

STAT 6390: Analysis of Survival Data

Textbook coverage: Chapter 1

Steven Chiou

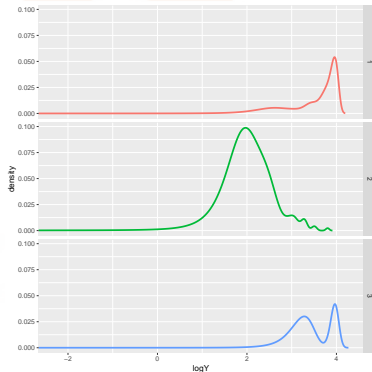
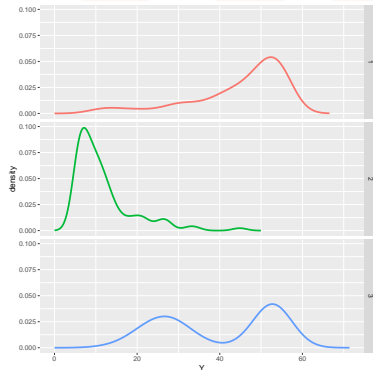
Department of Mathematical Sciences,
University of Texas at Dallas

Survival analysis?

- Survival analysis aka
 - duration analysis
 - event history analysis
 - time to event analysis
- Models the relationship between duration (Y) and covariates (X).
 - time until graduation
 - time until failure of an electronic component
 - time until a patient dies
- Linear regression, e.g., ordinary least squares $\ln(Y \sim X)$, is usually not feasible.

Why not use OLS?

- Inference for OLS assumes Y is normal.
 - Duration is always positive.
 - Duration is usually not-normal.
 - Log-transformation might not work.



Why not use OLS?

- OLS handles missing values via complete case analysis or imputation*.
 - Survival data consists of missing values that are meaningful, so dropping incomplete observations means losing information.
 - Imputation requires additional assumption from the distribution of Y .
 - Replacing missing values with mean or median would result in underestimation if the missingness are caused by *right censoring*.
- Common source of missing values in survival data: *censoring* and *truncation*.

Other reasons for survival models

- Survival models can handle time-varying covariates.
- Probabilities associated with survival times is more relevant.
- Many existing packages make routine survival analysis more accessible.
- A partial list of \mathbb{R} package can be found here:

<https://cran.r-project.org/web/views/Survival.html>

Censoring

- The survival time of an individual is said to be *right censored* when the end-point of interest has not been observed for that individual.
- The “end-point” is a well-defined event, say death from a disease.
- The actually survival time can be regarded as right censored when
 - lost to follow-up
 - death from a different cause
 - no event had occurred by the end of the study

Loading `survMisc`, Ver 0.4.6.

- Most datasets in the book are available via R package **survMisc**
- Some datasets are only available in version 0.4.6 or earlier.
- Archived R package can be installed with

```
> ## install.package("devtools")  
> library(devtools)  
> install_version("survMisc", version = "0.4.6")
```

- `install.package()` installs the latest version.
- `install_version()` installs a specified package.

WHAS

- Load data from Worcester Heart Attach Study (WHAS) in Table 1.1:

```
> data(whas100, package = "survMisc")
```

- The above code only works with **survMisc** version $\leq 0.4.6$.

```
> head(whas100)
```

	id	admitdate	foldate	los	lenfol	fstat	age	gender	bmi
1	1	3/13/1995	3/19/1995	4	6	1	65	0	31.38134
2	2	1/14/1995	1/23/1996	5	374	1	88	1	22.65790
3	3	2/17/1995	10/4/2001	5	2421	1	77	0	27.87892
4	4	4/7/1995	7/14/1995	9	98	1	81	1	21.47878
5	5	2/9/1995	5/29/1998	4	1205	1	78	0	30.70601
6	6	1/16/1995	9/11/2000	7	2065	1	82	1	26.45294

- A description of `whas100` can be called from

```
> ?whas100
```

```
> ?survMisc::whas100
```

- `whas100` is a `data.frame`.

```
> class(whas100)
```

```
[1] "data.frame"
```


WHAS

- A more effective way to manipulate data frame is through “tibble”.
- Install **tidyverse** (<https://www.tidyverse.org>)

```
> ## install.package(tidyverse)
> library(tidyverse)
> whas100 <- as.tibble(whas100)
> whas100
# A tibble: 100 x 9
   id admitdate foldate los lenfol fstat age gender bmi
  <int> <fct>    <fct>    <int> <int> <int> <int> <int> <dbl>
1     1 3/13/1995 3/19/1995     4     6     1    65     0 31.4
2     2 1/14/1995 1/23/1996     5    374     1    88     1 22.7
3     3 2/17/1995 10/4/2001     5   2421     1    77     0 27.9
4     4 4/7/1995  7/14/1995     9     98     1    81     1 21.5
5     5 2/9/1995  5/29/1998     4   1205     1    78     0 30.7
6     6 1/16/1995 9/11/2000     7   2065     1    82     1 26.5
7     7 1/17/1995 10/15/1997     3   1002     1    66     1 35.7
8     8 11/15/1994 11/24/2000    56   2201     1    81     1 28.3
9     9 8/18/1995  2/23/1996     5    189     1    76     0 27.1
10    10 7/22/1995 12/31/2002     9   2719     0    40     0 21.8
# ... with 90 more rows
```

WHAS

- A transposed version to print whas100:

```
> ## install.package(tidyverse)
> glimpse(whas100)
Observations: 100
Variables: 9
 $ id      <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1...
 $ admitdate <fct> 3/13/1995, 1/14/1995, 2/17/1995, 4/7/1995, 2/9/1995,...
 $ folddate  <fct> 3/19/1995, 1/23/1996, 10/4/2001, 7/14/1995, 5/29/199...
 $ los      <int> 4, 5, 5, 9, 4, 7, 3, 56, 5, 9, 6, 11, 6, 10, 7, 5, 6...
 $ lenfol    <int> 6, 374, 2421, 98, 1205, 2065, 1002, 2201, 189, 2719,...
 $ fstat     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1...
 $ age       <int> 65, 88, 77, 81, 78, 82, 66, 81, 76, 40, 73, 83, 64, ...
 $ gender    <int> 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0...
 $ bmi       <dbl> 31.38134, 22.65790, 27.87892, 21.47878, 30.70601, 26...
```

- See <https://r4ds.had.co.nz/tibbles.html> for details.

WHAS

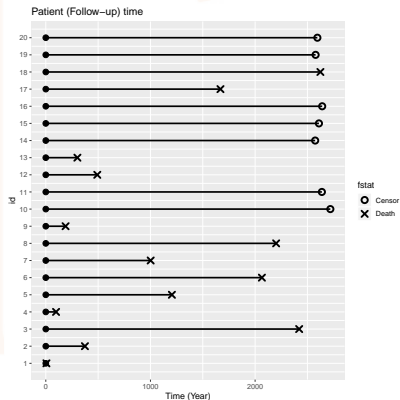
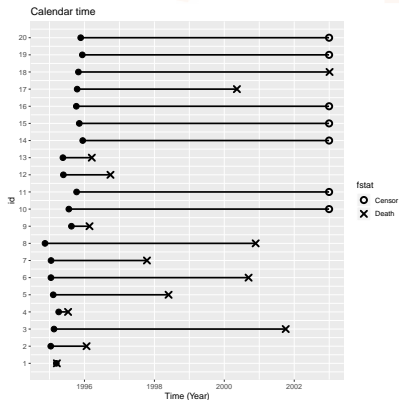
- Here is the screen shot of Table 1.1:

ID	Admission Date	Follow Up Date	Length of Stay	Follow Up Time	Vital Status	Age at Admission	Gender	BMI
1	3/13/95	3/19/95	4	6	Dead	65	Male	31.4
2	1/14/95	1/23/96	5	374	Dead	88	Female	22.7
3	2/17/95	10/4/01	5	2421	Dead	77	Male	27.9
4	4/7/95	7/14/95	9	98	Dead	81	Female	21.5
5	2/9/95	5/29/98	4	1205	Dead	78	Male	30.7
6	1/16/95	9/11/00	7	2065	Dead	82	Female	26.5
7	1/17/95	10/15/97	3	1002	Dead	66	Female	35.7
8	11/15/94	11/24/00	56	2201	Dead	81	Female	28.3
9	8/18/95	2/23/96	5	189	Dead	76	Male	27.1
10	7/22/95	12/31/02	9	2719	Alive	40	Male	21.8

- `los` corresponds to length of stay
- `fstat` corresponds to the vital status; this is also called the *status indicator*, or the *censoring indicator*.
 - It takes the value of 1 if an event has observed (death) and 0 otherwise.

WHAS

- There are two common ways to display follow-up times



WHAS

- Patients are *not* all recruited at exactly the same time.
- The end of study appear to be Jan. 05, 2003.

```
> max(strptime(whas100$foldate, format = "%m/%d/%Y"))
[1] "2003-01-05 CST"
```

- Patients remain alive at the end of study,
 - patient # 10, 11, 14, 15, 16, etc.
- or left the study by then are considered (right) censored.
 - none in this study.
- In the above figures, the **X** marks the events.
- There are two types of censoring:
 - Informative; dropout related to the outcome
 - Non-informative (independent); dropout not related to the outcome

Surv objects

- In this course, we will use t to denote the duration (right figure).
- The `Surv` function in the **survival** package produces a special structure for survival data:

```
> library(survival)
> args(Surv)
function (time, time2, event, type = c("right", "left", "interval",
    "counting", "interval2", "mstate"), origin = 0)
NULL
```

- Similar structure is adopted to several packages. For examples,

```
> args(reda::Survr)
function (ID, time, event, origin = 0, check = TRUE, ...)
NULL
> args(reReg::reSurv)
function (time1, time2, id, event, status, origin = 0)
NULL
```

Surv objects

- For the WHAS, the Surv object is

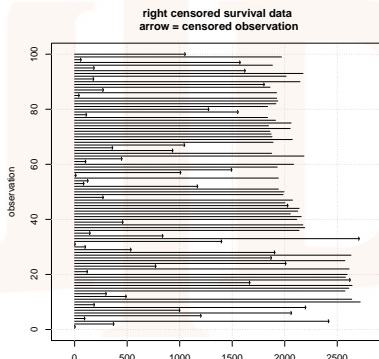
```
> whas100 %>% with(Surv(lenfol, fstat))
 [1]      6      374      2421      98      1205      2065      1002      2201      189      2719+      2638+
[12]     492      302      2574+      2610+      2641+      1669      2624      2578+      2595+      123      2613+
[23]     774      2012      2573+      1874      2631+      1907      538      104      6      1401      2710
[34]     841      148      2137+      2190+      2173+      461      2114+      2157+      2054+      2124+      2137+
[45]    2031      2003+      2074+      274      1984+      1993+      1939+      1172      89      128      1939+
[56]      14      1011      1497      1929+      2084+      107      451      2183+      1876+      936      363
[67]    1048      1889+      2072+      1879+      1870+      1859+      2052+      1846+      2061+      1912+      1836+
[78]     114      1557      1278      1836+      1916+      1934+      1923+      44      1922+      274      1860+
[89]    1806      2145+      182      2013+      2174+      1624      187      1883+      1577      62      1969+
[100]    1054
```

- There are 100 observation times, e.g., t_1, \dots, t_{100} .
- Censored events are accompanied with +.
- With the definition Y is the exact event time, C is the censoring time, then $T = \min(Y, C)$ is the *observed* event time.

Surv objects

- The `Surv` can be plotted with R's generic function `plot`.
- When **`survMisc`** \leq V0.4.6 is loaded, an event plot will be displayed.

```
> whas100 %>% with(Surv(lenfol, fstat)) %>% plot
```



- This feature has been deprecated with newer version of **`survMisc`**, where a *Kaplan-Meier* curve will be shown.

reSurv objects

- Although `whas100` does not contain recurrent event data, a similar event plot can be produced with package **reReg**.
- The latest (development) version of **reReg** can be installed via GitHub.

```
> ## devtools::install_github("stc04003/reReg")  
> library(reReg)
```

- A `reSurv` object must be declared first.

reSurv objects

- reSurv prints a list-column tibble.

```
> with(whas100, reSurv(lenfol, id, rep(0, 100), fstat))
# A tibble: 100 x 5
   id tij   recType   Yi status
<int> <lgl> <lgl>   <dbl> <dbl>
1     1 NA      NA         6      1
2     2 NA      NA        374      1
3     3 NA      NA       2421      1
4     4 NA      NA         98      1
5     5 NA      NA       1205      1
# ... with 95 more rows
```

- One way to plot the event plot with **reReg** is through `plotEvents`

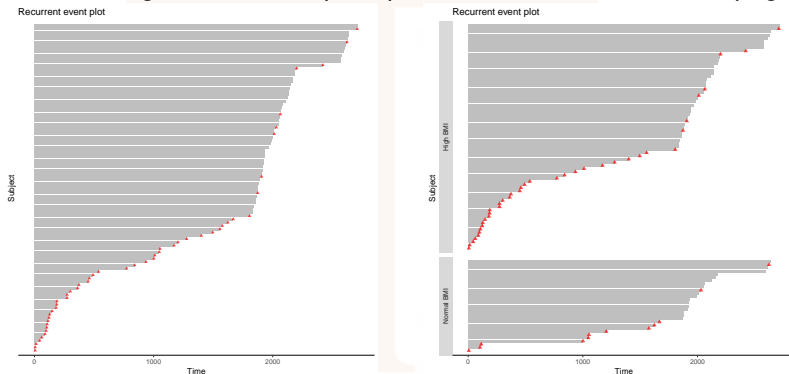
```
> plotEvents(reSurv(lenfol, id, rep(0, 100), fstat) ~ 1, data = whas100)
```

- The function `plotEvents` also allows stratifications.

```
> plotEvents(reSurv(lenfol, id, rep(0, 100), fstat) ~ bmi2,
+           data = whas100 %>% mutate(bmi2 = factor(bmi > 30, labels = c("High"
```

reSurv objects

- The following are the event plots produced with the on the last page.



- See <https://github.com/stc04003/reReg> for more details.

Surv objects

- Another example that is subject to right censoring is the Stanford Heart Transplant Data

```
> data(heart)
> head(heart)
```

	start	stop	event	age	year	surgery	transplant	id
1	0	50	1	-17.155373	0.1232033	0	0	1
2	0	6	1	3.835729	0.2546201	0	0	2
3	0	1	0	6.297057	0.2655715	0	0	3
4	1	16	1	6.297057	0.2655715	0	1	3
5	0	36	0	-7.737166	0.4900753	0	0	4
6	36	39	1	-7.737166	0.4900753	0	1	4

```
> heart %>% with(Surv(start, stop, event)) %>% head(14)
```

[1]	(0, 50]	(0, 6]	(0, 1+]	(1, 16]	(0, 36+]	(36, 39]	(0, 18]
[8]	(0, 3]	(0, 51+]	(51, 675]	(0, 40]	(0, 85]	(0, 12+]	(12, 58]

- In this dataset, `start` is the entry time, `stop` is the exit time, and `event` is the censoring indicator where death is indicated by `event = 1`.
- In this example, `Surv` displays the “calendar time”.

Other censoring

- *Left censoring* is encountered when the event of interest has already occurred when observation begins.
 - Less common.
 - If the event of interest has already occurred when observation begins, the subject is usually not selected in the study. If these subjects are left out, this is referred to *length biased sampling* or a special type of *left truncation*.
- *Interval censoring* is when individuals are known to have experienced an event within an interval of time.
 - When either end of the interval is undefined (∞ or 0), this reduced to either the left censoring or right censoring.
 - When the length of interval is small (e.g., $\rightarrow 0$), one might treat events as uncensored.

Left truncation (Section 7.4)

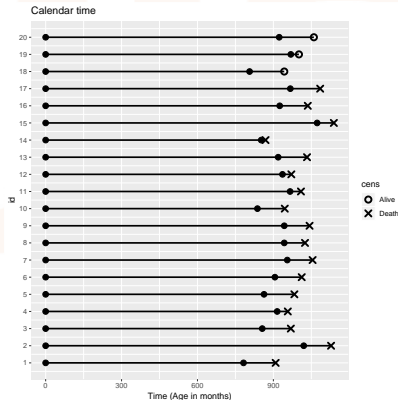
- After right censoring, the next most common source of incomplete observation is *left truncation* or *delayed entry*.
- An example is the Channing House Data, which can be loaded from the **boot** package.

```
> data(channing, package = "boot")
> head(channing)
  sex entry exit time cens
1 Male   782  909  127    1
2 Male  1020 1128  108    1
3 Male   856  969  113    1
4 Male   915  957   42    1
5 Male   863  983  120    1
6 Male   906 1012  106    1
```

- The variables are:
 - entry** age (months) of entry into the retirement home
 - exit** age (months) of exiting the retirement home
 - cens** death status at exit (1 = dead, 0 = alive)

Left truncation

- The data were collected between 1964 and 1975 and feature 52% (right) censoring, as well as left truncation.
- The observed age at death has to be higher than the age at which the subject entered the Channing House retirement house.



Left truncation

- Data that are truncated are unobservable.
- The survival experiences of subjects with delayed entry do not contribute to the analysis until time exceeds an intermediate event.
- If T_i and Y_i are the truncation time and the failure time for the i th patient, respectively. Left truncation implies $T_i < Y_i$.
- Standard survival analysis methods require independent censoring and *quasi-independence* of failure and truncation.