihood = -68.743214
Iteration 2:  log likelihood = -59.663856
Iteration 3:  log likelihood = -59.511256
Iteration 4:  log likelihood = -59.510919
Iteration 5:  log likelihood = -59.510919

Fitting full model:
Iteration 0:  log likelihood = -59.510919 (not concave)
Iteration 1:  log likelihood = -48.387777
Iteration 2:  log likelihood = -42.455265
Iteration 3:  log likelihood = -41.847689
Iteration 4:  log likelihood = -41.845325
Iteration 5:  log likelihood = -41.845325

Log-normal regression -- accelerated failure-time form
No. of subjects =          48          Number of obs   =          106
No. of failures =          31
Time at risk   =          714

Log likelihood = -41.845325          LR chi2(2)        =          35.33
                                      Prob > chi2       =          0.0000
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
protect	1.459569	.246986	5.91	0.000	.9754855 1.943653
age	-.0785641	.0222192	-3.54	0.000	-.1221129 -.0350152
_cons	7.45804	1.59032	4.69	0.000	4.341069 10.57501
/ln_sig	-.2952625	.1268218	-2.33	0.020	-.5438285 -.0466964
sigma	.7443362	.094398			.5805215 .9543771

Although the log-normal model does not fit into the proportional hazards framework, `predict` may still be used after estimation to obtain the predicted hazard function. We can compare the hazards for the `protect==0` and `protect==1` groups at age 70 by typing

```
. replace protect=0
(72 real changes made)
```

```

. replace age=70
(96 real changes made)
. predict h0, hazard
. replace protect=1
(106 real changes made)
. predict h1, hazard
. label var h0 "protect==0"
. label var h1 "protect==1"
. line h0 h1 _t, c(1 1) sort ltitle("h(t)")

```

which produces Figure 13.13.

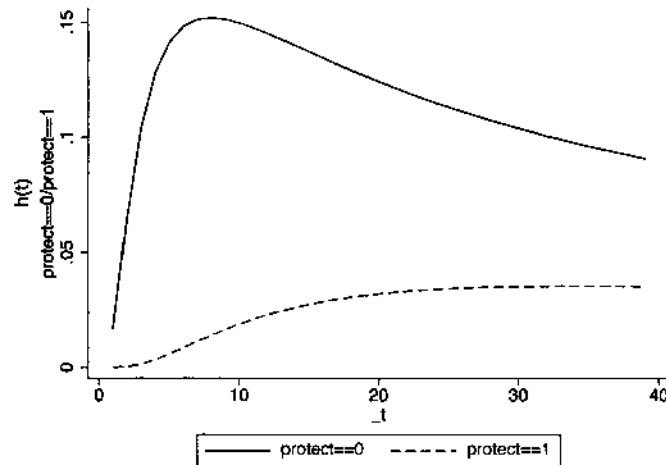


Figure 13.13: Comparison of hazards for a log-normal model

The nonproportional-hazard nature of this model is now obvious. Given what we already know about the hip data, we would argue that both of these hazard functions misspecify what is occurring.

13.5 Log-logistic regression (AFT metric)

In the AFT metric, $\tau_j = \exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j$, and for the log-logistic regression model, we assume that

$$\tau_j \sim \text{Log-logistic}(\beta_0, \gamma)$$

That is, τ_j is distributed as log-logistic with parameters (β_0, γ) with cumulative distribution function

$$F(\tau) = 1 - \left[1 + \{\exp(-\beta_0)\tau\}^{\frac{1}{\gamma}} \right]^{-1} \quad (13.10)$$

Thus,

$$\begin{aligned}\ln(t_j) &= \mathbf{x}_j\boldsymbol{\beta}_x + \ln(\tau_j) \\ &= \beta_0 + \mathbf{x}_j\boldsymbol{\beta}_x + u_j\end{aligned}$$

where u_j follows a logistic distribution with mean 0 and standard deviation $\pi\gamma/\sqrt{3}$. As a result,

$$E\{\ln(t_j)|\mathbf{x}_j\} = \beta_0 + \mathbf{x}_j\boldsymbol{\beta}_x$$

Alternatively, we can derive the AFT formulation by accelerating the effect of time on survival experience. At baseline values of the covariates \mathbf{x} , $\tau_j = t_j$ since all covariates are equal to zero. Thus, the baseline survivor function of t_j is obtained from (13.10) to be

$$S_0(t_j) = \left[1 + \{\exp(-\beta_0)t_j\}^{\frac{1}{\gamma}}\right]^{-1}$$

In an AFT model, the effect of the covariates is to accelerate time by a factor of $\exp(-\mathbf{x}_j\boldsymbol{\beta}_x)$. Thus, for the AFT model,

$$\begin{aligned}S(t_j|\mathbf{x}_j) &= S_0\{\exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j\} \\ &= \left[1 + \{\exp(-\beta_0)\exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j\}^{\frac{1}{\gamma}}\right]^{-1} \\ &= \left[1 + \{\exp(-\beta_0 - \mathbf{x}_j\boldsymbol{\beta}_x)t_j\}^{\frac{1}{\gamma}}\right]^{-1}\end{aligned}$$

The log-logistic distribution closely resembles the log-normal distribution, and some examples of log-logistic hazards are given in Figure 13.14.

(Continued on next page)

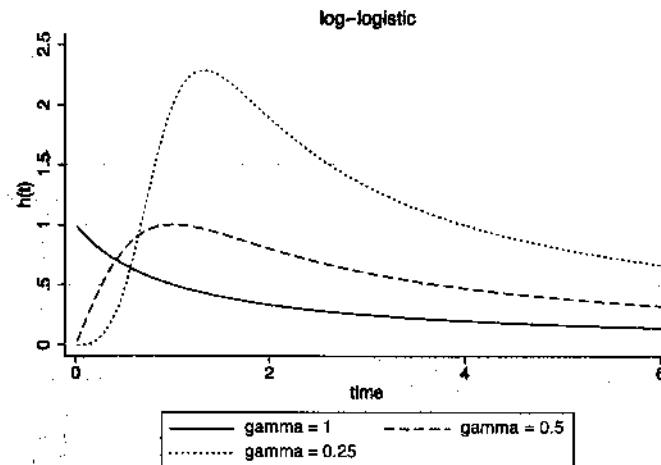


Figure 13.14: Examples of log-logistic hazard functions ($\beta_0 = 0$)

Like the log-normal model, the log-logistic model has no natural PH interpretation. One advantage of the log-logistic model over the log-normal model is that the log-logistic model has simpler mathematical expressions of the hazards and survivor functions, expressions that do not include the normal cumulative distribution function. If $\gamma < 1$, the log-logistic hazard increases and then decreases. If $\gamma \geq 1$, then the hazard is monotone decreasing.

Using our hip-fracture data,

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(llogistic)
      failure _d: fracture
      analysis time _t: time1
      id: id

Fitting constant-only model:
Iteration 0:  log likelihood = -69.090852
Iteration 1:  log likelihood = -60.599923
Iteration 2:  log likelihood = -59.3027
Iteration 3:  log likelihood = -59.298893
Iteration 4:  log likelihood = -59.298892

Fitting full model:
Iteration 0:  log likelihood = -59.298892 (not concave)
Iteration 1:  log likelihood = -48.606502
Iteration 2:  log likelihood = -42.884806
Iteration 3:  log likelihood = -42.242365
Iteration 4:  log likelihood = -42.240535
Iteration 5:  log likelihood = -42.240534
```


Log-logistic regression -- accelerated failure-time form

No. of subjects =	48	Number of obs	=	106
No. of failures =	31			
Time at risk =	714			

		LR chi2(2)	=	34.12
Log likelihood	=	-42.240534	Prob > chi2	= 0.0000

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
protect	1.434467	.2483224	5.78	0.000	.9477643	1.92117
age	-.0755823	.0221193	-3.42	0.001	-.1189352	-.0322293
_cons	7.284475	1.562057	4.66	0.000	4.2229	10.34605
/ln_gam	-.8566429	.1476461	-5.80	0.000	-1.145924	-.5671619
gamma	.4246275	.0626946			.3179301	.5671327

We obtain results nearly identical to those produced under the log-normal model and every bit as inappropriate for these data, and in fact, a comparison of the hazards for `protect==0` and `protect==1` with `age` held at 70 years,

```
. replace protect=0
(72 real changes made)

. replace age=70
(96 real changes made)

. predict h0, hazard

. replace protect=1
(106 real changes made)

. predict h1, hazard

. label var h0 "protect==0"

. label var h1 "protect==1"

. line h0 h1 _t, c(1 1) sort l1title("h(t)")
```

will produce a graph nearly identical to Figure 13.13.

In point of fact, the log-logistic and log-normal models are very similar, and for most purposes are indistinguishable, much like probit and logistic regression models for binary data. Returning to our linear model for $\ln(t_j)$,

$$\ln(t_j) = \beta_0 + \mathbf{x}_j\beta_{\mathbf{x}} + u_j$$

It was noted that for the log-normal model, u_j is normal with mean 0 and standard deviation σ ; for the log-logistic model, u_j is logistic with mean 0 and standard deviation $\pi\gamma/\sqrt{3}$. For the above data, $\hat{\gamma} = 0.425$, and thus $\pi\hat{\gamma}/\sqrt{3} = 0.771$, which nearly equals $\hat{\sigma} = 0.744$, estimated in Section 13.4.

13.6 Generalized gamma regression (AFT metric)

In the AFT metric, $\tau_j = \exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j$, and for gamma regression models, we assume that

$$\tau_j \sim \text{Gamma}(\beta_0, \kappa, \sigma)$$

That is, τ_j is distributed as generalized gamma with parameters $(\beta_0, \kappa, \sigma)$ with cumulative distribution function

$$F(\tau) = \begin{cases} I(\gamma, u), & \text{if } \kappa > 0 \\ \Phi(z), & \text{if } \kappa = 0 \\ 1 - I(\gamma, u), & \text{if } \kappa < 0 \end{cases} \quad (13.11)$$

where $\gamma = |\kappa|^{-2}$, $z_0 = \text{sign}(\kappa)\{\ln(\tau) - \beta_0\}/\sigma$, $u = \gamma \exp(\sqrt{\gamma}z_0)$, $\Phi()$ is the standard normal cumulative distribution function, and $I(a, x)$ is the incomplete gamma function

$$I(a, x) = \frac{1}{\Gamma(a)} \int_0^x e^{-v} v^{a-1} dv$$

Thus,

$$\begin{aligned} \ln(t_j) &= \mathbf{x}_j\boldsymbol{\beta}_x + \ln(\tau_j) \\ &= \beta_0 + \mathbf{x}_j\boldsymbol{\beta}_x + u_j \end{aligned}$$

where u_j has mean

$$E(u_j) = \frac{\sigma\Gamma(\gamma)}{\sqrt{\gamma}\Gamma'(\gamma)} + \ln(\gamma)$$

As a result,

$$E\{\ln(t_j)|\mathbf{x}_j\} = \beta_0 + \mathbf{x}_j\boldsymbol{\beta}_x + E(u_j)$$

Alternatively, we can derive the AFT formulation by accelerating the effect of time on survival experience. At baseline values of the covariates \mathbf{x} , $\tau_j = t_j$ since all covariates are equal to zero. Thus, the baseline survivor function of t_j is

$$S_0(t_j) = 1 - F(t_j)$$

where $F()$ is given in (13.11). Thus,

$$\begin{aligned} S(t_j|\mathbf{x}_j) &= S_0\{\exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j\} \\ &= 1 - F\{\exp(-\mathbf{x}_j\boldsymbol{\beta}_x)t_j\} \\ &= 1 - F^*(t_j) \end{aligned}$$

where $F^*(\cdot)$ is $F(\cdot)$ from (13.11) with z_0 replaced by

$$z = \text{sign}(\kappa) \frac{\ln(\tau) - (\beta_0 + \mathbf{x}_j \beta_x)}{\sigma}$$

The generalized gamma distribution is a three-parameter distribution $(\beta_0, \kappa, \sigma)$ possessing a highly flexible hazard function that allows for a large number of possible shapes. The gamma distribution includes as special cases the Weibull if $\kappa = 1$, in which case $p = 1/\sigma$; the exponential distribution if $\kappa = \sigma = 1$; and the log-normal distribution if $\kappa = 0$. As such, the generalized gamma model is commonly used for evaluating and selecting an appropriate parametric model for the data. For example,

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(gamma)
      failure _d: fracture
      analysis time _t: timel
      id: id

Fitting constant-only model:
Iteration 0:  log likelihood = -63.838211
Iteration 1:  log likelihood = -62.510044 (not concave)
Iteration 2:  log likelihood = -59.5327
Iteration 3:  log likelihood = -59.095428
Iteration 4:  log likelihood = -59.074315
Iteration 5:  log likelihood = -59.074271
Iteration 6:  log likelihood = -59.074271

Fitting full model:
Iteration 0:  log likelihood = -59.074271 (not concave)
Iteration 1:  log likelihood = -46.552166
Iteration 2:  log likelihood = -44.146478 (not concave)
Iteration 3:  log likelihood = -42.041913
Iteration 4:  log likelihood = -41.485857
Iteration 5:  log likelihood = -41.483472
Iteration 6:  log likelihood = -41.483472

Gamma regression -- accelerated failure-time form
No. of subjects =          48          Number of obs  =          106
No. of failures =          31
Time at risk   =          714

Log likelihood = -41.483472          LR chi2(2)      =          35.18
                                      Prob > chi2     =          0.0000
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
protect	1.4067	.2573213	5.47	0.000	.9023597	1.911041
age	-.0727836	.02328	-3.13	0.002	-.1184116	-.0271556
_cons	7.223275	1.60255	4.51	0.000	4.082335	10.36422
/ln_sig	-.3836558	.1842131	-2.08	0.037	-.7447069	-.0226047
/kappa	.4644343	.576288	0.81	0.420	-.6650695	1.593938
sigma	.6813659	.1255165			.4748735	.9776488

and we note that the 95% confidence interval for κ ($-0.665, 1.594$) includes both 0 and 1, and thus does not rule out a Weibull model ($\kappa = 1$) or a log-normal model ($\kappa = 0$), which is not surprising given what we know about these data.

As such, a graph of the comparison of hazards for `protect==0` versus `protect==1` via

```
. replace protect=0
(72 real changes made)
. replace age=70
(96 real changes made)
. predict h0, hazard
. replace protect=1
(106 real changes made)
. predict h1, hazard
. label var h0 "protect==0"
. label var h1 "protect==1"
. line h0 h1 _t, c(1 1) sort ltitle("h(t)")
```

will produce a graph similar to that given for the log-normal (Figure 13.13).

13.7 Choosing among parametric models

We may ask ourselves, given that we have several possible parametric models to choose from, how can we select one? The preferred answer is that the science for the problem at hand suggests an appropriate parametric model. If we think carefully about the underlying process that generated the failure times in our data and specifically about the possible shape of the hazard function, we can get a good idea of which parametric model(s) we should evaluate.

From a purely statistical view, several strategies are available for selecting a parametric model, of which we will consider two:

1. When models are nested, the likelihood-ratio or Wald tests can be used to discriminate between them. This can certainly be done in the case of Weibull versus exponential, or gamma versus Weibull or log-normal.
2. When models are not nested, the likelihood-ratio and Wald tests are unsuitable, and we can use, for instance, the Akaike (1974) information criterion (AIC).

13.7.1 Nested models

Using our hip-fracture data, we first fit a generalized gamma model (the most general of the models available in `streg`) and test the following hypotheses:

1. $H_0: \kappa = 0$, in which case if H_0 is true then the model is log-normal.
2. $H_0: \kappa = 1$, in which case if H_0 is true then the model is Weibull.

3. $H_0: \kappa = 1, \sigma = 1$, in which case if H_0 is true then the model is exponential (constant baseline hazard function).

We start by fitting the gamma model:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(gamma) nolog
      failure _d: fracture
      analysis time _t: time1
      id: id
```

Gamma regression -- accelerated failure-time form

No. of subjects =	48	Number of obs =	106
No. of failures =	31		
Time at risk =	714		

Log likelihood =	-41.483472	LR chi2(2) =	35.18
		Prob > chi2 =	0.0000

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
protect	1.4067	.2573213	5.47	0.000	.9023597 1.911041
age	-.0727836	.02328	-3.13	0.002	-.1184116 -.0271556
_cons	7.223275	1.60255	4.51	0.000	4.082335 10.36422
/ln_sig	-.3836558	.1842131	-2.08	0.037	-.7447069 -.0226047
/kappa	.4644343	.576288	0.81	0.420	-.6650695 1.593938
sigma	.6813659	.1255165			.4748735 .9776488

We can now perform our three tests:

- (1) $H_0: \kappa = 0$

We can read the Wald test directly from the output since `streg` reports that the test statistic for `/kappa` is $z = 0.81$ with significance level 0.420. Alternatively, we could perform the Wald test ourselves using `test`:

```
. test [kappa]_b[_cons] = 0
      ( 1)  [kappa]_cons = 0
             chi2( 1) =    0.77
             Prob > chi2 =    0.3798
```

The test statistic looks different because in this case, a chi-squared is reported and `streg` reported a normal, but these tests are identical, as illustrated by the identical levels of significance.

We could also perform a likelihood-ratio test, which in this context is asymptotically equivalent to the Wald test, and the choice between the two is a matter of personal taste. Wald tests have the advantage that they can be used with sampling weights and robust estimates of the variance-covariance matrix of the parameters, whereas likelihood-ratio tests cannot. Since we specify neither sampling weights or robust standard errors in this model, however, the likelihood-ratio

test is appropriate. For nested models (in this case, the log-normal nested within the gamma), the procedure for performing a likelihood-ratio test is as follows:

```
. streg protect age, dist(gamma)          /* Fit the saturated model */
(output omitted)
. estimates store sat                      /* Save the results */
. streg protect age, dist(lnormal)        /* Fit the nested model */
(output omitted)
. lrtest sat, force                       /* Likelihood-ratio test */
likelihood-ratio test                     LR chi2(1) =      0.72
(assumption: . nested in sat)             Prob > chi2 =    0.3949
```

This compares favorably to the asymptotically equivalent Wald test.

□ Technical Note

Note that in order to perform the likelihood-ratio test, we needed to specify the `force` option to `lrtest`; otherwise, Stata would refuse to perform this test. Normally, using `force` is not a good idea, since you are overriding Stata's judgment that the test has no statistical validity. In this case, however, Stata simply does not know that the models are nested because `streg` can fit lots of different models, and some of the combinations are not nested.

□

(2) $H_0: \kappa = 1$

For this test, we will obtain the Wald test only, but first we need to refit our gamma model:

```
. quietly streg protect age, dist(gamma) nolog
. test [kappa]_b[_cons] = 1
(1) [kappa]_cons = 1
      chi2( 1) =    1.03
      Prob > chi2 =    0.3111
```

We note that the test results do not preclude the use of the Weibull model for these data; these results agree with what we already know about the nature of the hazard for hip fracture.

(3) $H_0: \kappa = 1, \sigma = 1$

As in (1) and (2), we can perform either a likelihood-ratio test or a Wald test, but we opt for a Wald test so that we can demonstrate how to use `test` to test two parameters simultaneously:

```
. test [kappa]_b[_cons] = 1, notest
(1) [kappa]_cons = 1
. test [ln_sig]_b[_cons] = 0, accum
(1) [kappa]_cons = 1
(2) [ln_sig]_cons = 0
      chi2( 2) =   15.86
      Prob > chi2 =    0.0004
```

Note that `streg, dist(gamma)` estimated $\ln(\sigma)$ instead of σ , so to test $\sigma = 1$, we needed to test $\ln(\sigma) = 0$. The test results strongly reinforce what we already know, that the risk of hip fracture is certainly not constant over time.

Concerning the log-normal and Weibull, recall that in Section 13.1.1 we used exponential regression to fit a step function and argued that the hazard was increasing and perhaps at a decreasing rate, that we found that the Weibull fit this exponentially estimated step function well, and that when we produced log-normal results, we made comments that the increasing and then decreasing hazard were inappropriate for these data. However, the statistical test we just performed cannot reject the log-normal specification. Furthermore, the Weibull and log-normal are not nested models, and so a direct test such as the above to determine which is more suitable is not feasible.

When we fit our exponential step function, we did not put standard errors around our estimates of the steps (although `streg` reported them). If we had more data and if the pattern we think we saw persists, then the statistical test would reject the log-normal. The problem is that we do not have more data and cannot, on the basis of these data, justify our supposition that the hazard is monotonically increasing. Nevertheless, we would have no difficulty rejecting that assumption on the basis of our science, or at least based on our prior knowledge of hip fracture among elderly women.

13.7.2 Non-nested models

In the case of non-nested models, Akaike (1974) proposed penalizing each model's log likelihood to reflect the number of parameters being estimated and then comparing them. Although the best-fitting model is the one with the largest log likelihood, the preferred model is the one with the lowest value of the Akaike Information Criterion (AIC). For parametric survival models, the AIC is defined as

$$\text{AIC} = -2 \ln L + 2(k + c)$$

where k is the number of model covariates and c the number of model-specific distributional parameters.

For the models fit by Stata's `streg`, the values of c are given in Table 13.1.

Table 13.1: Comparison of `streg` models

Distribution	Metric	Hazard Shape	c
Exponential	PH, AFT	constant	1
Weibull	PH, AFT	monotone	2
Gompertz	PH	monotone	2
Log-normal	AFT	variable	2
Log-logistic	AFT	variable	2
Generalized gamma	AFT	variable	3

To illustrate the Akaike information criterion (AIC), we review the six parametric models as fit on the hip-fracture data with covariates `age` and `protect`. Table 13.2 gives the log likelihoods and Akaike information criterion values from each model:

Table 13.2: Comparison of AIC values for `streg` models

Distribution	Log Likelihood	k	c	AIC
Exponential	-47.534656	2	1	101.06931
Weibull	-41.992704	2	2	91.98541
Gompertz	-42.609352	2	2	93.21870
Log-normal	-41.845345	2	2	91.69065
Log-logistic	-42.240534	2	2	92.48107
Generalized gamma	-41.483472	2	3	92.96694

Per the AIC criterion, the log-normal model is selected. Note, however, that the more reasonable (based on the science of hip fractures) Weibull model has virtually the same AIC score, 91.99 versus 91.69, and on that combined basis, we would choose the Weibull.

14 Post-estimation commands for parametric models

14.1 Use of predict after streg

`predict` after `streg` is used to generate a new variable containing predicted values or residuals, and what is calculated is determined by the option you specify; see Table 14.1 for a list of what is available.

Table 14.1: Options for predict after `streg`

Option	Contents
<code>xb</code>	$\hat{\beta}_0 + \mathbf{x}_j \hat{\beta}_x$
<code>stdp</code>	standard error of $\hat{\beta}_0 + \mathbf{x}_j \hat{\beta}_x$
<code>median time</code>	predicted median survival time
<code>median lntime</code>	predicted median $\ln(\text{survival time})$
<code>mean time</code>	predicted mean survival time
<code>mean lntime</code>	predicted mean $\ln(\text{survival time})$
<code>hazard</code>	predicted $h(\cdot, \mathbf{t})$
<code>hr</code>	predicted hazard ratio
<code>surv</code>	predicted $S(\cdot, \mathbf{t} \mathbf{t}_0)$
<code>csurv</code>	predicted $S(\cdot, \mathbf{t} \text{earliest } \mathbf{t}_0 \text{ for the subject})$
<code>csnell</code>	(partial) Cox–Snell residuals
<code>ccsnell</code>	cumulative Cox–Snell residuals
<code>mgale</code>	(partial) martingale residuals
<code>cmgale</code>	cumulative martingale residuals
<code>deviance</code>	deviance residuals

Calculations are obtained from the data currently in memory and do not need to correspond to the data used in fitting the model. This is true for all the calculations `predict` can make following `streg`. Below, we give a more detailed list:

- `xb` --- linear prediction
`xb` calculates $\hat{\beta}_0 + \mathbf{x}_j \hat{\beta}_x$, the linear prediction.

- **stdp** — standard error of the linear prediction
stdp calculates the estimated standard error of $\hat{\beta}_0 + \mathbf{x}_j \hat{\beta}_x$ based on the estimated variance-covariance matrix of $(\hat{\beta}_0, \hat{\beta}_x)$.
- **median time** — predicted median survival time
This is the default prediction; that is, if you do not specify any options, **predict** will calculate the predicted median survival time, $\hat{Q}(0.5|\mathbf{x}_j)$, reported in analysis-time units. Note that the prediction is made from time $t = 0$ conditional on constant covariates, and thus if you have multiple-record-per-subject data with time varying covariates, each record will produce a distinct predicted median survival time.
- **median lntime** — predicted median ln(survival time)
The predicted median ln(survival time) is reported in ln(analysis time) units. As is the case with **median time**, predictions are from time $t = 0$ conditional on constant covariates, even in multiple-record data.
- **mean time** — predicted mean survival time
The predicted mean survival time is reported in analysis-time units and is calculated as the integral of the survival function from 0 to infinity, given each observation's covariate values and the estimated model parameters.
- **mean lntime** — predicted mean ln(survival time)
The predicted mean ln(survival time) is reported in analysis-time units and is calculated as the expected value of the distribution of ln(survival time), given each observation's covariate values and the estimated model parameters.
- **hazard** — predicted hazard function
Calculates $h(_t|\mathbf{x}_j)$ given the estimated model parameters. Note that this calculation is not restricted to models fit in the PH metric.
- **hr** — predicted hazard ratio
Calculates the relative hazard, $\exp(\mathbf{x}_j \hat{\beta}_x)$, for all models that may be parameterized as PH. This option is not allowed with AFT models. For the exponential and Weibull models (which may be cast in both metrics), the predicted hazard ratio is calculated regardless of the metric under which the model was fit; that is, the parameter estimates used in the above are from the PH version of the model.
- **surv** — predicted conditional survivor function
Calculates $S(_t|_t0, \mathbf{x}_j)$, which is the probability of survival past time $_t$ given survival to time $_t0$, given \mathbf{x}_j for each observation and the estimated model parameters. Note that $_t0$ and $_t$ are specific to each observation.
- **csurv** — predicted survivor function
Calculates a running product within each subject of the predictions produced by **predict**, **surv**. This amounts to the probability of survival past $_t$ for each subject, given that the subject survives any gaps for which it is left unobserved.

- **csnell** — (partial) Cox-Snell residuals
If you have single observations per subject, then **csnell** calculates the usual Cox-Snell residuals from (11.1). Otherwise, it calculates the additive contribution to this observation to the subject's overall Cox-Snell residual.
- **ccsnell** — cumulative Cox-Snell residuals
If you have single observations per subject, then **ccsnell** is equivalent to **csnell**. Otherwise, in the last observation (with respect to analysis time) on each subject, **ccsnell** records the sum of the partial Cox-Snell residuals and sets the other observations to missing. Only one value per subject is recorded—the overall sum—and it is placed on the last record for the subject.
- **mgale** — (partial) martingale-like residuals
If you have single observations per subject, then **mgale** calculates the usual martingale residuals. Otherwise, it calculates the additive contribution of this observation to the subject's overall martingale residual. We use the term “martingale-like” because, although these residuals do not arise naturally from martingale theory in the case of parametric survival models as they do for the Cox proportional hazard model, they do share similar properties.
- **cmgale** — cumulative martingale-like residuals
If you have single observations per subject, then **cmgale** is equivalent to **mgale**. Otherwise, in the last observation on each subject, **cmgale** records the sum of **mgale** and sets the other observations to missing. Only one value per subject is recorded—the overall sum—and it is placed on the last record for the subject.
- **deviance** — deviance residuals
deviance calculates the deviance residuals, which are a scaling of the martingale residuals to make them symmetric about zero. In the case of multiple-record-per-subject data, only one value per subject is calculated, and it is placed on the last record for the subject.

14.1.1 Predicting the time of failure

After estimation with **streg**, you can type

1. **predict newvar, time** to obtain the predicted time of failure given x_j .
2. **predict newvar, lntime** to obtain the predicted $\ln(\text{time of failure})$ given x_j .

If neither **median** nor **mean** is specified, the option **median** is assumed, but specifying one of these options removes any ambiguity, and so we highly recommend it.

Combining these options gives four (potentially) distinct predictions. To demonstrate, we fit a Weibull model to our hip-fracture data, **predict** using the four combinations of these options, and list these predictions (along with the covariates) for the last observation.

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(weibull)
(output omitted)
. predict t_mean, time mean
. predict t_median, time median
. predict lnt_mean, lntime mean
. predict lnt_median, lntime median
. list _t protect age t_mean t_median lnt_mean lnt_median in 1, abbrev(10)
```

	_t	protect	age	t_mean	t_median	lnt_mean	lnt_median
106.	39	1	67	41.60789	37.46217	3.497923	3.623332

Before we proceed with analyzing each of these predictions, let's fit the model again, this time in the AFT metric:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(weibull) time nolog
      failure _d: fracture
      analysis time _t: time1
      id: id
Weibull regression -- accelerated failure-time form
No. of subjects =          48          Number of obs   =          106
No. of failures =          31
Time at risk   =          714
Log likelihood  = -41.992704          LR chi2(2)       =          34.61
                                          Prob > chi2      =          0.0000
```

	_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
	protect	1.313965	.2366229	5.55	0.000	.8501928	1.777737
	age	-.0659554	.0221171	-2.98	0.003	-.1093041	-.0226067
	_cons	6.946524	1.575708	4.41	0.000	3.858192	10.03486
	/ln_p	.5188694	.1376486	3.77	0.000	.2490831	.7886556
	p	1.680127	.2312671			1.282849	2.200436
	1/p	.5951931	.0819275			.4544553	.7795152

```
. predict t_mean, time mean
. predict t_median, time median
. predict lnt_mean, lntime mean
. predict lnt_median, lntime median
. list _t protect age t_mean t_median lnt_mean lnt_median in 1, abbrev(10)
```

	_t	protect	age	t_mean	t_median	lnt_mean	lnt_median
106.	39	1	67	41.60789	37.46217	3.497923	3.623332

We did that just to demonstrate that nothing changes—it really is the same model—and `predict` was smart enough to take parameter estimates in either metric and apply them correctly when calculating the prediction. When we take a closer look at how these calculations are performed, we can switch between thinking in the PH and AFT metrics freely.

Looking at our `predict` commands one at a time,

1. `predict t_mean, time mean:`

This gives the expected value of survival time, otherwise known as the first moment of the distribution of survival time. Letting T_j be the time to failure for a random observation with covariate values \mathbf{x}_j , the expected value of T_j is given by

$$\begin{aligned}\mu_{T_j} = E(T_j|\mathbf{x}_j) &= \int_0^\infty t f(t|\mathbf{x}_j) dt \\ &= \int_0^\infty S(t|\mathbf{x}_j) dt\end{aligned}\quad (14.1)$$

where $f()$ is the probability density function of T and $S()$ is the survivor function, both determined by the choice of parametric model. The “prediction” aspect of the calculation has to do with the fact that the model parameters that characterize $f()$ and $S()$ are replaced by their maximum likelihood estimates.

In the case of the Weibull AFT model above, from (13.6) we have

$$S(t|\mathbf{x}_j) = \exp[-\{\exp(-\beta_0 - \mathbf{x}_j\beta_x)t\}^p] \quad (14.2)$$

Equivalently, we could use the Weibull PH survivor function, but we will stick with the AFT parameterization since those are the parameter estimates that we currently have handy. Performing the integration in (14.1) gives

$$\mu_{T_j} = \exp\{p(\beta_0 + \mathbf{x}_j\beta_x)\} \Gamma\left(1 + \frac{1}{p}\right)$$

where $\Gamma()$ is the gamma function. Thus, for our model the predicted mean survival time becomes

$$\hat{\mu}_{T_j} = \exp\left\{\hat{p}\left(\hat{\beta}_0 + \hat{\beta}_1\text{protect}_j + \hat{\beta}_2\text{age}_j\right)\right\} \Gamma\left(1 + \frac{1}{\hat{p}}\right)$$

and plugging in the estimated model parameters and the values of `protect` and `age` from the last observation in our data yields $\hat{\mu}_{T_{106}} = 41.60789$, which is what is given by `predict`.

2. `predict t_median, time median:`

This gives the median, or 50th percentile, of survival time. From (14.2) we can derive the quantile function $Q(u|\mathbf{x}_j)$ for the Weibull AFT model by noting that since T_j is continuous, $Q(u|\mathbf{x}_j) = t$ if and only if $1 - S(t|\mathbf{x}_j) = u$. Thus,

$$Q(u|\mathbf{x}_j) = \exp(\beta_0 + \mathbf{x}_j\beta_x) \{-\ln(1 - u)\}^{1/p}$$

The median of T_j is

$$\tilde{\mu}_{T_j} = Q(0.5|\mathbf{x}_j)$$

and is estimated by plugging in the covariate values \mathbf{x}_j and the parameter estimates. For our model,

$$\hat{\tilde{\mu}}_{T_j} = \exp \left(\hat{\beta}_0 + \hat{\beta}_1 \text{protect}_j + \hat{\beta}_2 \text{age}_j \right) \{\ln(2)\}^{1/\hat{p}} \quad (14.3)$$

and plugging in the estimated model parameters and the values of `protect` and `age` from the last observation in our data yields $\hat{\tilde{\mu}}_{T_{106}} = 37.46217$, which is what is given by `predict`.

3. `predict lnt_mean, lntime mean`:

This predicts the expected value of $\ln(T_j)$ and can be derived in many ways. The most direct way would be a generalization of (14.1), which instead of integrating t times the density, integrates $\ln(t)$ times the density:

$$E\{\ln(T_j)\} = \int_0^\infty \ln(t) f(t|\mathbf{x}_j) dt$$

For models that are parameterized as AFT, however, a better approach would be to derive the distribution of $\ln(T_j)$ via a change of variable and directly calculate the mean of this new distribution. With AFT models, the change of variable recasts the model as one that is linear in the regressors, and thus the calculation of the mean reduces to calculating the mean of a residual term.

For example, our Weibull AFT model can alternatively be expressed as

$$\ln(T_j) = \beta_0 + \mathbf{x}_j \beta_x + u_j \quad (14.4)$$

where the residual term u_j follows the extreme-value distribution with shape parameter p , such that $E(u_j) = \Gamma'(1)/p$, where $\Gamma'(1)$ is Euler's constant. Therefore,

$$\begin{aligned} E\{\ln(t_j)\} &= \beta_0 + \mathbf{x}_j \beta_x + E(u_j) \\ &= \beta_0 + \mathbf{x}_j \beta_x + \Gamma'(1)/p \end{aligned}$$

and in the case of our model, this becomes

$$\hat{E}\{\ln(T_j)\} = \hat{\beta}_0 + \hat{\beta}_1 \text{protect}_j + \hat{\beta}_2 \text{age}_j + \Gamma'(1)/\hat{p}$$

Plugging in our estimates and covariate values yields $\hat{E}\{\ln(T_{106})\} = 3.497923$, which is what is obtained by `predict`.

For other AFT models such as the log-normal and the log-logistic, $E(u_j) = 0$ when the model is recast in the log-linear form, and thus $E\{\ln(T_j)\}$ is simply the linear predictor. For the Weibull (and exponential as a special case) model, however,

$E(u_j)$ is nonzero, and in many instances, researchers choose to ignore this fact and predict $\ln(\text{time})$ using only the linear predictor. Even though this prediction is biased, it is widely used. In any case, if this alternate predicted $\ln(\text{time})$ is desired, it is easily obtained: `predict newvar, xb` will give the linear predictor.

Note that if you exponentiate a predicted mean of $\ln(T_j)$ in order to predict survival time, what you get is not equal to the mean of T_j but is rather another prediction of time to failure altogether. In addition, you can also exponentiate the linear predictor (i.e., ignore the nonzero mean residual) and produce yet another predicted time to failure.

```
. gen e_lmean = exp(lnt_mean)
. predict xb, xb
. gen e_xb = exp(xb)
. list _t protect age t_mean t_median e_lmean e_xb in 1
```

	_t	protect	age	t_mean	t_median	e_lmean	e_xb
106.	39	1	67	41.60789	37.46217	33.04674	46.59427

The four predicted survival times are comparable but not equal. `t_mean` is the predicted mean failure time, `t_median` is the predicted median failure time, and `e_lmean` and `e_xb` are other predictions of failure time obtained via the alternate methods we have described. Each of these four predictions has been used by researchers, and our aim is not to debate the relative merits of each but rather to point out how they are calculated and that they generally do not coincide.

The moral of the story is that if you want to predict survival time, be aware of what the software gives you by default and how you can change the default in order to obtain the prediction you want. In Stata, these four predictions are easily obtained, and given the commands and options required for each, the context is clear.

□ Technical Note

Note that exponentiating a linear predictor in order to predict survival time (`e_xb` in the above) makes sense in an AFT model but not for a PH model, in which case what you obtain are hazard ratios, which most likely have nothing to do with predicting a survival time.

□

4. `predict lnt_median, lntime median`:

The direct calculation of the median of $\ln(T_j)$ would involve the derivation of the distribution of $\ln(T_j)$ from that of T_j via a change of variable. Once the distribution of $\ln(T_j)$ is obtained, you can derive the quantile function for this distribution and use it to evaluate the 50th percentile.

In an AFT model, recasting the model as one that is log-linear in time such as (14.4), the calculation reduces to taking the median of the distribution of the

residual, u_j , and adding $\beta_0 + \mathbf{x}_j\beta_x$ to it. For many A-T models (such as the log-normal and log-logistic), the distribution of u_j is symmetric about zero, and thus $E(u_j) = \text{median}(u_j) = 0$. So, the median of $\ln(T_j)$ equals the mean of $\ln(T_j)$ and is simply the linear predictor in both cases.

The Weibull, however, is not such a model. Given the log-linear form (14.4), recall that for the Weibull u_j follows an extreme-value distribution with shape parameter p . As such,

$$\text{median}(u_j) = \ln\{\ln(2)\}/p$$

and thus our estimate of the median log-survival time for our model becomes

$$\widehat{\text{median}}\{\ln(T_j)\} = \hat{\beta}_0 + \hat{\beta}_1 \text{protect}_j + \hat{\beta}_2 \text{age}_j + \ln\{\ln(2)\}/\hat{p}$$

which equals $\ln(\hat{\mu}_{T_j})$, where $\hat{\mu}_{T_j}$ is given in (14.3).

This is generally true, since $\ln(T_j)$ is a monotone transformation of T_j , and the calculation of the median is invariant to such a transformation. Thus, for our covariates and parameter estimates,

$$\begin{aligned} \widehat{\text{median}}\{\ln(T_{106})\} &= \ln(\hat{\mu}_{T_{106}}) \\ &= \ln(37.46217) \\ &= 3.623332 \end{aligned}$$

which agrees with what `predict` calculates.

Note that since medians are invariant to the log transformation, exponentiating the predicted median of $\ln(\text{survival time})$ does not result in an alternate predicted survival time—it merely reproduces the predicted median survival time, something already considered.

Finally, be careful using `predict`, `time` and `predict, lntime` with multiple-record-per-subject data. Remember that each prediction is made for each record in isolation: the predicted failure time is from time 0 under the assumption of constant covariates. For example,

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect calcium, dist(weibull) time
(output omitted)
. predict t_hat, mean time
. list id _t0 _t protect calcium _d t_hat if id==10
```

	id	_t0	_t	protect	calcium	_d	t_hat
11.	10	0	5	0	9.69	0	8.384427
12.	10	5	8	0	9.47	0	7.956587

The interpretation of `t_hat` in the first observation displayed is that if we had a subject who had `protect==0` and `calcium==9.69`, and those covariates were fixed, then the predicted (mean) time of failure would be 8.38. In the second observation, the prediction is that if we had a subject who was under continual observation from time 0 forward and had the subject fixed covariates of `protect==0` and `calcium==9.47`, the predicted time of failure would be 7.96. Neither prediction really has much to do with subject 10 in our data, given the nature of the time-varying covariate `calcium`.

14.1.2 Predicting the hazard and related functions

After estimation with `streg`, you can type

1. `predict newvar, hazard`
to obtain the predicted hazard.
2. `predict newvar, hr`
to obtain the predicted relative hazard (hazard ratio), assuming that the model has a PH implementation. Otherwise, this option is not allowed.
3. `predict newvar, surv`
to obtain the predicted survivor function conditional on the beginning of an interval (determined by `_t0`).
4. `predict newvar, csurv`
to obtain the (unconditional) predicted survivor function.

It is important to appreciate that all of these calculations (with the exception of `hr`) can be made even for models fit in and with only an AFT parameterization. All survival models may be characterized by their hazard function $h(t|\mathbf{x})$; it just may be that $h(t|\mathbf{x})$ cannot be written according to the proportional hazards decomposition $h_0(t)\exp(\mathbf{x}\beta_x)$. `predict`, however, calculates $h()$ and not $h_0()$, and if you want $h_0()$ because it is appropriate given the model you have fit, you can obtain it by first setting `x = 0` and then predicting $h()$.

The values that `predict, hazard` calculates are $h(_t)$. For example,

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(weibull)
(output omitted)
. predict h, hazard
. list id _t0 _t protect age _d h if id==10 | id==21
```

	id	_t0	_t	protect	age	_d	h
11.	10	0	5	0	73	0	.1397123
12.	10	5	8	0	73	0	.1923367
35.	21	0	5	1	82	0	.041649
36.	21	5	6	1	82	1	.0471474

The above output reports

11. $\hat{h}(5|\text{protect} == 0, \text{age} == 73) = 0.1397$
12. $\hat{h}(8|\text{protect} == 0, \text{age} == 73) = 0.1923$
35. $\hat{h}(5|\text{protect} == 1, \text{age} == 82) = 0.0416$
36. $\hat{h}(6|\text{protect} == 1, \text{age} == 82) = 0.0471$

If we change the data after estimation, `predict` calculates results based on the previously estimated model parameters and the new data in memory,

```
. drop h
. replace age = 70
(96 real changes made)
. replace protect = 0
(72 real changes made)
. predict h, hazard
. list id _t0 _t protect age _d h if id==10 | id==21
```

	id	_t0	_t	protect	age	_d	h
11.	10	0	5	0	70	0	.1001977
12.	10	5	8	0	70	0	.1379384
35.	21	0	5	0	70	0	.1001977
36.	21	5	6	0	70	1	.1134256

and the interpretation of this output is

11. $\hat{h}(5|\text{protect} == 0, \text{age} == 70) = 0.1002$
12. $\hat{h}(8|\text{protect} == 0, \text{age} == 70) = 0.1379$
35. $\hat{h}(5|\text{protect} == 0, \text{age} == 70) = 0.1002$
36. $\hat{h}(6|\text{protect} == 0, \text{age} == 70) = 0.1134$

`predict` always works like this, regardless of the option specified. `predict` calculates results; it does not retrieve results stored at the time of estimation. Changing the data in memory after estimation is a favorite trick to obtain desired calculations. Just remember to save your dataset before you change it.

`predict, surv` and `predict, csurv` calculate current and cumulative survival probabilities using the data currently in memory.

```
. predict s, surv
. predict cs, csurv
```

```
. list id _t0 _t protect age _d s cs if id==10 | id==21
```

	id	_t0	_t	protect	age	_d	s	cs
11.	10	0	5	0	70	0	.7421641	.7421641
12.	10	5	8	0	70	0	.6986433	.518508
35.	21	0	5	0	70	0	.7421641	.7421641
36.	21	5	6	0	70	1	.8986372	.6669363

The interpretation of the previous output (for T = survival time) is given in Table 14.2.

Table 14.2: Use of `predict`, `surv` and `predict`, `csurv`

Obs.	Var.	Calculation	Value
11	s	$\Pr(T > 5 \mid \text{protect} == 0, \text{age} == 70)$	0.7422
12	s	$\Pr(T > 8 \mid \text{protect} == 0, \text{age} == 70, T > 5)$	0.6986
12	cs	$\Pr(T > 8 \mid \text{history})$	$0.7422 \times 0.6986 = 0.5185$
35	s	$\Pr(T > 5 \mid \text{protect} == 0, \text{age} == 70)$	0.7422
35	s	$\Pr(T > 6 \mid \text{protect} == 0, \text{age} == 70, T > 5)$	0.8986
36	cs	$\Pr(T > 6 \mid \text{history})$	$0.7422 \times 0.8986 = 0.6669$

Note that the cumulative probabilities calculated by `predict`, `csurv` are obtained by multiplying (within subject) the `predict`, `surv` results. This means that if the covariates change during saucepans, that change would be reflected in the calculated cumulative probabilities. We labeled the cumulative results as those given the “history” to emphasize that fact and did that even though, in our example, the covariates do not change.

□ Technical Note

This method of calculating cumulative probabilities via multiplication across the covariate history is not without issues. In cases where there are gaps or delayed entry, the net effect of the technique is to assume the probability of survival over the gaps is one, just as it is in the likelihood function. This empirically based notion of survival probabilities can make comparison of subjects with similar covariates more difficult if one of the subjects exhibits gaps and the other does not. (We will see another manifestation of the interpretation problem caused by gaps in Section 14.2.)

□

Finally, you may have noticed that there is no `cumhazard` option to `predict` after `streg` in current Stata. There is really no good reason for this, except to say that obtaining an estimate of the cumulative hazard function at time `_t` is easy given the one-to-one relationship between the survivor function and the cumulative hazard function. One types

```
. predict cs, csurv          /* We want cumulative survival here */
. gen chaz = -ln(cs)         /* H(t) = -ln{S(t)} */
```

14.1.3 Calculating residuals

The residuals calculated by `predict` can be used in the same way as discussed in Chapter 11, where we discussed regression diagnostics for the Cox model. The interpretation is the same in all cases. There is, however, a difference in how (and when) you request the residuals. With the parametric models, the residuals are calculated after estimation using `predict`, whereas with the Cox model most of the residuals must be requested at the time of estimation via options to the `stcox` command. Requests after the fact are more convenient, and the reason `stcox` requires that you request the residuals at the time of estimation has to do with computer efficiency. In the case of Cox regression, the residuals (like the likelihood) are based on calculations involving summations across risk pools, and it is in the identification of those risk pools that the computer spends substantial time. Thus, rather than reforming the risk pools post-estimation, it is more efficient to use the risk pools that were already formed during the estimation.

In any case, the intricacies of diagnostics for Cox models carry over to parametric models for survival data, and thus are not something on which we elaborate in detail. That is not to say that such diagnostics are not important in parametric models; it is just that we (the authors) feel the application of those topics covered in Chapter 11 can be applied to this context in a straightforward manner.

As an example, we fit a Weibull model to our hip data and calculate the (cumulative) Cox-Snell residuals in order to look for outliers:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(weib)
(output omitted)
. predict cc, ccsnell
(58 missing values generated)
. scatter cc _t, mlab(id) s(1)
```

The result is shown in Figure 14.1.

(Continued on next page)

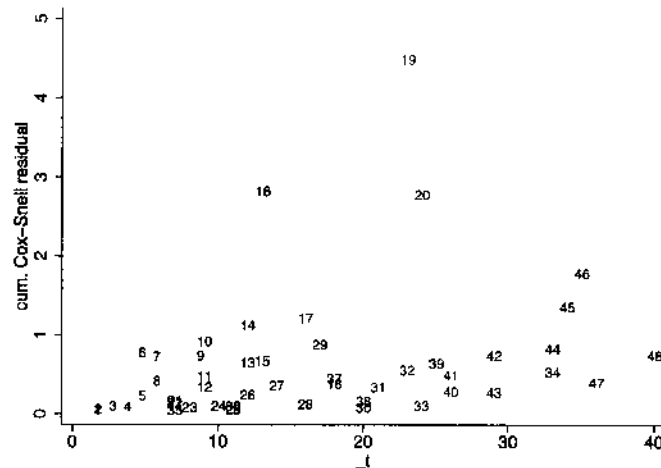


Figure 14.1: Cumulative Cox-Snell residuals for a Weibull model

We see that `id==16`, `19`, & `20` are probable outliers. In looking at outliers, we find it useful to also have the predicted time of failure (the mean is fine) based on the model:

```
. predict pred_t, time mean
. list _t0 _t pred_t age protect _d cc if id==16
```

	_t0	_t	pred_t	age	protect	_d	cc
23.	0	5	5.782123	77	0	0	.
24.	5	12	5.782123	77	0	1	2.819476

We see that this 77-year-old woman failed at time 12, which was a long time after the predicted (mean) time of 5.78. Note also that `cc` is missing in observation 23. We asked for the cumulative Cox-Snell residuals because we had multiple-observation-per-subject data, and when we did that, the summed residual was filled in only on the last record of each subject. That is why `predict` in the previous log said “58 missing values generated”.

Also note that variable `pred_t` has the same value in both observations. As we warned, predicted failure times are made as if covariates are constant. In our model of `age` and `protect`, the covariates are indeed constant and so that is a correct result, but in general, this may not be what we want.

You may be asking yourself, why not just find outliers based on `pred_t`? There are two answers:

1. You can only correctly calculate “residual time” = `_t - pred_t` for those who fail; censored observations can also be outliers.

2. The distribution of `pred.t` can be very nonsymmetric, which makes finding outliers based on residual time difficult.

14.2 Using `stcurve`

`stcurve` is a wonderfully handy command for graphing, after estimation, the estimated survivor functions, hazard functions, and cumulative hazard functions. `stcurve` does nothing you cannot do for yourself using the `predict` and `graph` commands, but `stcurve` makes doing certain things easier.

Remember that `predict` calculates predicted values, for each observation in the data, at that observation's recorded values of the covariates. If you were to type, say, `predict h, hazard` and then `graph h` versus `_t`, what you would see is a hopeless mishmash of predicted hazards, each at different values of the covariates. The solution is to replace, in each observation, the values of the covariates with constant values (such as `protect==0` and `age==70`) before making the prediction and then graph that. Remember, however, to save your dataset first so that you can get your real data back. In Chapter 13, that was exactly the approach we followed for making graphs of the functions, and that approach is well worth learning because it will let you draw any graph of which you can conceive, such as that in Figure 13.8 where we compared Weibull and piecewise exponential hazards.

Most of the time, however, what we want are "simple", graphs and those `stcurve` can draw for us, and, moreover, `stcurve` can draw graphs comparing groups that might otherwise tax to the point of breaking our knowledge of Stata syntax.

`stcurve` can graph

1. The survivor function. Type `stcurve, survival ...`
2. The cumulative hazard function. Type `stcurve, cumhaz ...`
3. The hazard function. Type `stcurve, hazard ...`

`stcurve` can graph any of those functions at the values for the covariates you specify. The syntax is as follows:

```
stcurve, ...at(varname=# varname=# ...)
```

If you do not specify a variable's value, the average value is used, and thus if the `at()` option is omitted altogether, a graph is produced for all the covariates held at their average values. The `at()` option can also be generalized to graph the function evaluated at different values of the covariates on the same graph. The syntax is

```
stcurve, ...at1(varname=# varname=# ...) at2(...) at3(...) ...
```

For example, we can fit a generalized gamma model to our hip data and plot the estimated hazard function for several covariate patterns,

```

. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(gamma)
(output omitted)
. stcurve, hazard /*
>      /* at1(protect=0 age=70) /*
>      /* at2(protect=0 age=75) /*
>      /* at3(protect=0 age=80) /*
>      /* at4(protect=0 age=85)

```

the result of which is shown in Figure 14.2.

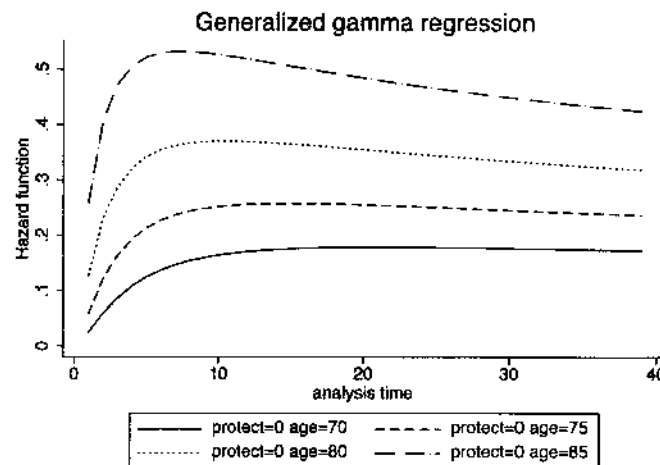


Figure 14.2: Hazard curves for the gamma model (`protect==0`)

In the case of hazard curves for models that are not constrained to fit the PH assumption (such as the gamma), we emphasize that there is not one hazard function but one function per pattern of covariates: $h(t|\mathbf{x})$. In the case of PH models, these functions can be summarized as $h(t|\mathbf{x}) = h_0(t) \exp(\mathbf{x}\beta_x)$. Here, we are being flexible by fitting a gamma model, and we can thus check the proportional hazards assumption graphically.

These graphs only gently violate the proportional hazards assumption; note that the top graph for the oldest group actually turns down, while the bottom graph for the youngest group does not.

The corresponding graph for those with `protect==1` is

```

. stcurve, hazard /*
>      /* at1(protect=1 age=70) /*
>      /* at2(protect=1 age=75) /*
>      /* at3(protect=1 age=80) /*
>      /* at4(protect=1 age=85)

```

and the result is shown in Figure 14.3.

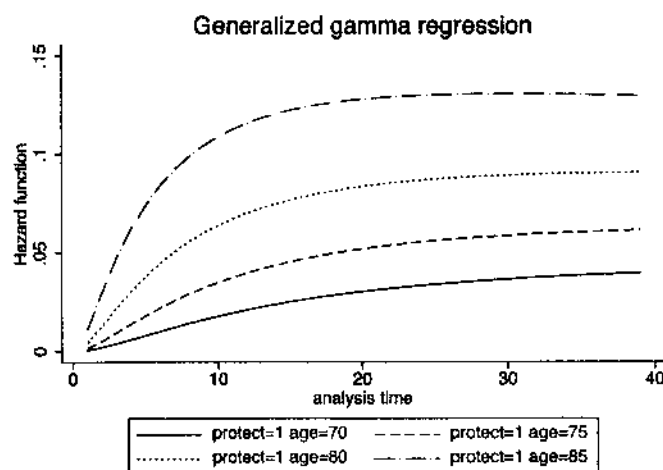


Figure 14.3: Hazard curves for the gamma model (`protect==1`)

In comparing this graph with the previous one, note carefully the different scales of the y -axis.

In a technical note in Section 14.1.2, we mentioned that `predict`, `csurv` calculates the cumulative probability of survival, and that given the way this prediction is calculated, special care must be given when comparing subjects with similar covariate patterns where one subject has gaps and the other does not. Because of gaps, even if two subjects have the same covariate values, they may have different predictions of cumulative survival for any given time. We can see this by plotting the cumulative survival function as calculated using `predict` after setting `age==70` and `protect==0`:

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(weibull)
(output omitted)
. replace age = 70
(96 real changes made)
. replace protect = 0
(72 real changes made)
. predict cs, csurv
. line cs _t, c(1) sort ltitle("Cumulative survival probability")
```

The result is shown in Figure 14.4.

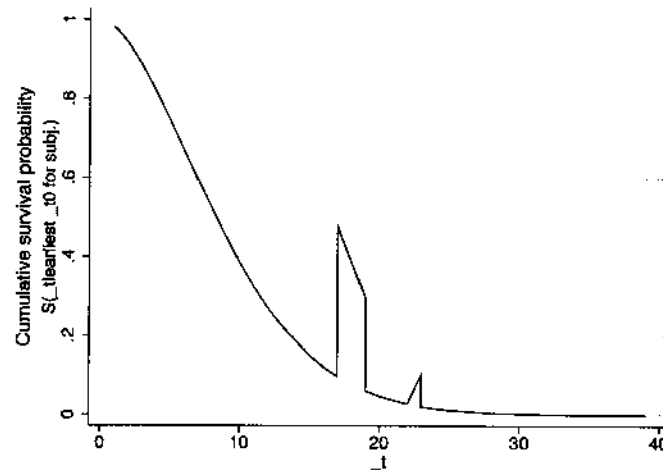


Figure 14.4: Cumulative survival probability as calculated by **predict**

The spikes in the plot indicate that there are time gaps in the data, and even though all observations have the same covariate values (we set them that way), the predicted cumulative survival function is not a smooth function of $_t$.

When covariates are held constant, the cumulative survival probability as calculated by **predict**, *csurv* will equal $S(_t|\mathbf{x})$ only if there are no time gaps in the data. In situations where there are gaps, **stcurve** comes to the rescue. **stcurve** calculates $S(_t|\mathbf{x})$ directly from the parametric form of this function, ignoring the gaps. A superior plot of the predicted $S(_t|\mathbf{x})$ is obtained using

```
. use http://www.stata-press.com/data/cgg/hip2, clear
(hip fracture study)
. streg protect age, dist(weibull)
(output omitted)
. stcurve, survival at(protect=0 age=70)
```

The result is shown in Figure 14.5.

(Continued on next page)

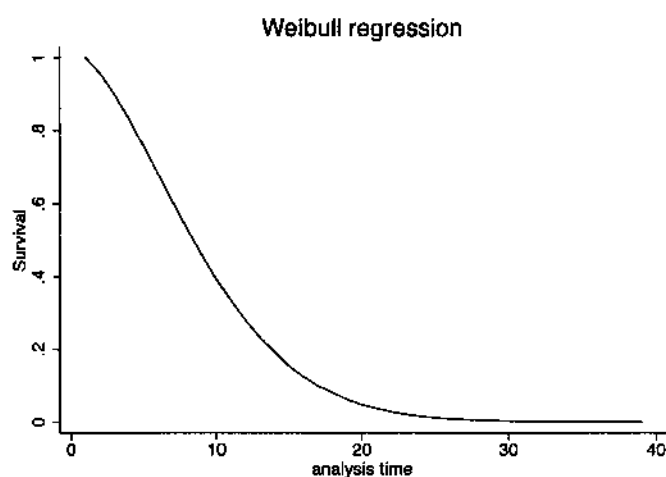


Figure 14.5: Survival function as calculated by `stcurve`

Given what we have seen, our advice is to use `predict, surv` if you want individual-specific survival probabilities that take gaps and time-varying covariates into account. Use `stcurve, survival` if you want to plot the estimated overall survival function.

15 Generalizing the parametric regression model

In Chapter 13, we discussed the six parametric models for survival data available in Stata's `streg` command, namely the exponential, Weibull, log-normal, log-logistic, Gompertz, and generalized gamma. These models represent the basis of parametric survival models available in Stata. In this chapter, we discuss other aspects and options to `streg` that serve to expand and generalize these models.

15.1 Using the `ancillary()` option

The `ancillary()` option is how you specify linear predictors for the “other” parameters of the assumed distribution. For instance, the Weibull distribution has one ancillary parameter, p , which is in most cases fitted as a constant. p has been constant in every example we have considered so far in this book. By specifying `ancillary(male)`, for instance, you could specify that $p = p_0 + p_1\text{male}$, and if `male` were an indicator variable that assumed the value 0 for females and 1 for males, you would effectively be saying that p has one constant value for males and another constant value for females. Why you might want to do that we will explore below.

In terms of ancillary parameters, the exponential model has none (thus, `ancillary()` is not relevant); The Weibull, Gompertz, log-normal, and log-logistic models each have one ancillary parameter, and the generalized gamma model has two (for which there is a corresponding `anc2()` option to go along with `ancillary()`).

To understand why we might want to parameterize ancillary parameters, let us reconsider the expanded hip-fracture data first discussed in Chapter 9. This dataset, in addition to the data on female patients, contains data on males and includes the additional variable `male` specifying the patient's sex.

Let us begin by fitting a combined male and female Weibull PH model with `protect` and `age` as covariates.

```

. use http://www.stata-press.com/data/cgg/hip3, clear
(hip fracture study)
. streg protect age, dist(weibull) nohr nolog
      failure _d: fracture
      analysis time _t: time1
      id: id

Weibull regression -- log relative-hazard form
No. of subjects =          148          Number of obs   =          206
No. of failures =           37
Time at risk    =          1703
Log likelihood   =   -77.446477          LR chi2(2)      =          49.97
                                          Prob > chi2      =          0.0000

```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
protect	-2.383745	.3489219	-6.83	0.000	-3.067619 -1.699871
age	.0962561	.03467	2.78	0.006	.0283042 .1642079
_cons	-10.63722	2.597762	-4.09	0.000	-15.72874 -5.545696
/ln_p	.4513032	.1265975	3.56	0.000	.2031767 .6994297
p	1.570357	.1988033			1.225289 2.012605
1/p	.6367977	.080617			.4968686 .816134

Recall that the hazard function for the Weibull PH model is

$$h(t_j|\mathbf{x}_j) = \exp(\beta_0 + \mathbf{x}_j\beta_x) p t_j^{p-1}$$

which for our model translates into the estimated hazard

$$\hat{h}(t_j|\mathbf{x}_j) = \exp(-10.64 - 2.38\text{protect}_j + 0.10\text{age}_j) 1.57 t_j^{0.57}$$

If we wanted to compare males to females, we could add the variable `male` to our covariate list.

(Continued on next page)

```
. streg protect age male, dist(weibull) nohr nolog
      failure _d: fracture
      analysis time _t: time1
      id: id

Weibull regression -- log relative-hazard form
No. of subjects =      148      Number of obs   =      206
No. of failures =       37
Time at risk    =     1703

Log likelihood   =   -71.224402      LR chi2(3)    =     62.42
                                      Prob > chi2    =     0.0000
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
protect	-2.095949	.3632392	-5.77	0.000	-2.807885 -1.384013
age	.0907551	.0339681	2.67	0.008	.0241788 .1573314
male	-1.413811	.4555996	-3.10	0.002	-2.30677 -0.5208524
_cons	-9.667022	2.562691	-3.77	0.000	-14.6898 -4.64424
/ln_p	.3784328	.1302032	2.91	0.004	.1232392 .6336264
p	1.459995	.190096			1.131155 1.884432
1/p	.684934	.0891806			.5306639 .8840521

Since `male` is in our covariate list, we assume that the hazards for males and females are proportional (given `age` and `protect`). The net effect is that only the scale of the distribution of t_j changes according to the patient's sex. Since `male==0` for females and `male==1` for males, the above output gives the estimated hazards

$$\hat{h}(t_j|\mathbf{x}_j) = \begin{cases} \exp(-9.67 - 2.10\text{protect}_j + 0.09\text{age}_j) 1.46t_j^{0.46}, & \text{if female} \\ \exp(-11.08 - 2.10\text{protect}_j + 0.09\text{age}_j) 1.46t_j^{0.46}, & \text{if male} \end{cases}$$

Since the Wald test for `male` is $z = -3.10$ with significance level 0.002, we are confident that gender does have some effect on the risk of hip fracture. We are not, however, convinced that the effect is proportional. It may be that the effect of gender is to alter the shape of the hazard instead and that the above output is just a best approximation as to the effect of gender given the constraints of the proportional hazards assumption.

The Weibull distribution has two parameters, a scale parameter and a shape parameter. The scale parameter is the exponentiated linear predictor and is thus modeled using the covariate list in `streg`. The other parameter is the shape, p , and if we want this parameter to be modeled, we use the `ancillary()` option to `streg`:

```

. streg protect age, dist(weibull) ancillary(male) nohr nolog
      failure _d: fracture
      analysis time _t: time1
      id: id

Weibull regression -- log relative-hazard form
No. of subjects =      148      Number of obs   =      206
No. of failures =       37
Time at risk   =     1703
Log likelihood  = -69.323532      LR chi2(2)     =     39.80
                                      Prob > chi2    =     0.0000

```

	_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
_t	protect	-2.130058	.3567005	-5.97	0.000	-2.829178 -1.430938
	age	.0939131	.0341107	2.75	0.006	.0270573 .1607689
	_cons	-10.17575	2.551821	-3.99	0.000	-15.17722 -5.174269
ln_p	male	-.4887189	.185608	-2.63	0.008	-.8525039 -.1249339
	_cons	.4540139	.1157915	3.92	0.000	.2270667 .6809611

In this model, we are assuming that $\ln(p)$ is not constant over the data, but instead is a linear function of some covariates \mathbf{z} , $\ln(p) = \alpha_0 + \mathbf{z}\boldsymbol{\alpha}_z$, where \mathbf{z} need not be distinct from \mathbf{x} . The reason we parameterize in terms of $\ln(p)$ instead of p itself is that p is constrained to be positive, whereas a linear predictor can take on any values on the real line.

From our estimation results, we see that $\widehat{\ln(p)} = 0.454$ for females and $\widehat{\ln(p)} = 0.454 - 0.489 = -0.035$ for males. Thus, $\widehat{p} = \exp(0.454) = 1.57$ for females and $\widehat{p} = \exp(-0.035) = 0.97$ for males. The estimated hazards are then

$$\widehat{h}(t_j|\mathbf{x}_j) = \begin{cases} \exp(-10.18 - 2.13\text{protect}_j + 0.09\text{age}_j) 1.57t_j^{0.57}, & \text{if female} \\ \exp(-10.18 - 2.13\text{protect}_j + 0.09\text{age}_j) 0.97t_j^{-0.03}, & \text{if male} \end{cases}$$

and if we believe this model, we would say that the hazard for males given `age` and `protect` is almost constant.

The Wald test for `male` in the above output is $z = -2.63$ with level of significance 0.008, and thus we are still convinced of the effect of gender on the risk of hip fracture. However, we are no better off than we were before. In this case, it may be that the effect of gender is actually proportional, and by not allowing a shift in the hazard to take place, the model actually had to change the shape of the hazard in order to accommodate this effect.

When we included `male` in the covariate list, we assumed that the effect of gender was on the scale of the hazard and that the shape of the hazard was the same for both sexes. When we specified `ancillary(male)`, we assumed that the shape of the hazard

changed according to gender but that the scale remained unaffected. We can be fully general and allow both the scale and shape to vary by sex by using

```
. streg protect age if male, dist(weib)
. streg protect age if !male, dist(weib)
```

but if we did that we would be assuming that the effect of `protect` and `age` differed according to gender as well. There is certainly nothing wrong with this, but let us assume that what we want is to allow the scale and shape of the hazard to vary with gender but constrain the effect of `protect` and `age` to remain the same. This is achieved by specifying `male` both in the covariate list and in `ancillary()`:

```
. streg protect age male, dist(weibull) ancillary(male) nohr nolog
      failure _d: fracture
      analysis time _t: time1
      id: id

Weibull regression -- log relative-hazard form
No. of subjects =      148                Number of obs   =      206
No. of failures =       37
Time at risk   =     1703
Log likelihood =  -69.082313              LR chi2(3)      =     40.28
                                          Prob > chi2      =     0.0000
```

	_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
_t	protect	-2.185115	.3645006	-5.99	0.000	-2.899523	-1.470707
	age	.0966629	.0345663	2.80	0.005	.0289142	.1644117
	male	.7382003	1.036608	0.71	0.476	-1.293514	2.769916
	_cons	-10.61465	2.646846	-4.01	0.000	-15.80238	-5.42693
ln_p	male	-.7116757	.3834735	-1.86	0.063	-1.46327	.0399185
	_cons	.5079011	.1358255	3.74	0.000	.2416881	.7741142

Comparing the Wald tests for both instances of `male` in this model favors our second version of this story: the effect of gender is on the shape of the hazard, and given that the shape changes, the effect on the scale is not significant.

Satisfied that we need to include `male` only in the estimation of the ancillary parameter, we can now use `ancillary()` to test the proportional hazards assumption on the other two covariates, `protect` and `age`. For example,

(Continued on next page)

```
. streg protect age, dist(weibull) ancillary(male protect) nohr nolog
      failure _d: fracture
      analysis time _t: time1
      id: id

Weibull regression -- log relative-hazard form
No. of subjects =      148      Number of obs   =      206
No. of failures =       37
Time at risk   =     1703

Log likelihood =  -69.264251      LR chi2(2)    =     13.69
                                   Prob > chi2    =     0.0011
```

	_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
_t						
	protect	-2.456007	1.023378	-2.40	0.016	-4.46179 - .4502238
	age	.0934049	.0340489	2.74	0.006	.0266704 .1601394
	_cons	-10.02396	2.580379	-3.88	0.000	-15.08141 -4.966508
ln_p						
	male	-.4801077	.1850198	-2.59	0.009	-.8427398 -.1174756
	protect	.0763409	.2215628	0.34	0.730	-.3579142 .510596
	_cons	.4220745	.1505132	2.80	0.005	.1270741 .7170749

and thus we see no evidence to contradict the proportional hazards assumption for `protect`. A similar analysis performed on `age` would also fail to reject the proportional hazards assumption. Since the above test of proportional hazards is based on a comparison of nested models, we could also perform a likelihood-ratio test,

```
. streg protect age, dist(weibull) ancillary(male protect) nohr nolog
      (output omitted)
. estimates store protect
. streg protect age, dist(weibull) ancillary(male) nohr nolog
      (output omitted)
. lrtest protect ., force

likelihood-ratio test      LR chi2(1) =      0.12
(Assumption: . nested in protect)      Prob > chi2 =      0.7306
```

with similar results. Note that to perform this test we had to use the `force` option to `lrtest`. This is because we are omitting a parameter from the ancillary parameter list and not the main parameter list. This causes `lrtest` to think the models are not nested since they have differing log likelihoods under the null hypothesis that all main parameters are zero. In any case, the use `force` is justified since we know for a fact that the models are nested.

The use of the `ancillary()` option is not restricted to PH models; the option may be used with AFT models, and the interpretation of results is a straightforward extension of the above.

15.2 Stratified models

Stratified models, you may remember from Section 9.3, concern estimation when the baseline hazard is allowed to differ for different groups. Males and females, for instance, might each have their own hazard function rather than that of males being constrained to be a proportional replica of that for females. In Section 9.3, we allowed this in the context of semiparametric estimation.

In parametric stratified estimation, each group is similarly allowed to have its own baseline hazard function, but the hazard functions are constrained to be of the same family. If the hazard for females is Weibull, then so must be the hazard for males.

There is an obvious connection of this idea with the `ancillary()` option. In the previous section, when we typed

```
. streg protect age male, dist(weibull) ancillary(male) nohr nolog
```

we allowed both the scale and shape of the hazard to vary with `male`, yet we constrained the effects of `protect` and `age` to be the same for both sexes. Examining the output from that estimation, we can construct the estimated hazards for males and females as

$$\hat{h}(t_j|\mathbf{x}_j) = \begin{cases} \exp(-10.61 - 2.19\text{protect}_j + 0.10\text{age}_j) 1.66t_j^{0.66}, & \text{if female} \\ \exp(-9.88 - 2.19\text{protect}_j + 0.10\text{age}_j) 0.82t_j^{-0.18}, & \text{if male} \end{cases}$$

and since this is a PH model, we can write the estimated hazard as

$$\hat{h}(t_j|\mathbf{x}_j) = \hat{h}_0(t_j) \exp(-2.19\text{protect}_j + 0.10\text{age}_j)$$

for the estimated baseline hazard $\hat{h}_0(t_j)$ such that

$$\hat{h}_0(t_j) = \begin{cases} \exp(-10.61) 1.66t_j^{0.66}, & \text{if female} \\ \exp(-9.88) 0.82t_j^{-0.18}, & \text{if male} \end{cases}$$

and we can graph the comparison of these baseline hazards using

```
. gen h0female = exp(-10.61)*1.66*_t^(0.66)
. gen h0male = exp(-9.88)*0.82*_t^(-0.18)
. line h0female h0male _t, sort ltitle("Baseline hazard")
```

This produces Figure 15.1 and graphically verifies what we already know: the shape of the hazard is different for males and females; the hazards are not proportional.

(Continued on next page)

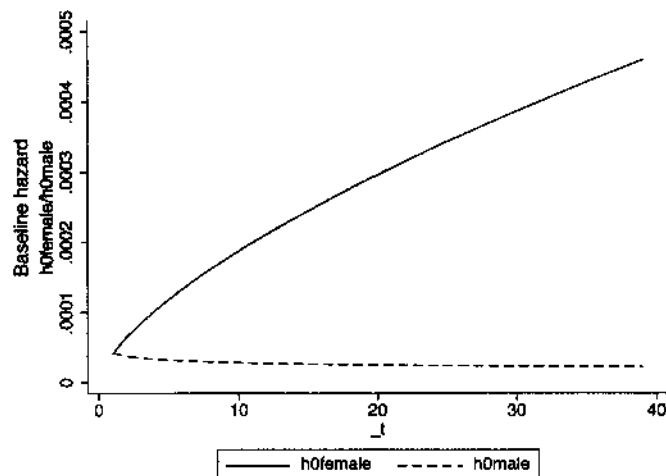


Figure 15.1: Comparison of baseline hazards for males and females

Thus, we have a model that assumes proportional hazards with respect to `protect` and `age`, yet we have different baseline hazard functions for males and females. Said differently, we have a proportional hazards model on `protect` and `age` that is stratified on `male` and is analogous to the stratified Cox model.

`streg`'s `strata(varname)` option provides an easier way to fit these models:

```
. streg protect age, dist(weibull) strata(male) nohr nolog
      failure _d: fracture
      analysis time _t: time1
      id: id
```

Weibull regression -- log relative-hazard form

```
No. of subjects =      148                Number of obs   =      206
No. of failures =       37
Time at risk    =     1703
Log likelihood   =  -69.082313
LR chi2(3)      =      40.28
Prob > chi2     =      0.0000
```

		Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
_t	protect	-2.185115	.3645006	-5.99	0.000	-2.899523	-1.470707
	age	.0966629	.0345663	2.80	0.005	.0289142	.1644117
	_Smale_1	.7382003	1.036608	0.71	0.476	-1.293514	2.769915
	_cons	-10.61465	2.646846	-4.01	0.000	-15.80238	-5.42693
ln_p	_Smale_1	-.7116757	.3834735	-1.86	0.063	-1.46327	.0399185
	_cons	.5079011	.1358255	3.74	0.000	.2416881	.7741142

These are precisely the same estimates we obtained previously, the only difference being that the effect of gender is now labeled as `_Smale_1`. The prefix `_S` is used to denote variables created by `streg`.

When you specify `strata(varname)`, `varname` is to identify a categorical variable that identifies the strata. `streg` takes this categorical variable and from it constructs indicator variables that uniquely identify the strata, and then puts those indicator variables in the model everywhere they need to appear so that the baseline hazards are allowed to differ. In the case of the model fit above, the variable `male` was already an indicator variable, and so this amounted to putting `male` in the two places required, first in the main equation (scale), and second in the ancillary equation (shape).

Suppose that instead of `age`, we had the variable `agecat` that equaled 1 if `age` is less than 65, 2 if `age` is between 65 and 74 inclusive, and 3 if `age` is greater than 74.

```
. gen agecat = 1
. replace agecat = 2 if age >=65
(166 real changes made)
. replace agecat = 3 if age > 74
(52 real changes made)
```

We could then hypothesize a model where the hazard is proportional with respect to `protect` and `male`, but stratified on `agecat`.

```
. streg protect male, dist(weibull) strata(agecat) nolog nohr
      failure _d: fracture
      analysis time _t: time1
      id: id

Weibull regression -- log relative-hazard form
No. of subjects =          148                Number of obs =          206
No. of failures =           37
Time at risk    =          1703

Log likelihood = -72.948131                LR chi2(4) =          57.86
                                          Prob > chi2 =          0.0000
```

	_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
_t	protect	-2.082556	.3676436	-5.66	0.000	-2.803125	-1.361988
	male	-1.501734	.4554929	-3.30	0.001	-2.394483	-.6089838
	_Sagecat_2	.3400986	1.610656	0.21	0.833	-2.816729	3.496927
	_Sagecat_3	1.132034	1.670177	0.68	0.498	-2.141454	4.405521
	_cons	-3.646434	1.453472	-2.51	0.012	-6.495187	-.7976804
ln_p	_Sagecat_2	-.0683576	.370067	-0.18	0.853	-.7936756	.6569604
	_Sagecat_3	-.0875935	.3915644	-0.22	0.823	-.8550456	.6798586
	_cons	.4174532	.3334877	1.25	0.211	-.2361707	1.071077

We see that for this model, the scale and shape of the hazard are allowed to vary with respect to `agecat`. It is evident from the labeling of the output that `agecat==1` is

used as the base category. That is, `agecat==1` when both indicators `_Sagecat_2` and `_Sagecat_3` are equal to zero. For this model, the estimated baseline hazards are

$$\hat{h}_0(t_j) = \begin{cases} \exp(-3.65) 1.52t_j^{0.52}, & \text{if } \text{agecat}==1 \\ \exp(-3.31) 1.42t_j^{0.42}, & \text{if } \text{agecat}==2 \\ \exp(-2.51) 1.39t_j^{0.39}, & \text{if } \text{agecat}==3 \end{cases}$$

and given the Wald tests displayed, none of these baseline hazards are significantly different from any other.

We demonstrated the `strata()` option using a PH model, but there is nothing stopping us from applying it to a AFT model. For an AFT model, when we stratify, we not only allow time to accelerate or decelerate with respect to the strata, but we also allow the actual shape of the baseline survival function to vary with the strata as well.

Note that the `strata()` option is really just a convenience option that attaches the term “stratification” to something that we could have done manually without this option. With the exception of variable labels in the output,

```
. streg ..., strata(varname) ...
```

is synonymous with

```
. xi: streg ... i.varname, ancillary(i.varname) ...
```

for models with one ancillary parameter, with

```
. xi: streg ... i.varname, ancillary(i.varname) anc2(i.varname) ...
```

for the generalized gamma model, and with

```
. xi: streg ... i.varname, ...
```

for the exponential. See [R] xi for more information about the creation of indicator variables.

15.3 Frailty models

Frailty models are a further generalization of the parametric regression models available to users of `streg`. A *frailty* model can take one of two forms, an overdispersion/heterogeneity model or a random-effects model. The latter are referred to as *shared frailty* models in order to make the distinction between the two, and here we define the term *unshared frailty* model to mean the former. What follows is a brief discussion of these models; for a more detailed treatment (and a more complete bibliography), see Gutierrez (2002).

15.3.1 Unshared frailty models

Each parametric regression model available in `streg` can be characterized by the hazard function $h(t_j|\mathbf{x}_j)$, regardless of whether the model is parametrized in the PH or the AFT metric. In the case of PH models, the hazard function is conveniently represented as a baseline hazard that is multiplicatively affected by the covariates, but such a representation is not necessary in the development that follows.

A frailty model in the unshared case defines the hazard to be

$$h(t_j|\mathbf{x}_j, \alpha_j) = \alpha_j h(t_j|\mathbf{x}_j) \quad (15.1)$$

where α_j is some unobserved observation-specific effect. The effect, α_j , is known as a frailty and serves to represent that individuals in the population are heterogeneous due to factors that remain unobserved. The frailties are positive quantities not estimated from the data but instead assumed to have mean one (for purposes of identifiability) and variance θ , and θ is estimated from the data. If $\alpha_j < 1$, then the effect is to decrease the hazard, and thus such subjects are known to be less frail than their counterparts. If $\alpha_j > 1$, then these more frail subjects face an increased risk.

The term *frailty* was first suggested by Vaupel et al. (1979) in the context of mortality studies, and by Lancaster (1979) in the context of duration of unemployment. Much of the initial work on frailty models is due to Hougaard (1984; 1986a; 1986b).

Given the relationship between the hazard and survivor functions, it can be shown from (15.1) that

$$S(t_j|\mathbf{x}_j, \alpha_j) = \{S(t_j|\mathbf{x}_j)\}^{\alpha_j}$$

where $S(t_j|\mathbf{x}_j)$ is the survival function for a standard parametric model, such as the ones described in Chapter 13. The unconditional survival function is obtained by integrating out the unobservable α_j , and for this we need to assume a distribution for α_j . Two popular choices (popular due to many reasons, one of which is mathematical tractability) are the gamma distribution and the inverse-Gaussian distribution, both available in Stata.

If α_j has probability density function $g(\alpha_j)$, then the unconditional survival function is obtained by integrating out the frailty,

$$S_\theta(t_j|\mathbf{x}_j) = \int_0^\infty \{S(t_j|\mathbf{x}_j)\}^{\alpha_j} g(\alpha_j) d\alpha_j \quad (15.2)$$

where we use the subscript θ to emphasize the dependence on the frailty variance θ .

When α_j follows a gamma distribution with mean one and variance θ ,

$$g(\alpha_j) = \frac{\alpha_j^{1/\theta-1} \exp(-\alpha_j/\theta)}{\Gamma(1/\theta)\theta^{1/\theta}}$$

and (15.2) becomes

$$S_\theta(t_j|\mathbf{x}_j) = [1 - \theta \ln \{S(t_j|\mathbf{x}_j)\}]^{-1/\theta}$$

When α_j follows an inverse-Gaussian distribution with mean one and variance θ ,

$$g(\alpha_j) = \left(\frac{1}{2\pi\theta\alpha_j^3} \right)^{1/2} \exp \left\{ -\frac{1}{2\theta} \left(\alpha_j - 2 + \frac{1}{\alpha_j} \right) \right\}$$

and (15.2) becomes

$$S_\theta(t_j|\mathbf{x}_j) = \exp \left\{ \frac{1}{\theta} \left(1 - [1 - 2\theta \ln\{S(t_j|\mathbf{x}_j)\}]^{1/2} \right) \right\}$$

As such, when modeling individual heterogeneity, a frailty model is just the standard parametric model with the addition of one new parameter θ , and a new definition of the survivor function from which all the likelihood calculations derive in the standard way. For example, for the Weibull model in the PH metric,

$$S(t_j|\mathbf{x}_j) = \exp\{-\exp(\beta_0 + \mathbf{x}_j\beta_x)t_j^p\}$$

For the Weibull-PH model with gamma-distributed heterogeneity,

$$S_\theta(t_j|\mathbf{x}_j) = \{1 + \theta \exp(\beta_0 + \mathbf{x}_j\beta_x)t_j^p\}^{-1/\theta}$$

and notice that in this case and in general, $S_\theta(t_j|\mathbf{x}_j)$ reduces to $S(t_j|\mathbf{x}_j)$ as θ goes to zero.

15.3.2 Kidney data

In order to illustrate frailty models, we will be analyzing a dataset from a study of 38 kidney dialysis patients. Originally considered in McGilchrist and Aisbett (1991), this study is concerned with the prevalence of infection at the catheter insertion point. Two recurrence times are measured for each patient. A catheter is inserted, and the first time to infection (in days) is measured. If the catheter is removed for reasons other than infection, then the first recurrence time is censored. Should infection occur, the catheter is removed, the infection cleared, and then after some predetermined time period the catheter is reinserted. The second time to infection is measured as time elapsed between the second insertion and the second infection or censoring. The second recurrence time is censored if either (a) the catheter is removed for reasons other than infection or (b) the follow-up period for the patient ends before infection occurs.

```
. use http://www.stata-press.com/data/cgg/kidney, clear
(Kidney data, McGilchrist and Aisbett, Biometrics, 1991)
. describe
Contains data from http://www.stata-press.com/data/cgg/kidney.dta
   obs:              38                      Kidney data, McGilchrist and
                                         Aisbett, Biometrics, 1991
   vars:              7                      21 Feb 2002 15:37
   size:              722 (99.9% of memory free)
```

variable name	storage type	display format	value label	variable label
patient	float	%7.0g		Patient ID
time1	int	%9.0g		recurrence time to first infection
fail1	byte	%4.0g		equals 1 if infection, 0 if censored
time2	int	%9.0g		recurrence time to second infection
fail2	byte	%4.0g		equals 1 if infection, 0 if censored
age	float	%6.0g		Patient age
gender	byte	%6.0g		Patient gender (0=male, 1=female)

Sorted by:

```
. list in 21/30
```

	patient	time1	fail1	time2	fail2	age	gender
21.	21	562	1	152	1	46.5	0
22.	22	24	0	402	1	30	1
23.	23	66	1	13	1	62.5	1
24.	24	39	1	46	0	42.5	1
25.	25	40	1	12	1	43	0
26.	26	113	0	201	1	57.5	1
27.	27	132	1	156	1	10	1
28.	28	34	1	30	1	52	1
29.	29	2	1	25	1	53	0
30.	30	26	1	130	1	54	1

The variables `time1` and `time2` record the two times to infection or censoring, and the variables `fail1` and `fail2` are the associated failure indicator variables (1 = failure, 0 = censored). Other covariates that were measured for each patient were patient age, age, and gender (0 = male, 1 = female).

Before we begin analyzing these data, we note that the data are in wide form and thus must first be reshaped into long form before we `stset` the dataset.

```
. reshape long time fail, i(patient) j(pat_insert)
(note: j = 1 2)
```

```
Data                wide  ->  long
Number of obs.      38  ->   76
Number of variables   7  ->    6
j variable (2 values) -> pat_insert
xij variables:
               time1 time2 -> time
               fail1 fail2 -> fail
```

```
. list in 1/14, sepby(patient)
```

	patient	pat_in-t	time	fail	age	gender
1.	1	1	16	1	28	0
2.	1	2	8	1	28	0
3.	2	1	13	0	48	1
4.	2	2	23	1	48	1
5.	3	1	22	1	32	0
6.	3	2	28	1	32	0
7.	4	1	318	1	31.5	1
8.	4	2	447	1	31.5	1
9.	5	1	30	1	10	0
10.	5	2	12	1	10	0
11.	6	1	24	1	16.5	1
12.	6	2	245	1	16.5	1
13.	7	1	9	1	51	0
14.	7	2	7	1	51	0

```
. summarize, sep(0)
```

Variable	Obs	Mean	Std. Dev.	Min	Max
patient	76	19.5	11.03872	1	38
pat_insert	76	1.5	.5033223	1	2
time	76	97.68421	128.3424	2	562
fail	76	.7631579	.4279695	0	1
age	76	43.69737	14.73795	10	69
gender	76	.7368421	.4432733	0	1

We note that our data now consist of 76 observations, each observation chronicles a single catheter insertion, and there are two observations per patient.

Before we `stset` the dataset, we need to consider carefully how we want to define analysis time. Looking at the first patient, we find that the recurrence times are 16 and 8. Given what we know about these data, we realize that these times represent two separate observations that took place on different time lines, and not one observation of a two-failure process for which failure occurred at time $t = 16$ and then 8 days later at time $t = 24$ (for instance). We know this from our description of how the data

were collected: The time to the first infection is measured, and then the infection is cleared and some additional time is allowed to pass (ten weeks), which essentially resets the “risk clock” back to zero. The time at which the second catheter insertion takes place marks the onset of risk for the second infection, and the second recurrence time is measured as the time that has elapsed from this second onset of risk.

Thus, analysis time is just recurrence time as measured, and since the time at which the catheter is inserted marks the onset of risk, `_t0` will equal zero for each observation. It is important to understand that the way we have chosen to measure analysis time is an assumption we are making. We assume that five days after the first catheter insertion is, in terms of accumulated risk, indistinguishable from five days after the second insertion. It is not statistical considerations that force us to make that particular assumption. Statistically, we must make some assumption in order to measure analysis time, but we are free to make any assumption we please. We make this particular assumption because of the substantive arguments given above, because this seems to us the appropriate way to measure analysis time.

Our assumption does not imply that we must treat each insertion as an *independent* observation, and in fact we would not be at all surprised if the two recurrence times for each patient were correlated. What we are saying is that the two recurrence times are distinct observations on the same failure process, and whether these times are correlated is an issue left to be resolved by our analysis, and not one we need to concern ourselves with when `stsetting` the data.

Therefore, for the purposes of `stset` we define an “observation” (or synonymously a “subject”) to be a single catheter insertion and not the aggregated data for a single patient. Using this definition of a subject, we have single-record-pcr-subject data, and thus we are not required to `stset` an id variable that uniquely identifies each subject. However, it is always good practice to define one anyway, since we may want to `stsplit` our records later, and `stsplit` does require that an id variable be `stset`.

```
. gen insert_id = _n
. stset time, failure(fail) id(insert_id)
  (output omitted)
. format _t0 %4.0g
. format _t %5.0g
. format _d %4.0g
. format gender %6.0g
```

(Continued on next page)

```
. list insert_id patient pat_insert _t0 _t _d gender age in 1/10, sepby(patient)
```

	insert-d	patient	pat_in-t	_t0	_t	_d	gender	age
1.	1	1	1	0	16	1	0	28
2.	2	1	2	0	8	1	0	28
3.	3	2	1	0	13	0	1	48
4.	4	2	2	0	23	1	1	48
5.	5	3	1	0	22	1	0	32
6.	6	3	2	0	28	1	0	32
7.	7	4	1	0	318	1	1	31.5
8.	8	4	2	0	447	1	1	31.5
9.	9	5	1	0	30	1	0	10
10.	10	5	2	0	12	1	0	10

Satisfied at how our data are `stset`, we save the data as `kidney2.dta`.

□ Technical Note

Determining what is an “observation” can be confusing. Say that you agree with us that analysis time ought to be measured as time between insertion and infection. That definition, we have just argued, will lead us to `stset` the data by typing

```
. stset ..., id(insert_id) ...
```

but you may wonder why we could not type

```
. stset ..., id(patient) exit(time .)
```

where we would make an “observation” a patient and include `exit(time .)` to allow for multiple failures (two) per patient.

Here is a test that will help you dismiss inconsistent ways of defining “observations”: Determine the maximum number of potential failures in the dataset at the earliest possible instance at which failure could occur. In our dataset we have 38 patients, but in theory, after a short time, there could have been up to 76 failures because each catheter insertion for each patient could have developed an infection immediately.

The maximum number of potential failures must equal the number of “observations”, and so you are lead to a contradiction.

An “observation”, given our definition of analysis time, is a catheter insertion.

For cases where the observations do represent multiple failures per subject, there exist a myriad of methods for analysis; however, these methods are beyond the scope of this introductory text; see Cleves (1999) for a survey of the methodology in this area.

□

15.3.3 Testing for heterogeneity

Continuing our study of the kidney data, we avoid (only for now) the issue of correlation between observations within patient by temporarily confining our analysis only to consider the first catheter insertion for each patient (`pat_insert==1`). We begin by fitting the most general of our parametric models, the generalized gamma:

```
. use http://www.stata-press.com/data/cgg/kidney2, clear
(Kidney data, McGilchrist and Aisbett, Biometrics, 1991)
. keep if pat_insert==1
(38 observations deleted)
. streg age gender, dist(gamma) nolog
      failure _d: fail
      analysis time _t: time
      id: insert_id

Gamma regression -- accelerated failure-time form
No. of subjects =          38          Number of obs =          38
No. of failures =          27
Time at risk   =         3630

Log likelihood =    -48.3764          LR chi2(2) =          4.16
                                Prob > chi2 =          0.1247
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
age	-.0094451	.0148001	-0.64	0.523	-.0384627	.0195625
gender	1.219731	.5547008	2.20	0.028	.1325372	2.306924
_cons	3.640613	.8041598	4.53	0.000	2.064488	5.216737
/ln_sig	.2113222	.1347525	1.57	0.117	-.0527877	.4754322
/kappa	-.2071561	.5673513	-0.37	0.715	-1.319144	.9048321
sigma	1.23531	.1664611			.9485813	1.608709

The first thing we notice is that the Wald test of $H_0: \kappa = 0$ has level of significance 0.715, and thus there is insufficient evidence to reject H_0 , which in essence says that a log-normal model would serve our purposes equally well. In fact, out of `streg`'s six parametric models, the log-normal gives the highest Akaike information criterion (AIC) index (see Section 13.7.2 for a description of AIC).

(Continued on next page)

```
. streg age gender, dist(lnormal) nolog
      failure _d: fail
      analysis time _t: time
      id: insert_id

Log-normal regression -- accelerated failure-time form
No. of subjects =          38          Number of obs   =          38
No. of failures =          27
Time at risk   =          3630
Log likelihood = -48.455243          LR chi2(2)       =          4.81
                                          Prob > chi2    =          0.0902
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
age	-.0079733	.0143949	-0.55	0.580	-.0361867 .0202402
gender	1.113414	.4948153	2.25	0.024	.1435937 2.083234
_cons	3.779857	.7215091	5.24	0.000	2.365725 5.193989
/ln_sig	.2090131	.1336573	1.56	0.118	-.0529503 .4709766
sigma	1.232461	.1647274			.9484272 1.601557

With these regression results, we can use `stcurve` to graph the comparative estimated hazards for males and females with age held at its mean value:

```
. stcurve, hazard at1(gender=0) at2(gender=1)
```

The result is shown in Figure 15.2.

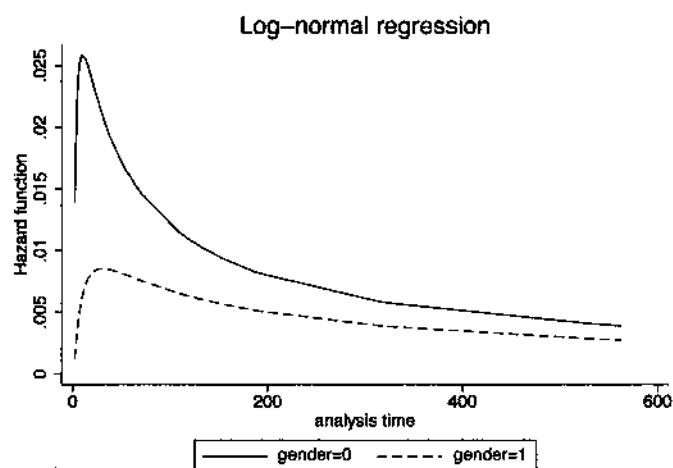


Figure 15.2: Comparison of log-normal hazards for males and females

The hazard for males is much greater than that for females. The hazard, in any case, is estimated from the log-normal model to rise and then fall, and if we choose to adhere to the AIC criterion, this is as good an estimate of the hazard as we can get out of using `streg` in the standard way.

If we interpreted this hazard as the hazard each individual faces, then this says that as time passes the instantaneous risk of infection falls. If this were indeed the individual hazard, then physicians should tell their patients that if infection hasn't taken place by a certain time after the catheter has been inserted, then it is unlikely to happen at all, and that they need not worry. On the other hand, we obtained these estimates by assuming that all patients are identical other than in age and gender. If subjects differ in unobserved ways in their inherent probability of failure—if some are more robust and others more frail—then we must be more careful about our interpretation of results and about how we estimate them.

In frailty models, there is a distinction between the hazard individuals face and the population hazard that arises by averaging over all the survivors. In a heterogeneous population, it turns out that the population hazard can fall while the individual hazards all rise because, over time, the population becomes populated by more and more robust individuals as the more frail members fail. This is known as the frailty effect, and it virtually assures that population hazards decline over time, regardless of the shape of the hazards that individuals face. We will provide evidence that this is indeed occurring in these data.

The implication of this is that, under the assumption of a heterogeneous population, it could actually be the case that each individual's risk rises the longer the catheter remains inserted, and so the clinical recommendation is to remove the catheter as soon as possible, even though for the population as a whole, the infection rate falls.

Under the frailty model, the individual hazard function is written $h(t_j|\mathbf{x}_j, \alpha_j) = \alpha_j h(t_j|\mathbf{x}_j)$, and the resulting population hazard is written as $h_\theta(t_j|\mathbf{x}_j)$, which is obtained from (15.2) in the standard way,

$$h_\theta(t_j|\mathbf{x}_j) = - \left\{ \frac{d}{dt} S_\theta(t_j|\mathbf{x}_j) \right\} \{S_\theta(t_j|\mathbf{x}_j)\}^{-1}$$

It can be shown that as θ tends to zero, the population and individual hazard functions coincide and that $\lim_{\theta \rightarrow 0} h_\theta() = h()$.

Just as when fitting a model without frailty, you assume a parametric form for the hazard function, but in models with frailty, you make that assumption about the hazard function individuals face. The population hazard function is just whatever it turns out to be given the estimate of θ and the assumed distribution of α_j , which in Stata may be gamma or inverse-Gaussian.

We can fit a model with log-normal individual hazard and gamma-distributed frailty by specifying the additional option `frailty(gamma)` to `streg`,

```
. streg age gender, dist(lnormal) frailty(gamma) nolog
      failure _d: fail
      analysis time _t: time
      id: insert_id

Log-normal regression -- accelerated failure-time form
                        Gamma frailty
No. of subjects =      38                Number of obs   =      38
No. of failures =      27
Time at risk   =     3630
Log likelihood  = -48.351289              LR chi2(2)      =      5.02
                                      Prob > chi2       =      0.0813
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
age	-.0097803	.0144447	-0.68	0.498	-.0380913	.0185307
gender	1.260102	.5566201	2.26	0.024	.1691466	2.351057
_cons	3.653898	.7309858	5.00	0.000	2.221192	5.086604
/ln_sig	.1257008	.2199912	0.57	0.568	-.305474	.5568757
/ln_the	-1.965189	2.298354	-0.86	0.393	-6.46988	2.539502
sigma	1.133943	.2494575			.736774	1.745211
theta	.1401294	.3220669			.0015494	12.67336

Likelihood-ratio test of theta=0: chibar2(01) = 0.21 Prob>=chibar2 = 0.324

or if you prefer, a model with inverse-Gaussian distributed frailties:

```
. streg age gender, dist(lnormal) frailty(invgauss) nolog
      failure _d: fail
      analysis time _t: time
      id: insert_id

Log-normal regression -- accelerated failure-time form
                        Inverse-Gaussian frailty
No. of subjects =      38                Number of obs   =      38
No. of failures =      27
Time at risk   =     3630
Log likelihood  = -48.347311              LR chi2(2)      =      5.03
                                      Prob > chi2       =      0.0810
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
age	-.009801	.0144568	-0.68	0.498	-.0381357	.0185337
gender	1.255811	.5422437	2.32	0.021	.1930326	2.318589
_cons	3.646516	.7389398	4.93	0.000	2.19822	5.094811
/ln_sig	.1186605	.2374315	0.50	0.617	-.3466967	.5840176
/ln_the	-1.727569	2.700183	-0.64	0.522	-7.01983	3.564692
sigma	1.125988	.2673449			.7070198	1.793228
theta	.1777159	.4798655			.000894	35.32857

Likelihood-ratio test of theta=0: chibar2(01) = 0.22 Prob>=chibar2 = 0.321

The choice of frailty distribution has implications in the interpretation of how the relative hazard (with respect to the covariates) changes with time; see the technical note

below. Regardless of the choice of frailty distribution, however, from examining the likelihood-ratio test of $H_0: \theta = 0$ at the bottom of each output, we realize that if we are willing to accept that the individual hazards are indeed log-normal, then there is not much evidence pointing towards a population that is heterogeneous.

□ Technical Note

In hazard-metric frailty models, exponentiated coefficients have the interpretation of hazard ratios at $t = 0$ only. After that, the effects of covariate differences become muted as the more frail experience failure and so are removed from the surviving population. In gamma frailty models, the effect of covariate differences eventually diminishes completely in favor of the frailty effect. In inverse-Gaussian frailty models, the effect of covariate differences never vanishes, going instead to the square root of the hazard ratio effect at $t = 0$. For more information, see Gutierrez (2002).

□

Suppose, however, that the science underlying infection due to catheter insertions dictated that individual hazards must increase with time. Our seemingly good results with the log-normal model would then be an artifact caused by the heterogeneity in the data. We could examine this possibility by fitting a Weibull individual hazard model with gamma frailty, and then ask (1) is the Weibull parameter $p > 1$ (which is to say, do individual hazard functions monotonically rise), and (2) does this alternate model fit the data about as well or better?

Let us begin by fitting the Weibull model with gamma frailty:

```
. streg age gender, dist(weibull) frailty(gamma) nolog time
      failure _d: fail
      analysis time _t: time
      id: insert_id
Weibull regression -- accelerated failure-time form
                    Gamma frailty
No. of subjects =      38                Number of obs   =      38
No. of failures =      27
Time at risk   =     3630
Log likelihood = -48.410832                LR chi2(2)      =      4.78
                                      Prob > chi2       =     0.0917
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
age	-.0109489	.013751	-0.80	0.426	-.0379004	.0160026
gender	1.338382	.5071762	2.64	0.008	.3443351	2.332429
_cons	3.570254	.6510318	5.48	0.000	2.294255	4.846253
/ln_p	.4965479	.3129399	1.59	0.113	-.1168032	1.109899
/ln_the	.3610338	.7067754	0.51	0.609	-1.02422	1.746288
p	1.643039	.5141727			.8897603	3.034052
1/p	.6086281	.190464			.3295923	1.123898
theta	1.434812	1.01409			.3590763	5.733282

```
Likelihood-ratio test of theta=0: chibar2(01) =      4.20 Prob>=chibar2 = 0.020
```

We chose the AFT parameterization of the Weibull for no reason other than to ease the comparison with the log-normal, which is only available in the AFT metric.

From these results, we see that our alternative theory has some merit:

- (1) The point estimate of the Weibull parameter p is greater than one (although the 95% confidence interval includes values below 1).
- (2) The fit of this model, as reflected in log-likelihood values, is about the same as that of the log-normal model (being about -48.4 in both cases).

We will not feel completely comfortable until we see graphs of the population and individual hazards, and we will be discussing how to obtain those graphs. In the meantime, you can look ahead to Figure 15.3 and 15.4. Individual hazards rise monotonically, and yet the model reproduces the rise-and-then-fall shape of the population hazard previously exhibited by the log-normal model. (Later, we will also explore what might be the true shape of the individual hazard.)

Individual hazards rise, and yet, population hazards ultimately fall because of heterogeneity, and so we are not surprised to see that in the reported results above, the likelihood-ratio test for $H_0: \theta = 0$ would be rejected, in this case at the 0.02 level.

Let us now return to the issue of obtaining graphs of the hazard functions. After fitting standard (nonfrailty) parametric models, you can type `predict ..., hazard` to obtain the estimated hazard function. After fitting a parametric model with frailty, you type

```
. predict ..., hazard unconditional
```

or

```
. predict ..., hazard alpha1
```

to obtain the estimated hazard functions. With the `unconditional` option, you obtain the population hazard function $h_\theta(t_j|\mathbf{x}_j)$, and with the `alpha1` option, you obtain the mean individual hazard $h(t_j|\mathbf{x}_j)$. The options `unconditional` and `alpha1` can also be used when using `predict` to obtain other predictions after fitting parametric frailty models.

□ Technical Note

For univariate (nonshared) frailty models, the default prediction (that is if neither `unconditional` nor `alpha1` is specified) is the unconditional (population) function. For shared frailty models, which we discuss later, the default prediction is the individual ($\alpha_j = 1$) function. The reason for the change in default behavior has to do with the interpretation of the results. In the shared frailty case, careful consideration must be given to the interpretation of the unconditional functions. See Gutierrez (2002) for further details. In any case, we recommend always specifying either `unconditional` or `alpha1` when predicting the hazard or survivor function after fitting a frailty model.

□

The `alpha1` and `unconditional` options may also be specified in `stcurve` so that we can easily graph either implementation (conditional or unconditional) of the hazard, cumulative hazard, and survivor functions. From our Weibull/gamma model, we can graph the comparative population hazards for males and females,

```
. stcurve, hazard unconditional at1(gender=0) at2(gender=1)
```

which produces Figure 15.3, and the similarity to Figure 15.2 is striking.

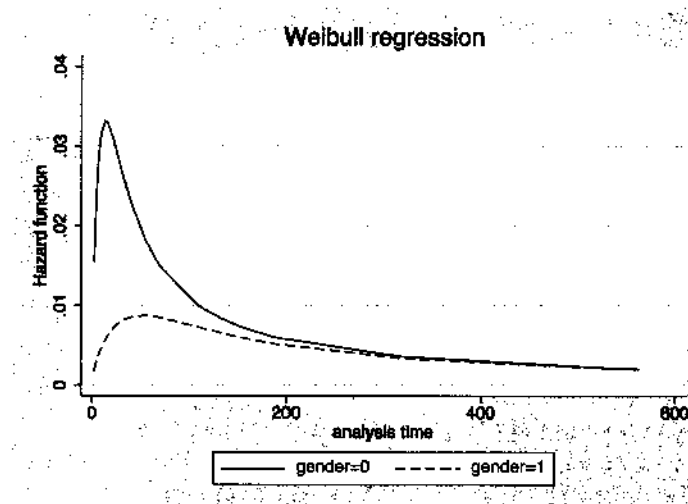


Figure 15.3: Comparison of Weibull/gamma population hazards

Regardless of which story we believe, whether the log-normal individual hazard with no frailty (Figure 15.2) or the Weibull individual hazard with significant frailty (Figure 15.3), the estimate of the population hazard is virtually the same.

We can also graph the individual hazards for the Weibull/gamma model,

```
. stcurve, hazard alpha1 at1(gender=0) at2(gender=1)
```

which produces Figure 15.4.

(Continued on next page)

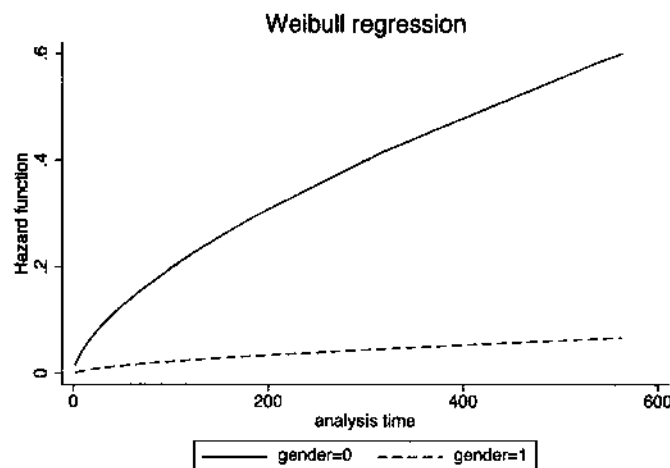


Figure 15.4: Comparison of Weibull/gamma individual ($\alpha_j = 1$) hazards

As expected, the individual hazards are monotone increasing. Thus, if we believe this model—that is, if we believe that the science dictates a monotone increasing hazard—then we must also believe that there is some patient-level effect that we are not measuring. A physician looking at this model would realize that some individuals are just more susceptible to infection than others for reasons unmeasured, and when consulting with a patient would actually warn that the risk of infection increases (for that patient) the longer the catheter remains inserted.

The question then becomes, which model better fits the data? Do individual hazards monotonically rise, and is the fact that we see hazards falling in the data merely an artifact of the heterogeneity, or do individual hazards rise and then fall in lock step with the population hazard? Clearly, patients are heterogeneous—we would never argue that age and sex fully describe the differences. In the two extremes we have posed, we are really asking whether that heterogeneity is so great that it dominates the production of population hazards from individual hazards, or if, instead, for this particular problem, that heterogeneity can be ignored.

Which of our two alternate theories is better supported by the data? Recall that throughout this section we have confined the analysis to the first infection time for each patient, and thus we wish to consider the full dataset (both infection times) to answer this question.

15.3.4 Shared frailty models

In Section 15.3.3, we confined our analysis to the first time to infection for each patient because we wished to investigate the existence of a latent patient frailty in the univariate case. We now consider both infection times for each patient:

```
. use http://www.stata-press.com/data/cgg/kidney2, clear
(Kidney data, McGilchrist and Aisbett, Biometrics, 1991)
. list insert_id patient pat_insert _t0 _t _d gender age in 1/6, separator(0)
```

	insert-d	patient	pat_in-t	_t0	_t	_d	gender	age
1.	1	1	1	0	16	1	0	28
2.	2	1	2	0	8	1	0	28
3.	3	2	1	0	13	0	1	48
4.	4	2	2	0	23	1	1	48
5.	5	3	1	0	22	1	0	32
6.	6	3	2	0	28	1	0	32

Realizing that the two recurrence times for each patient are probably correlated, we could take one of our standard parametric survival models and adjust the standard errors to take into account the intra-patient correlation.

```
. streg age gender, dist(weibull) cluster(patient) time nohr nolog
      failure _d: fail
      analysis time _t: time
      id: insert_id
Weibull regression -- accelerated failure-time form
No. of subjects      =           76      Number of obs      =           76
No. of failures      =           58
Time at risk         =           7424
Log pseudo-likelihood = -103.44362      Wald chi2(2)      =           3.38
                                          Prob > chi2       =           0.1848
                                          (standard errors adjusted for clustering on patient)
```

_t	Coef.	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
age	-.004559	.0097206	-0.47	0.639	-.0236111	.0144932
gender	.9194971	.5413907	1.70	0.089	-.1416092	1.980603
_cons	4.30243	.7075207	6.08	0.000	2.915715	5.689145
/ln_p	-.1028083	.0798087	-1.29	0.198	-.2592305	.053614
p	.9023	.0720114			.7716451	1.055077
1/p	1.108279	.0884503			.9477979	1.295933

By specifying `cluster(patient)`, we are saying that we do not believe these to be 76 independent observations but rather 38 independent “clusters” of observations. When you specify `cluster()`, you get robust standard errors, those obtained via the Huber/White/sandwich estimator of variance. Should there exist intra-cluster correlation, the robust standard errors are better indicators of the sample-to-sample variability of the parameter estimates and thus produce more accurate tests of the effects of covariates. For more details on the sandwich estimator of variance, see [P] `_robust`. We do not want to make much of such an analysis here, since for most situations we can imagine where observations are correlated within groups, they are usually correlated due to some overall group characteristic (a frailty) that is not being measured. What we instead want to do is to model the correlation using a shared frailty model.

Shared frailty models are the survival-data analog to random-effects models. A shared frailty is a frailty model where the frailties are no longer observation specific, but instead are shared across groups of observations, thus causing those observations within the same group to be correlated. In our example, the two infection times for each patient share the same frailty because we assume the frailty to be a characteristic of the patient, not of the catheter insertion.

The generalization of (15.1) to a case where frailty is assumed to be shared across groups of observations is

$$h(t_{ij}|\mathbf{x}_{ij}, \alpha_i) = \alpha_i h(t_{ij}|\mathbf{x}_{ij})$$

for data consisting of n groups with the i th group comprised of n_i observations. The index i denotes the group ($i = 1, \dots, n$), and j denotes the observation within group, $j = 1, \dots, n_i$.

The frailties, α_i , are shared within each group and are assumed to follow either a gamma or inverse-Gaussian distribution (as before). The frailty variance, θ , is estimated from the data and measures the variability of the frailty across groups.

To fit a shared frailty model, one need only add the option `shared(varname)` to `streg, frailty()`, where `varname` is an id variable describing those groups wherein frailties are shared. Returning to our example, we again compare the log-normal and Weibull models, and since this time we are considering both catheter insertions for each patient, we assume that the frailty is shared at the patient level.

We first fit a log-normal with gamma-shared frailty:

```
. streg age gender, dist(lnormal) frailty(gamma) shared(patient) nolog
      failure _d: fail
      analysis time _t: time
      id: insert_id

Log-normal regression --
      accelerated failure-time form
      Gamma shared frailty
Group variable: patient
No. of subjects =          76
No. of failures =          58
Time at risk   =       7424
Log likelihood =   -97.594575

Number of obs   =          76
Number of groups =          38
Obs per group: min =          2
                  avg =          2
                  max =          2
LR chi2(2)      =       16.65
Prob > chi2     =       0.0002
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
age	-.0067002	.0098278	-0.68	0.495	-.0259623	.0125619
gender	1.422234	.3343544	4.25	0.000	.7669111	2.077556
_cons	3.331102	.4911662	6.78	0.000	2.368434	4.29377
/ln_sig	.0670784	.1187767	0.56	0.572	-.1657197	.2998764
/ln_the	-1.823592	.995803	-1.83	0.067	-3.77533	.1281455
sigma	1.069379	.1270173			.8472837	1.349692
theta	.1614447	.1607671			.0229295	1.136718

Likelihood-ratio test of theta=0: chibar2(01) = 1.57 Prob>=chibar2 = 0.105

From the likelihood-ratio test of $H_0: \theta = 0$, we find the frailty effect to be insignificant at the 10% level. This is not the overwhelming lack of evidence of heterogeneity that we had before in the univariate case, but the effect of frailty is still small in this model. If the individual hazards are indeed log-normal, then there is little evidence of a heterogeneous population of patients.

If we fit a Weibull with gamma shared frailty,

```
. streg age gender, dist(weibull) fr(gamma) shared(patient) time nohr nolog
      failure _d: fail
      analysis time _t: time
      id: insert_id

Weibull regression --
      accelerated failure-time form      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

Log likelihood = -98.006931              LR chi2(2)          =     14.81
                                      Prob > chi2         =     0.0006
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
age	-.0067052	.0102377	-0.65	0.512	-.0267707	.0133602
gender	1.506616	.3659291	4.12	0.000	.7894085	2.223824
_cons	3.557985	.5224117	6.81	0.000	2.534077	4.581894
/ln_p	.2410369	.1336503	1.80	0.071	-.0209129	.5029866
/ln_the	-.4546298	.4747326	-0.96	0.338	-1.385089	.475829
p	1.272568	.1700791			.9793043	1.653653
1/p	.7858127	.1050241			.6047219	1.021133
theta	.6346829	.3013047			.2503016	1.609348

Likelihood-ratio test of theta=0: chibar2(01) = 10.87 Prob>=chibar2 = 0.000

we see a significant frailty effect. Thus, if we believe the individual hazard to be Weibull, then we must also be willing to believe in an unobserved patient-level effect. Also, we estimated $\hat{p} = 1.27$; that is, the estimated individual hazard for this model is monotone increasing.

This is the same situation we were in before in the unshared case, and if we used `stcurve` to plot and compare the hazard functions as we did before, we would see the same phenomena.

To help us settle the issue of which of the two competing models is preferable, we will use our favorite trick of fitting a piecewise exponential model (piecewise constant hazard) with a gamma-shared frailty model. By splitting time into several discrete intervals and fitting a separate hazard on each, we are being more flexible in our specification of the individual hazard, and by being more flexible we are allowing the data, if not to entirely speak for themselves, at least to get a word in as to what the individual hazard and level of heterogeneity are.

In order to fit the piecewise exponential model, we will generate a set of indicator variables that comprise a grid of ten intervals spanning $[0, 60)$, $[60, 120)$, \dots , $[540, \infty)$ (note that `_t==562` is the largest value in our data). Before we define our intervals, however, we need to `stsplit` our data at the appropriate times.

```
. stsplit my_t, at(60(60)562)
(92 observations (episodes) created)
. forvalues k = 1/9 {
  2.     gen in_`k' = ((`k'-1)*60 <= my_t) & (my_t < `k'*60)
  3. }
```

The indicators are constructed so that the baseline interval (that for which all the indicators equal zero) is the interval $[540, \infty)$.

We now fit the piecewise exponential model with gamma-shared frailty by including the indicator variables in the covariate list:

```
. streg age gender in_*, dist(exp) fr(gamma) shared(patient) nolog nohr time
      failure _d: fail
      analysis time _t: time
      id: insert_id

Exponential regression --
      accelerated failure-time form      Number of obs      =      168
      Gamma shared frailty              Number of groups   =      38
Group variable: patient
No. of subjects =      76                Obs per group: min =      2
No. of failures =      58                avg =    4.421053
Time at risk    =    7424                max =      14

LR chi2(11)      =    20.90
Log likelihood   =   -95.241838          Prob > chi2      =    0.0344
```

_t	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
age	-.0060301	.0119445	-0.50	0.614	-.0294409	.0173807
gender	1.416426	.4525747	3.13	0.002	.5293963	2.303456
in_1	2.566041	1.350038	1.90	0.057	-.0799854	5.212067
in_2	3.229574	1.339194	2.41	0.016	.6048016	5.854347
in_3	2.366064	1.29691	1.82	0.068	-.1758328	4.907961
in_4	2.1349	1.305189	1.64	0.102	-.4232243	4.693024
in_5	2.44805	1.373428	1.78	0.075	-.2438189	5.139919
in_6	2.788362	1.52622	1.83	0.068	-.2029743	5.779697
in_7	2.659777	1.518699	1.75	0.080	-.3168189	5.636373
in_8	2.319784	1.501516	1.54	0.122	-.6231325	5.262701
in_9	1.233349	1.296951	0.95	0.342	-1.308628	3.775326
_cons	1.057467	1.488013	0.71	0.477	-1.858986	3.973919
/ln_the	-.8007126	.5527555	-1.45	0.147	-1.884094	.2826683
theta	.4490089	.2481921			.1519667	1.326665

```
Likelihood-ratio test of theta=0: chibar2(01) =    6.62 Prob>=chibar2 = 0.005
```

The first thing we notice is the likelihood-ratio test of $H_0: \theta = 0$. By not specifying a strict parametric form for the individual hazard, but instead leaving the model to its own devices with a more flexible specification, we still find evidence of a heterogeneous

population of patients. In terms of our previous debate over the form of the individual hazard, this leads us to initially favor the Weibull model over the log-normal model, since for the Weibull we also have significant heterogeneity.

Note also the similarity in estimation results between the piecewise exponential shared frailty model and the Cox shared frailty model (which makes no assumption about the functional form of the hazard) as fitted to the same data in Section 9.4. In the Cox shared frailty model, $\hat{\theta} = 0.475$ versus $\hat{\theta} = 0.449$ for the piecewise exponential. The magnitudes of the coefficients on `age` and `gender` are also similar across both models, yet the signs are reversed. The sign reversal is due to the fact that we fit the piecewise exponential model in the AFT metric, whereas the Cox model is inherently a PH model.

We can use `predict` followed by `line` to examine the estimated individual hazards for males and females from the piecewise exponential model. Note that `stcurve` won't work here because we have indicator variables that vary over time, and `stcurve` likes to hold things constant. Because we are generating our own graph, and because of the sparseness of the data in the right tail of the distribution, we generate our own plotting grid rather than use the existing `_t` in our data with

```
. drop _all
. set obs 400
obs was 0, now 400
. gen time = _n
. stset time
      failure event: (assumed to fail at time=time)
obs. time interval: (0, time]
exit on or before: failure
```

```
      400  total obs.
       0  exclusions
```

```
      400  obs. remaining, representing
      400  failures in single record/single failure data
    80200  total analysis time at risk, at risk from t =          0
              earliest observed entry t =          0
              last observed exit t =          400
```

```
. forvalues k = 1/9 {
2.      gen in_`k' = (((`k'-1)*60 <= _t) & (_t < `k'*60)
3. }

. gen age = 43.7      /* mean age */
. gen gender = 0
. predict h_male, hazard alpha1
. label var h_male "males"
. replace gender = 1
(400 real changes made)
. predict h_female, hazard alpha1
. label var h_female "females"
. line h_female h_male _t, sort ltitle("hazard")
```

which produces Figure 15.5.

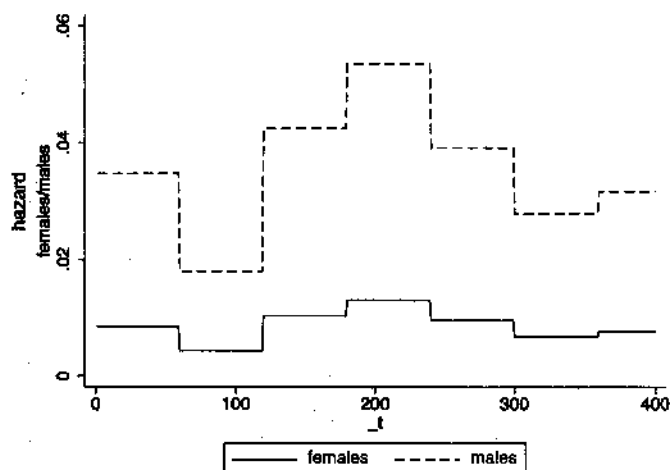


Figure 15.5: Comparison of piecewise-constant individual ($\alpha = 1$) hazards

Even though the piecewise exponential model features a similar level of heterogeneity than the Weibull, the hazard is by no means monotone increasing, and in fact begins to decrease steadily at about time $t = 200$. Admittedly, the data are pretty sparse past this point, so we should not make too much of the estimated hazards at the later time points. In any case, our model-agnostic, piecewise-constant hazard offers little in the way of a scientific interpretation, but fitting this model at least served to further confirm that, whatever the form of the individual hazard, there is evidence of patient heterogeneity.

The answer to the first part of our question is that heterogeneity cannot simply be ignored for this problem. In terms of the second part of our question—whether individual hazards are rising or falling—we have inconclusive evidence. In the absence of a strong theory to answer that question, we would need more data.

References

- Aalen, O. O. 1978. Nonparametric inference for a family of counting processes. *Annals of Statistics* 6: 701–726.
- Akaike, H. 1974. A new look at the statistical model identification. *IEEE Transaction and Automatic Control* AC-19: 716–723.
- Breslow, N. E. 1970. A generalized Kruskal–Wallis test for comparing k samples subject to unequal patterns of censorship. *Biometrika* 57: 579–594.
- . 1974. Covariance analysis of censored survival data. *Biometrics* 30: 89–99.
- Casella, G. and R. L. Berger. 2002. *Statistical Inference*. 2d ed. Pacific Grove, CA: Duxbury.
- Cleves, M. A. 1999. ssa13: Analysis of multiple failure-time data with Stata. *Stata Technical Bulletin* 49: 30–39. In *Stata Technical Bulletin Reprints*, vol. 9, 338–349. 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.
- 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.
- Efron, B. 1977. The efficiency of Cox's likelihood function for censored data. *Journal of the American Statistical Association* 72: 557–565.
- Garrett, J. M. 1997. Graphical assessment of the Cox model proportional hazards assumption. *Stata Technical Bulletin* 35: 9–14. In *Stata Technical Bulletin Reprints*, vol. 6, 38–44. College Station, TX: Stata Press.
- Gehan, E. A. 1965. A generalized Wilcoxon test for comparing arbitrarily singly censored data. *Biometrika* 52: 203–223.
- Grambsch, P. M. and T. M. Therneau. 1994. Proportional hazards tests and diagnostics based on weighted residuals. *Biometrika* 81: 515–526.
- Greenwood, M. 1926. The natural duration of cancer. *Reports on Public Health and Medical Subjects* 33: 1–26.
- Gutierrez, R. G. 2002. Parametric frailty and shared frailty survival models. *The Stata Journal* 2(1): 22–44.
-

- Harrington, D. P. and T. R. Fleming. 1982. A class of rank test procedures for censored survival data. *Biometrika* 69: 553–566.
- Hess, K. R. 1995. Graphical methods for assessing violations of the proportional hazards assumption in Cox regression. *Statistics in Medicine* 14: 1707–1723.
- Hosmer, D. and S. Lemeshow. 1999. *Applied Survival Analysis: Regression Modeling of Time to Event Data*. New York: John Wiley & Sons.
- Hougaard, P. 1984. Life table methods for heterogeneous populations: Distributions describing the heterogeneity. *Biometrika* 71: 75–83.
- . 1986a. A class of multivariate failure time distributions. *Biometrika* 73: 671–8.
- . 1986b. Survival models for heterogeneous populations derived from stable distributions. *Biometrika* 73: 397–96.
- Kalbfleisch, J. D. and R. L. Prentice. 2002. *The Statistical Analysis of Failure Time Data*. 2d ed. New York: John Wiley & Sons.
- Kaplan, E. L. and P. Meier. 1958. Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association* 53: 457–481.
- Klein, J. P. and M. L. Moeschberger. 2003. *Survival Analysis: Techniques for Censored and Truncated Data*. 2d ed. New York: Springer.
- Lancaster, T. 1979. Econometric methods for the duration of unemployment. *Econometrica* 47: 939–56.
- 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.
- Mantel, N. and W. Haenszel. 1959. Statistical aspects of the analysis of data from retrospective studies of disease. *Journal of the National Cancer Institute* 22: 719–748.
- McGilchrist, C. A. and C. W. Aisbett. 1991. Regression with frailty in survival analysis. *Biometrics* 47: 461–466.
- Nelson, W. 1972. Theory and applications of hazard plotting for censored failure data. *Technometrics* 14: 945–965.
- Peto, R. and J. Peto. 1972. Asymptotically efficient rank invariant test procedures (with discussion). *Journal of the Royal Statistical Society* 135: 185–206.
- Prentice, R. L. 1978. Linear rank tests with right-censored data. *Biometrika* 65: 167–179.
- Schoenfeld, D. 1982. Partial residuals for the proportional hazards regression model. *Biometrika* 69: 239–241.

- Tarone, R. E. and J. H. Ware. 1977. On distribution-free tests for equality of survival distributions. *Biometrika* 64: 156–160.
- Therneau, T. M. and P. M. Grambsch. 2000. *Modeling Survival Data: Extending the Cox Model*. New York: Springer.
- Vaupel, J. W., K. Manton, and E. Stallard. 1979. The impact of heterogeneity in individual frailty on the dynamics of mortality. *Demography* 16: 439–54.

Author index

A

Aalen, O. O. 5, 106, 107
 Aisbett, C. W. 280
 Akaike, H. 246, 249

B

Berger, R. L. 21
 Breslow, N. E. 113, 116, 142

C

Casella, G. 21
 Cleves, M. A. 284
 Cox, D. R. 4, 121, 190

E

Efron, B. 143

F

Fleming, T. R. 113, 117

G

Garrett, J. M. 181
 Gehan, E. A. 113, 116
 Grambsch, P. M. 152, 178, 179
 Greenwood, M. 96
 Gutierrez, R. G. 278, 289, 290

H

Haenszel, W. 113, 114
 Harrington, D. P. 113, 117
 Hess, K. R. 181
 Hosmer, D. W., Jr. 109
 Hougaard, P. 279

K

Kalbfleisch, J. D. 121, 131, 133, 138
 Kaplan, E. L. 5, 93
 Klein, J. P. 108, 112, 235

L

Lancaster, T. 279
 Lemeshow, S. 109
 Lin, D. Y. 150

M

Mantel, N. 113, 114
 Manton, K. 279
 McGilchrist, C. A. 280
 Meier, P. 5, 93
 Moeschberger, M. L. 108, 112, 235

N

Nelson, W. 5, 106

P

Peto, J. 113, 117
 Peto, R. 113, 117
 Prentice, R. L. 96, 113, 117, 121, 131,
 133, 138

S

Schoenfeld, D. 178
 Snell, E. J. 190
 Stallard, E. 279

T

Tarone, R. E. 113
 Therneau, T. M. 152, 178, 179

V

Vaupel, J. W. 279

W

Ware, J. H. 113, 116
 Wei, L. J. 150

Subject index

A

Aalen–Nelson estimate see
 Nelson–Aalen estimate
additive-hazards model 20
AFT metric
 definition 207–209
Akaike information criterion 246,
 249–250, 285

B

Bonferroni adjustment 176
Breslow approximation for tied failures
 142–143

C

categorical variables 158–160, 277
censored normal regression
 see regression models,
 censored normal
ci 92
cnreg 2, 201
conditional logistic regression see
 regression models, conditional
 logistic
contingency table 114–115
Cox regression .. see regression models,
 Cox
Cox–Snell residuals 190–193, 252,
 262–264
cumulative hazard
 count data interpretation ... 14–16
cumulative hazard function
 relation to survivor function 9

D

date format 48–50
deviance residuals 253
dispersion parameter 233

E

efficient score residuals 193–195
Efron approximation for tied failures ..
 143
empirical distribution function .. 99–100
enduring variables 42, 47, 76
Euler’s constant 223, 233, 256
exponential regression ... see regression
 models, exponential

F

frailty models
 shared 147–155, 292–298
 unshared 279–292

G

gamma function 17, 223, 233, 244,
 255, 256
generalized gamma regression see
 regression models, generalized
 gamma
Gompertz regression see regression
 models, Gompertz
Gumbel distribution 223, 233

H

hip fracture data
 description 82–89

I

indicator variables see categorical
 variables

instantaneous variables .. 42, 47, 54, 63
 interaction terms 162–165
 interval regression see regression
 models, interval
intreg 201

K

Kaplan–Meier estimate 5, 93–113,
 117, 130–131, 182–184, 198
 kernel smoothing 111–113, 131–133
 Kidney data 148–155, 280–284

L

likelihood-ratio test 151, 247–248,
 289, 290, 295
linktest 175–176
 log-rank test 5, 113–115, 186
logistic 3
 logistic regression see regression
 models, logistic
lrtest 247–248
 lung cancer 25–26

M

martingale residuals 186–190, 253
 maximum likelihood estimation .. 20–21,
 31–35, 38, 138–139, 199, 213–
 214
 Mill's ratio 7
mkmat 194

N

Nelson–Aalen estimate 5, 105–111,
 130–131, 191, 192
 nested models 246–249

O

OLS regression .. see regression models,
 OLS

P

partial likelihood 138
 penalized likelihood 152
 percentiles 11

PH metric

 definition 201–207

predict 251–264

Probability Integral Transform 11

product limit estimate see
 Kaplan–Meier estimate

proportional hazard assumption

 graphical assessment 181–184

 test of 178–181

proportional hazards regression see
 regression models, Cox

Q

quantile function 11–13
 for Weibull 11

R

random number generator 11

regress 1, 201

regression models

 censored normal 2, 201

 conditional logistic 4, 174

 Cox 4, 6, 21–24, 119–195, 197,
 202–206, 209, 222

 exponential 20, 24, 215–224

 generalized gamma 244–246

 Gompertz 234–237

 interval 201

 log-logistic 240–243

 log-normal 237–240

 logistic 3

 OLS 1–2, 21

 Weibull 224–234

reshape 42, 45

residuals

 Cox–Snell residuals .. 190–193, 252,
 262–264

 deviance residuals 253

 efficient score residuals ... 193–195

 martingale residuals .. 186–190, 253

 scaled Schoenfeld residuals ... 178–
 181

 Schoenfeld residuals 178–181

Reye's syndrome data

 description 185

S

- sample size 40
- scaled Schoenfeld residuals 178–181
- Schoenfeld residuals 178–181
- simulate 15
- simulation 13
- snapshot data 41–44
- snapspan 44–46
- stcox
 - basechazard() 146
 - basesurv() 146
 - efron 143
 - exactm 141, 170
 - exacttp 142, 170, 174
 - nohr 122, 124, 125, 160
 - tvc() 167–169, 176–177
- stcoxkm 181–184
- stcurve 133–134, 154, 264–268
- stdes 73, 77–78
- stfill 73, 80–82
- stjoin 172
- stphplot 181–184
- stphtest 178–181
- stratification
 - Cox models 144–147
 - nonparametric tests 117–119
 - parametric models 275–278
- streg
 - ancillary() 269–274
 - anc2() 269
 - dist() 201
 - nohr 206, 215
 - strata() 275–278
 - time 233
- streset 73
- sts 91
 - sts generate .. 108, 130, 175, 192
 - sts graph 102–105, 107, 113
 - sts list 95, 98, 99, 101–102, 106
 - sts test 113–119, 186
- stset 46–71
 - analysis time 52–55
 - enter() 62–64
 - exit() 59–62
- stset, *continued*
 - failure() 57–58
 - id() 65–66
 - PROBABLE ERROR 65, 73–76
 - time0() 66–69
 - variables defined by 55–56
- stsplit...169–174, 177, 217, 224, 228, 283, 296
- stvary 73, 78–80
- survivor function
 - conditional 10, 13
 - relation to cumulative hazard...9
 - test 113–119, 186

T

- test
 - Kaplan–Meier survivor function ..
..... *see* sts test
 - likelihood-ratio *see*
likelihood-ratio test
 - log-rank *see* log-rank test
 - nonparametric stratification
..... *see* stratification,
nonparametric tests
 - proportional hazards assumption..
..... *see* proportional hazard
assumption
 - Wald *see* Wald test
 - Wilcoxon *see* Wilcoxon test
- tied failure times 33, 140–143
- time-varying covariates
25, 84, 166–174, 199, 209–212,
214, 216, 259, 268

U

- uniform distribution 11, 33

V

- variables
 - categorical 158–160, 277
 - enduring 42, 47, 76
 - indicator.. *see* categorical variables
 - instantaneous 42, 47, 54, 63

W

Wald test 125, 161, 226, 246–248,
271–273, 278, 285

Weibull

functions of 9

mean and median of 17

regression model see regression
models, Weibull

Wilcoxon test 116–117

X

xi 159, 278