

# Nonparametric Survival Curve Estimation | Confidence Intervals | Left Truncation |

Eric Delmelle | Lehigh University

## Table of contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Nonparametric Estimation of the Survival Function</b>	<b>2</b>
2.1	The Kaplan-Meier (KM) Estimator . . . . .	2
2.2	Example Data and Calculation: Gastric Cancer . . . . .	3
2.3	Confidence Intervals (CI) . . . . .	4
2.3.1	Plain (Linear) Confidence Intervals . . . . .	4
2.3.2	Problems with Plain Confidence Intervals . . . . .	5
2.3.3	Log Transformation . . . . .	5
2.3.4	Complementary Log-Log Transformation (Most Satisfying) . . . . .	6
2.3.5	Why Log-Log is “More Satisfying” . . . . .	7
<b>3</b>	<b>Finding the Median Survival</b>	<b>10</b>
<b>4</b>	<b>Median Follow-Up Time</b>	<b>11</b>
<b>5</b>	<b>Quantiles</b>	<b>12</b>
<b>6</b>	<b>Smoothed Hazard and Survival Function Estimates (Kernel)</b>	<b>12</b>
6.1	Smooth Survival Function from Hazard . . . . .	14
<b>7</b>	<b>Left Truncation</b>	<b>15</b>
7.1	Channing House Example . . . . .	15
7.2	What do we see from the results? . . . . .	20
<b>8</b>	<b>Key Takeaways</b>	<b>21</b>
8.1	Additional Notes . . . . .	21
8.2	Exercises . . . . .	21
8.2.1	Exercise 3.1 . . . . .	21

8.2.2	Exercise 3.2 . . . . .	23
8.2.3	Exercise 3.3 . . . . .	23
8.2.4	Exercise 3.4 . . . . .	24

## 1 Introduction

- Many possible hazard function shapes when using parametric models
  - Challenge: which parametric family to choose?
  - For human/animal survival, no single family always fits well
- Nonparametric estimators of survival function
  - Already discussed
  - Most common: **Kaplan–Meier (product-limit) estimator**
  - Confidence intervals; mean and median survival
  - Kernel smoothing
  - **data:** Gastric Cancer Survival dataset
- Left truncation
  - **data:** Channing House Retirement dataset

## 2 Nonparametric Estimation of the Survival Function

### 2.1 The Kaplan-Meier (KM) Estimator

- (we have seen this already)
- Product over the failure times of the conditional probabilities of surviving to the next failure time.

$$\hat{S}(t) = \prod_{t_i \leq t} (1 - \hat{q}_i) = \prod_{t_i \leq t} \left(1 - \frac{d_i}{n_i}\right)$$

- where  $n_i$  is the number of subjects at risk at time  $t_i$ , and  $d_i$  is the number of individuals who fail at that time.

## 2.2 Example Data and Calculation: Gastric Cancer

- Let's work through the Kaplan-Meier calculation using the **Gastric Cancer** Survival Data (see page 6 in the book, section 1.4, example 1.2 - Xelox in patients with advanced gastric cancer).
- The data is loaded with the **asa** package
- A single-arm trial = everyone in the study gets the same treatment (here, XELOX). No control group or comparison arm (like placebo or standard therapy).
- Phase II - mid-stage study where all participants get XELOX chemo; researchers tracked survival outcomes to decide if it's worth moving to a larger, controlled Phase III trial.
- The **Delta** variable tell us whether the event occurred or not (1=dead).
- Note the confidence intervals are set to False here.
- $n = 48 \rightarrow$  total number of patients in the dataset (the trial enrolled 48).
- Events = 32  $\rightarrow$  the number of patients who experienced the event of interest.
- About 10.3 months, 50% of patients had experienced progression or death.

```
# Load required libraries
library(survival)
library(muhaz)
library(asa)
library(boot)

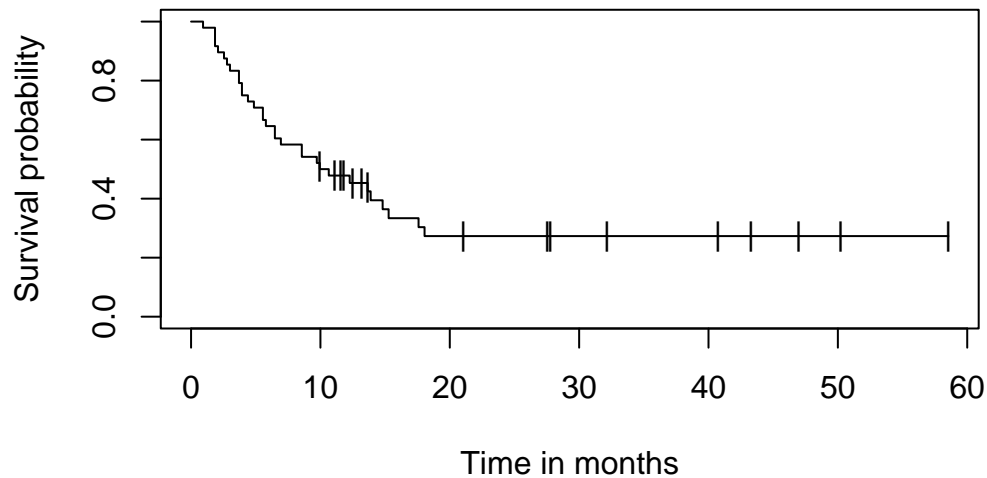
# see example 1.2
timeMonths <- gastricXelox$timeWeeks*7/30.25
delta <- gastricXelox$delta

# Create survival object
result.km <- survfit(Surv(timeMonths, delta) ~ 1, conf.type="none")
print(result.km)
```

```
Call: survfit(formula = Surv(timeMonths, delta) ~ 1, conf.type = "none")
```

```
      n events median
[1,] 48      32  10.3
```

```
plot(result.km, conf.int=F, mark="|", xlab="Time in months",ylab="Survival probability")
```



## 2.3 Confidence Intervals (CI)

- There are several approaches to constructing confidence intervals for the Kaplan-Meier estimator.
- Each with different properties and advantages.
- Plain (can be negative!!!)
- Log
- Log-Log

### 2.3.1 Plain (Linear) Confidence Intervals

The most straightforward approach uses the delta method to obtain the variance of  $\hat{S}(t)$  directly:

$$\text{var}[\hat{S}(t)] \approx [\hat{S}(t)]^2 \sum_{t_i \leq t} \frac{d_i}{n_i(n_i - d_i)}$$

This leads to the **plain confidence interval**:  $\hat{S}(t) \pm z_{\alpha/2} \sqrt{\text{var}[\hat{S}(t)]}$

```
result.km.plain <- survfit(Surv(timeMonths, delta) ~ 1, conf.type = "plain")
# Plot comparison
plot(result.km.plain, conf.int = TRUE, col = "red",
      xlab = "Time in years", ylab = "Survival probability",
      main = "Plain Confidence Intervals")
legend("topright",
      legend = c("Plain"),
      col = c("red"),
```

```
lty = 1)
```

## Plain Confidence Intervals

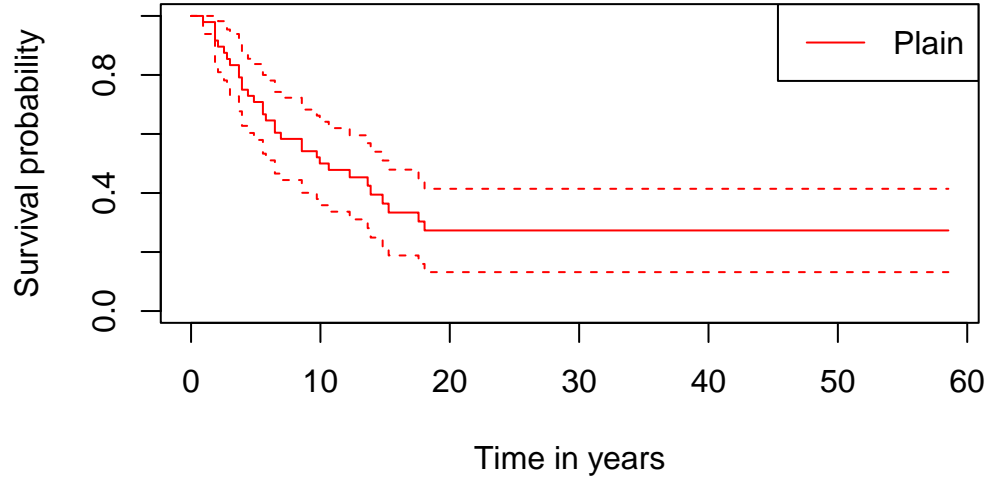


Figure 1: Plain Confidence Intervals

### 2.3.2 Problems with Plain Confidence Intervals

The plain confidence interval has a fundamental problem: **it can extend below 0 or above 1**, which is nonsensical for probabilities. This occurs because:

- The normal approximation may be poor, especially with small sample sizes
- The survival function is bounded between 0 and 1, but the normal distribution is unbounded
- The distribution of  $\hat{S}(t)$  can be quite skewed, especially near the tails

### 2.3.3 Log Transformation

- An improvement uses the log transformation of  $\hat{S}(t)$ :

$$\text{var}[\log \hat{S}(t)] = \sum_{t_i \leq t} \frac{d_i}{n_i(n_i - d_i)}$$

- The log confidence interval is:  $\exp \left\{ \log[\hat{S}(t)] \pm z_{\alpha/2} \sqrt{\text{var}[\log \hat{S}(t)]} \right\}$
- This ensures the confidence interval stays within  $(0, 1]$
- Can still have issues when  $\hat{S}(t)$  approaches 1.

```
# Demonstrate different confidence interval types
result.km.plain <- survfit(Surv(timeMonths, delta) ~ 1, conf.type = "plain")
result.km.log <- survfit(Surv(timeMonths, delta) ~ 1, conf.type = "log")

# Plot comparison
plot(result.km.plain, conf.int = TRUE, col = "red",
      xlab = "Time in years", ylab = "Survival probability",
      main = "Comparison of Confidence Interval Types")
lines(result.km.log, conf.int = TRUE, col = "blue")

legend("topright",
      legend = c("Plain", "Log"),
      col = c("red", "blue"),
      lty = 1)
```

## Comparison of Confidence Interval Types

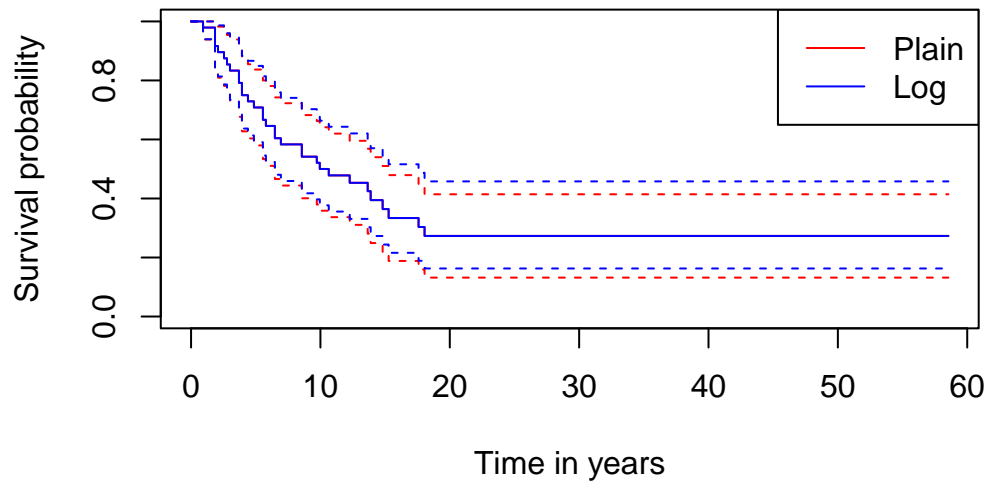


Figure 2: Plain vs Log Confidence Intervals

### 2.3.4 Complementary Log-Log Transformation (Most Satisfying)

- The **complementary log-log transformation** is generally preferred because it has the best statistical properties:

$$\text{var}[\log[-\log \hat{S}(t)]] \approx \frac{1}{[\log \hat{S}(t)]^2} \sum_{t_i \leq t} \frac{d_i}{n_i(n_i - d_i)}$$

- The confidence interval becomes:  $\exp \left\{ -\exp \left[ \log(-\log[\hat{S}(t)]) \pm z_{\alpha/2} \sqrt{\text{var}[\log[-\log \hat{S}(t)]]} \right] \right\}$

```
# Demonstrate different confidence interval types
result.km.plain <- survfit(Surv(timeMonths, delta) ~ 1, conf.type = "plain")
result.km.log <- survfit(Surv(timeMonths, delta) ~ 1, conf.type = "log")
result.km.loglog <- survfit(Surv(timeMonths, delta) ~ 1, conf.type = "log-log")

# Plot comparison
plot(result.km.plain, conf.int = TRUE, col = "red",
      xlab = "Time in years", ylab = "Survival probability",
      main = "Comparison of Confidence Interval Types")
lines(result.km.log, conf.int = TRUE, col = "blue")
lines(result.km.loglog, conf.int = TRUE, col = "green")

legend("topright",
      legend = c("Plain", "Log", "Log-log"),
      col = c("red", "blue", "green"),
      lty = 1)
```

### Comparison of Confidence Interval Types

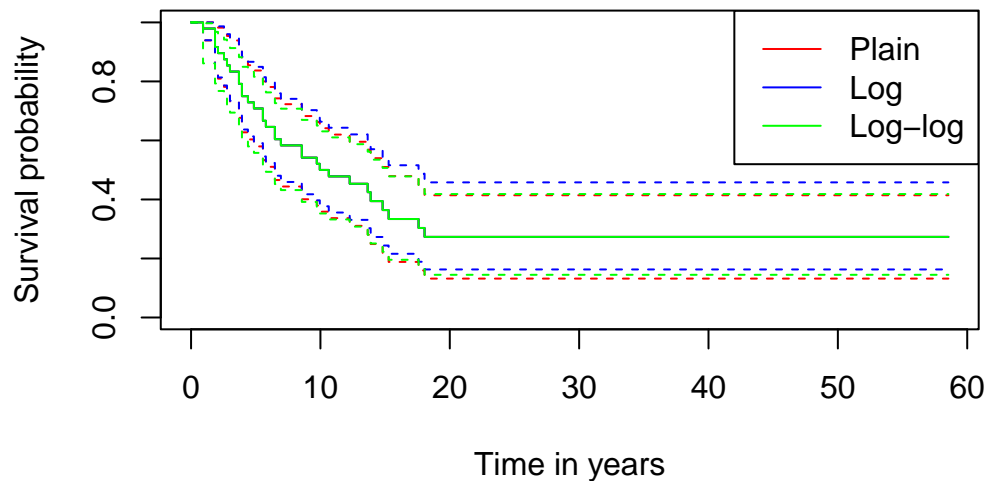


Figure 3: Comparison of All Confidence Interval Types

#### 2.3.5 Why Log-Log is “More Satisfying”

- **Bounded intervals:** Confidence intervals are always between 0 and 1

- **Better symmetry:** The transformation creates more symmetric distributions, making the normal approximation more accurate
- **Improved coverage:** Studies show log-log intervals have coverage probabilities closer to the nominal level (e.g., 95%)
- **Theoretical justification:** The transformation is related to the cumulative hazard function, which has better asymptotic properties
- **Stable behavior:** Performs well even when  $\hat{S}(t)$  is close to 0 or 1

```
# Compare the three methods for our example data
summary(result.km.plain)
```

```
Call: survfit(formula = Surv(timeMonths, delta) ~ 1, conf.type = "plain")
```

time	n.risk	n.event	survival	std.err	lower 95% CI	upper 95% CI
0.926	48	1	0.979	0.0206	0.939	1.000
1.851	47	3	0.917	0.0399	0.838	0.995
2.083	44	1	0.896	0.0441	0.809	0.982
2.545	43	1	0.875	0.0477	0.781	0.969
2.777	42	1	0.854	0.0509	0.754	0.954
3.008	41	1	0.833	0.0538	0.728	0.939
3.702	40	2	0.792	0.0586	0.677	0.907
3.934	38	2	0.750	0.0625	0.628	0.872
4.397	36	1	0.729	0.0641	0.603	0.855
4.860	35	1	0.708	0.0656	0.580	0.837
5.554	34	2	0.667	0.0680	0.533	0.800
5.785	32	1	0.646	0.0690	0.511	0.781
6.479	31	2	0.604	0.0706	0.466	0.743
6.942	29	1	0.583	0.0712	0.444	0.723
8.562	28	2	0.542	0.0719	0.401	0.683
9.719	26	1	0.521	0.0721	0.380	0.662
9.950	25	1	0.500	0.0722	0.359	0.641
10.645	23	1	0.478	0.0722	0.337	0.620
12.264	19	1	0.453	0.0727	0.311	0.596
13.653	16	1	0.425	0.0735	0.281	0.569
13.884	14	1	0.394	0.0742	0.249	0.540
14.810	13	1	0.364	0.0744	0.218	0.510
15.273	12	1	0.334	0.0742	0.188	0.479
17.587	11	1	0.303	0.0734	0.160	0.447
18.050	10	1	0.273	0.0720	0.132	0.414

```
cat("\n--- Log transformation ---\n")
```



--- Log transformation ---

```
summary(result.km.log)
```

Call: survfit(formula = Surv(timeMonths, delta) ~ 1, conf.type = "log")

time	n.risk	n.event	survival	std.err	lower 95% CI	upper 95% CI
0.926	48	1	0.979	0.0206	0.940	1.000
1.851	47	3	0.917	0.0399	0.842	0.998
2.083	44	1	0.896	0.0441	0.813	0.987
2.545	43	1	0.875	0.0477	0.786	0.974
2.777	42	1	0.854	0.0509	0.760	0.960
3.008	41	1	0.833	0.0538	0.734	0.946
3.702	40	2	0.792	0.0586	0.685	0.915
3.934	38	2	0.750	0.0625	0.637	0.883
4.397	36	1	0.729	0.0641	0.614	0.866
4.860	35	1	0.708	0.0656	0.591	0.849
5.554	34	2	0.667	0.0680	0.546	0.814
5.785	32	1	0.646	0.0690	0.524	0.796
6.479	31	2	0.604	0.0706	0.481	0.760
6.942	29	1	0.583	0.0712	0.459	0.741
8.562	28	2	0.542	0.0719	0.418	0.703
9.719	26	1	0.521	0.0721	0.397	0.683
9.950	25	1	0.500	0.0722	0.377	0.663
10.645	23	1	0.478	0.0722	0.356	0.643
12.264	19	1	0.453	0.0727	0.331	0.620
13.653	16	1	0.425	0.0735	0.303	0.596
13.884	14	1	0.394	0.0742	0.273	0.570
14.810	13	1	0.364	0.0744	0.244	0.544
15.273	12	1	0.334	0.0742	0.216	0.516
17.587	11	1	0.303	0.0734	0.189	0.487
18.050	10	1	0.273	0.0720	0.163	0.458

```
cat("\n--- Log-log transformation ---\n")
```

--- Log-log transformation ---

```
summary(result.km.loglog)
```

```
Call: survfit(formula = Surv(timeMonths, delta) ~ 1, conf.type = "log-log")
```

time	n.risk	n.event	survival	std.err	lower 95% CI	upper 95% CI
0.926	48	1	0.979	0.0206	0.861	0.997
1.851	47	3	0.917	0.0399	0.793	0.968
2.083	44	1	0.896	0.0441	0.768	0.955
2.545	43	1	0.875	0.0477	0.743	0.942
2.777	42	1	0.854	0.0509	0.718	0.928
3.008	41	1	0.833	0.0538	0.694	0.913
3.702	40	2	0.792	0.0586	0.647	0.882
3.934	38	2	0.750	0.0625	0.602	0.850
4.397	36	1	0.729	0.0641	0.580	0.833
4.860	35	1	0.708	0.0656	0.558	0.816
5.554	34	2	0.667	0.0680	0.515	0.781
5.785	32	1	0.646	0.0690	0.494	0.763
6.479	31	2	0.604	0.0706	0.452	0.726
6.942	29	1	0.583	0.0712	0.432	0.708
8.562	28	2	0.542	0.0719	0.392	0.670
9.719	26	1	0.521	0.0721	0.372	0.650
9.950	25	1	0.500	0.0722	0.353	0.631
10.645	23	1	0.478	0.0722	0.332	0.610
12.264	19	1	0.453	0.0727	0.308	0.587
13.653	16	1	0.425	0.0735	0.280	0.562
13.884	14	1	0.394	0.0742	0.251	0.535
14.810	13	1	0.364	0.0744	0.223	0.507
15.273	12	1	0.334	0.0742	0.196	0.478
17.587	11	1	0.303	0.0734	0.170	0.449
18.050	10	1	0.273	0.0720	0.145	0.418

### 3 Finding the Median Survival

- The median survival time is defined as:

$$\hat{t}_{med} = \inf\{t : \hat{S}(t) \leq 0.5\}$$

```
# Making sure that we actually generate CI
result.km <- survfit(Surv(timeMonths, delta) ~ 1, conf.type="log-log")
print(result.km)
```

```
Call: survfit(formula = Surv(timeMonths, delta) ~ 1, conf.type = "log-log")
```

```

      n events median 0.95LCL 0.95UCL
[1,] 48      32   10.3    5.79   15.3

```

To find a  $1 - \alpha$  confidence interval for the median, we use:

$$-z_{\alpha/2} \leq \frac{g\{\hat{S}(t)\} - g(0.5)}{\sqrt{\text{var}[g\{\hat{S}(t)\}]}} \leq z_{\alpha/2}$$

where  $g(u) = \log[-\log(u)]$  is the complementary log-log transformation.

## 4 Median Follow-Up Time

- One measure of the quality of a clinical trial is the duration of follow-up.
- The “potential” median survival uses the “reverse” Kaplan-Meier method:

```

# Reverse censoring indicators
delta.followup <- 1 - delta

# Compute reverse Kaplan-Meier
reverse_km <- survfit(Surv(timeMonths, delta.followup) ~ 1)
print(reverse_km)

```

Call: `survfit(formula = Surv(timeMonths, delta.followup) ~ 1)`

```

      n events median 0.95LCL 0.95UCL
[1,] 48      16   27.8    21.1   50.2

```

```

# Compare with simple median
cat("Simple median follow-up:", median(timeMonths), "months\n")

```

Simple median follow-up: 9.950413 months

```

cat("Potential follow-up (reverse KM):", reverse_km$median, "months\n")

```

Potential follow-up (reverse KM): months

## 5 Quantiles

- What is the survival probabilities
- First and third quartiles

```
# Quantiles for gastric cancer data
quantile(result.km, probs = c(0.25, 0.75))
```

```
$quantile
      25      75
4.165289      NA
```

```
$lower
      25      75
2.545455 14.809917
```

```
$upper
      25      75
6.479339      NA
```

- and remember the other question:
- what is my probability to survive at 5 years?

```
result.km <- survfit(Surv(timeMonths, delta) ~ 1, conf.type="log-log")
summary(result.km, times = 5)
```

```
Call: survfit(formula = Surv(timeMonths, delta) ~ 1, conf.type = "log-log")
```

```
time n.risk n.event survival std.err lower 95% CI upper 95% CI
  5      34      14    0.708  0.0656    0.558    0.816
```

## 6 Smoothed Hazard and Survival Function Estimates (Kernel)

- The hazard function estimate at failure times is quite unstable.
- A kernel smoother provides a better visualization.
- This is not a replacement for parametric distribution like `exponential` or `Weibull`, but provides an interesting visualization

$$\hat{h}(t) = \frac{1}{b} \sum_{i=1}^D K\left(\frac{t - t_{(i)}}{b}\right) \frac{d_i}{n_i}$$

where  $K(u)$  is a kernel function (e.g., Epanechnikov kernel:  $K(u) = \frac{3}{4}(1 - u^2)$  for  $-1 \leq u \leq 1$ ).

```
# Piecewise exponential hazard estimate
result.pe5 <- pehaz(timeMonths, delta, width = 5, max.time = 20)
```

```
max.time= 20
width= 5
nbins= 4
```

```
result.pe1 <- pehaz(timeMonths, delta, width = 1, max.time = 20)
```

```
max.time= 20
width= 1
nbins= 20
```

```
# Plot piecewise estimates
plot(result.pe5,
      ylim = c(0, 0.15),
      col = "black",
      xlab = "Time",
      ylab = "Hazard Rate",
      main = "Hazard Function Estimates")
lines(result.pe1, col = "gray")

# Smooth hazard estimate
result.smooth <- muhaz(timeMonths, delta,
                       bw.smooth = 20,
                       b.cor = "left",
                       max.time = 20)
lines(result.smooth, col = "blue", lwd = 2)

legend("topright",
      legend = c("5-month intervals", "1-month intervals", "Smoothed"),
      col = c("black", "gray", "blue"),
      lty = 1,
```

```
lwd = c(1, 1, 2))
```

## Hazard Function Estimates

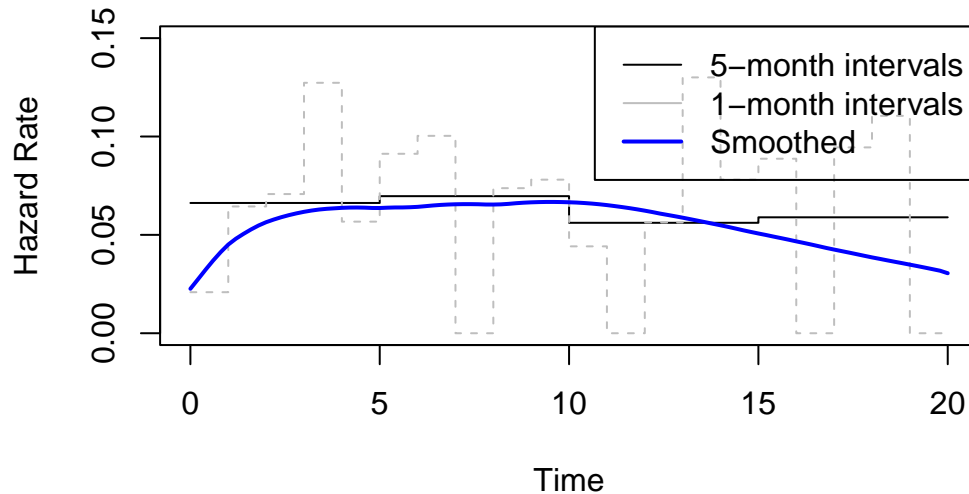


Figure 4: Smoothed Hazard Function Estimate

### 6.1 Smooth Survival Function from Hazard

```
# Extract smoothed hazard estimates
haz <- result.smooth$haz.est
times <- result.smooth$est.grid

# Compute smooth survival function
surv_smooth <- exp(-cumsum(haz[1:(length(haz)-1)] * diff(times)))

# Plot comparison
result.km.compare <- survfit(Surv(timeMonths, delta) ~ 1, conf.type = "none")
plot(result.km.compare,
     conf.int = FALSE,
     mark = "|",
     xlab = "Time in months",
     xlim = c(0, 30),
     ylab = "Survival probability",
     main = "Kaplan-Meier vs Smoothed Survival Estimates")

lines(surv_smooth ~ times[1:(length(times) - 1)], col = "red", lwd = 2)
```

```

legend("topright",
      legend = c("Kaplan-Meier", "Smoothed"),
      col = c("black", "red"),
      lty = 1,
      lwd = c(1, 2))

```

## Kaplan-Meier vs Smoothed Survival Estimates

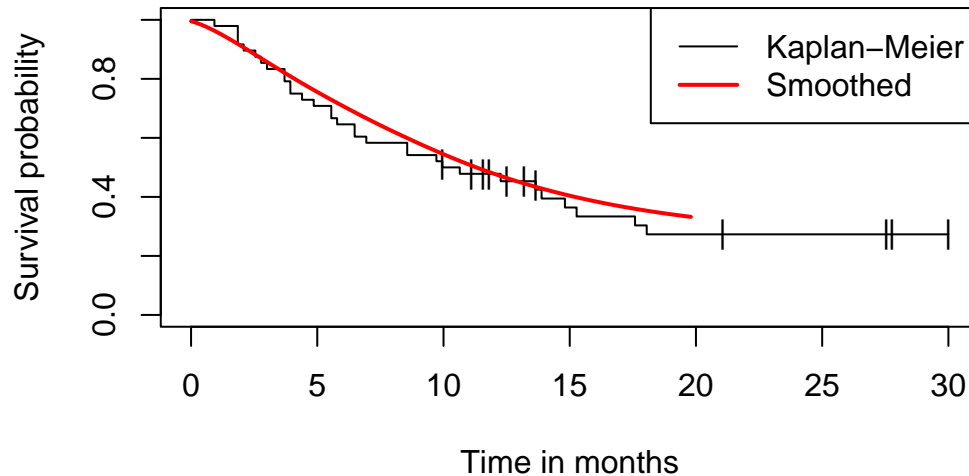


Figure 5: Comparison of Kaplan-Meier and Smoothed Survival Estimates

## 7 Left Truncation

- Left truncation occurs when subjects can only be observed after a certain time point.
- Requires modification of the Kaplan-Meier estimator where subjects enter the risk set at their truncation time.

### 7.1 Channing House Example

- Collection of survival data from the Channing House retirement home in Palo Alto, California, collected between 1964 and 1975.
- It records the age at entry and at death or exit (due to leaving or being alive at the study's end) for 97 men and 365 women, focusing on differences in survival between sexes after accounting for age.
- The data is used for survival analysis and features left truncation (residents entered at different ages, so their pre-Channing House lifetimes were not observed).

- Also right censoring (some residents were still alive at the study's end).

```
# We will use the channing dataset - loaded with 'boot' earlier

# Convert ages from months to years
channing$entryYears <- channing$entry / 12
channing$exitYears <- channing$exit / 12

# Check sex coding
cat("Sex coding in dataset:\n")
```

Sex coding in dataset:

```
#print(table(channing$sex))

# Filter for males using text matching

cat("Age range at entry:", range(channing$entryYears), "\n")
```

Age range at entry: 61.08333 95

```
cat("Age range at exit:", range(channing$exitYears), "\n")
```

Age range at exit: 64.75 100.5833

```
# Standard Kaplan-Meier with left truncation
result.km.standard <- survfit(Surv(entryYears, exitYears, cens, type = "counting") ~ 1,
                             data = channing)

# Conditional on reaching age 68
result.km.68 <- survfit(Surv(entryYears, exitYears, cens, type = "counting") ~ 1,
                       data = channing,
                       start.time = 68)

# Plot all three estimates
plot(result.km.standard,
```



```

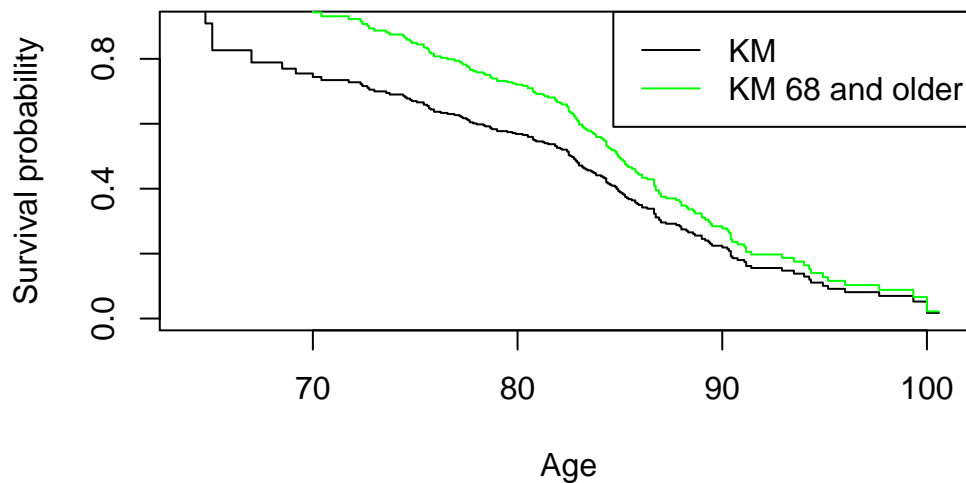
xlim = c(64, 101),
xlab = "Age",
ylab = "Survival probability",
conf.int = FALSE,
main = "Survival Estimates for Channing House Data (Males)")

lines(result.km.68, col = "green", conf.int = FALSE)

legend("topright",
      legend = c("KM", "KM 68 and older"),
      lty = 1,
      col = c("black", "green"))

```

### Survival Estimates for Channing House Data (Males)



```

# Print summaries
cat("\n=== Standard KM with left truncation ===\n")

```

=== Standard KM with left truncation ===

```
print(result.km.standard)
```

```
Call: survfit(formula = Surv(entryYears, exitYears, cens, type = "counting") ~
1, data = channing)
```

```

5 observations deleted due to missingness
records n.max n.start events median 0.95LCL 0.95UCL
[1,]      457    202      11    175    82.7    74.8    85.8

```

```
cat("\n=== KM conditional on age 68+ ===\n")
```

```
=== KM conditional on age 68+ ===
```

```
print(result.km.68)
```

```
Call: survfit(formula = Surv(entryYears, exitYears, cens, type = "counting") ~
1, data = channing, start.time = 68)
```

```

5 observations deleted due to missingness
records n.max n.start events median 0.95LCL 0.95UCL
[1,]      451    202      36    172    84.9    83.8    86.7

```

```
# Remove rows with NA values in key variables
```

```
channing <- channing[complete.cases(channing$entryYears, channing$exitYears, channing$cens), ]
```

```
# Check for problematic observations where exit <= entry
```

```
cat("Observations where exit <= entry:\n")
```

```
Observations where exit <= entry:
```

```

problematic <- channing$exitYears <= channing$entryYears
print(sum(problematic, na.rm = TRUE))

```

```
[1] 5
```

```
# Remove problematic observations
```

```
channing <- channing[channing$exitYears > channing$entryYears, ]
```

```
cat("Remaining observations after cleaning:", nrow(channing), "\n")
```

```
Remaining observations after cleaning: 457
```

```
# Create age group based on entry age
channing$ageGroup <- ifelse(channing$entryYears >= 75, "75+ at entry", "Under 75 at entry")

# Check the distribution
cat("Age group distribution:\n")
```

Age group distribution:

```
print(table(channing$ageGroup))
```

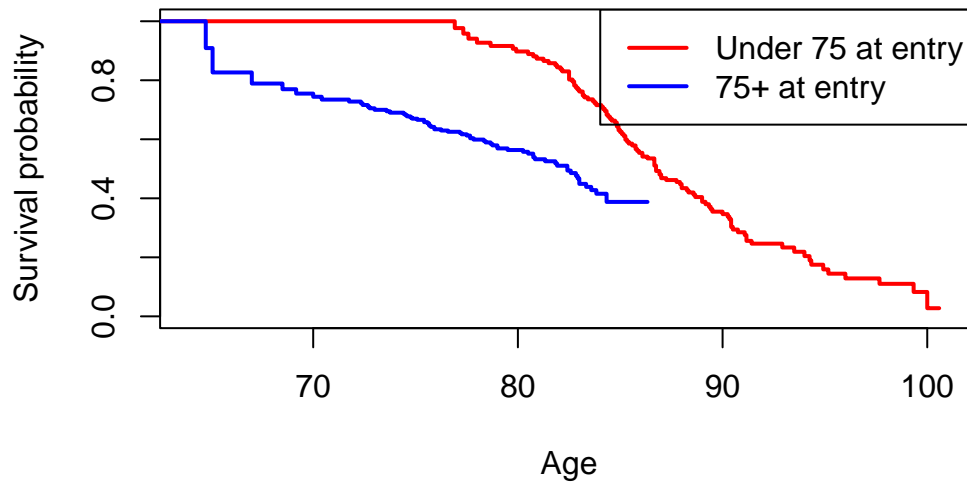
```
75+ at entry Under 75 at entry
          231           226
```

```
# Survival analysis by age group with left truncation
result.km.byage <- survfit(Surv(entryYears, exitYears, cens, type = "counting") ~ ageGroup
                           data = channing)

# Plot comparison
plot(result.km.byage,
     xlim = c(64, 101),
     xlab = "Age",
     ylab = "Survival probability",
     conf.int = FALSE,
     col = c("red", "blue"),
     lty = c(1, 1),
     lwd = 2,
     main = "Survival by Entry Age: Channing House Data")

legend("topright",
     legend = c("Under 75 at entry", "75+ at entry"),
     col = c("red", "blue"),
     lty = c(1, 1),
     lwd = 2)
```

## Survival by Entry Age: Channing House Data



```
# Print summaries for each group
cat("\n=== Survival by Entry Age Group ===\n")
```

```
=== Survival by Entry Age Group ===
```

```
print(result.km.byage)
```

```
Call: survfit(formula = Surv(entryYears, exitYears, cens, type = "counting") ~
  ageGroup, data = channing)
```

	records	n.max	n.start	events	median	0.95LCL	0.95UCL
ageGroup=75+ at entry	231	125	25	111	86.8	85.6	88.7
ageGroup=Under 75 at entry	226	172	11	64	82.4	74.8	NA

### 7.2 What do we see from the results?

- 231 people entered at 75+, 226 entered under 75 (well-balanced groups)
- Median survival age: 86.8 years for 75+ entry group vs 82.4 years for under 75 group
- More events (deaths) in the 75+ group (111 vs 64)
- Be careful!!!
- Left truncation can create complex interpretational challenges in survival analysis
- The “better” survival in the 75+ group might reflect selection effects rather than true differences in longevity.

- **Selection bias:** People who entered at 75+ had to survive to at least 75 to be observed, so they represent a subset who were already “survivors”
- **Different risk periods:** The under 75 group includes people who entered much younger and were observed through more of their life course
- **Survivor effect:** The 75+ entry group might represent individuals with particularly good health/genetics who lived long enough to enter the retirement home at advanced ages

## 8 Key Takeaways

1. **Kaplan-Meier estimator** is the most widely used nonparametric survival function estimator
2. **Confidence intervals** should use the complementary log-log transformation for better properties
3. **Median survival** is the time when the survival function first drops to 0.5 or below
4. **Smoothed hazard functions** provide better visualization than step functions
5. **Left truncation** requires careful handling to avoid bias in survival estimates

### 8.1 Additional Notes

1. The `bshazard` package provides B-spline based smoothing for hazard functions
2. Other percentiles can be estimated similarly to the median using:

$$\hat{t}_p = \inf\{t : \hat{S}(t) \leq 1 - p\}$$

3. Simultaneous confidence bands are available in the `kmconfband` package
4. Right truncation is more complex and requires specialized methods like those in the `DTDA` package

### 8.2 Exercises

#### 8.2.1 Exercise 3.1

Find the median survival and 95% confidence interval from the example data. Explain why the upper limit might be undefined.

```
# Median survival from our example
result.km
```

```
Call: survfit(formula = Surv(timeMonths, delta) ~ 1, conf.type = "log-log")
```

```
      n events median 0.95LCL 0.95UCL  
[1,] 48      32  10.3   5.79   15.3
```

```
summary(result.km)
```

```
Call: survfit(formula = Surv(timeMonths, delta) ~ 1, conf.type = "log-log")
```

time	n.risk	n.event	survival	std.err	lower 95% CI	upper 95% CI
0.926	48	1	0.979	0.0206	0.861	0.997
1.851	47	3	0.917	0.0399	0.793	0.968
2.083	44	1	0.896	0.0441	0.768	0.955
2.545	43	1	0.875	0.0477	0.743	0.942
2.777	42	1	0.854	0.0509	0.718	0.928
3.008	41	1	0.833	0.0538	0.694	0.913
3.702	40	2	0.792	0.0586	0.647	0.882
3.934	38	2	0.750	0.0625	0.602	0.850
4.397	36	1	0.729	0.0641	0.580	0.833
4.860	35	1	0.708	0.0656	0.558	0.816
5.554	34	2	0.667	0.0680	0.515	0.781
5.785	32	1	0.646	0.0690	0.494	0.763
6.479	31	2	0.604	0.0706	0.452	0.726
6.942	29	1	0.583	0.0712	0.432	0.708
8.562	28	2	0.542	0.0719	0.392	0.670
9.719	26	1	0.521	0.0721	0.372	0.650
9.950	25	1	0.500	0.0722	0.353	0.631
10.645	23	1	0.478	0.0722	0.332	0.610
12.264	19	1	0.453	0.0727	0.308	0.587
13.653	16	1	0.425	0.0735	0.280	0.562
13.884	14	1	0.394	0.0742	0.251	0.535
14.810	13	1	0.364	0.0744	0.223	0.507
15.273	12	1	0.334	0.0742	0.196	0.478
17.587	11	1	0.303	0.0734	0.170	0.449
18.050	10	1	0.273	0.0720	0.145	0.418

```
# The upper confidence limit is undefined (NA) when the upper confidence  
# band for the survival curve never drops to 0.5
```

### 8.2.2 Exercise 3.2

Find the first and third quartiles for the gastric cancer data with 95% confidence intervals.

```
# Quantiles for gastric cancer data
quantile(result.km, probs = c(0.25, 0.75))
```

\$quantile

25	75
4.165289	NA

\$lower

25	75
2.545455	14.809917

\$upper

25	75
6.479339	NA

```
# Note: Quartiles are times when S(t) = 0.75 (first quartile) and S(t) = 0.25 (third quartile)
```

### 8.2.3 Exercise 3.3

Create a smooth hazard function estimate with bandwidth 20 and explain any multiple peaks.

```
# Already created in the main text with bw.smooth = 20
plot(result.smooth,
      xlab = "Time",
      ylab = "Hazard Rate",
      main = "Smooth Hazard Function (bandwidth = 20)")
```

### Smooth Hazard Function (bandwidth = 20)

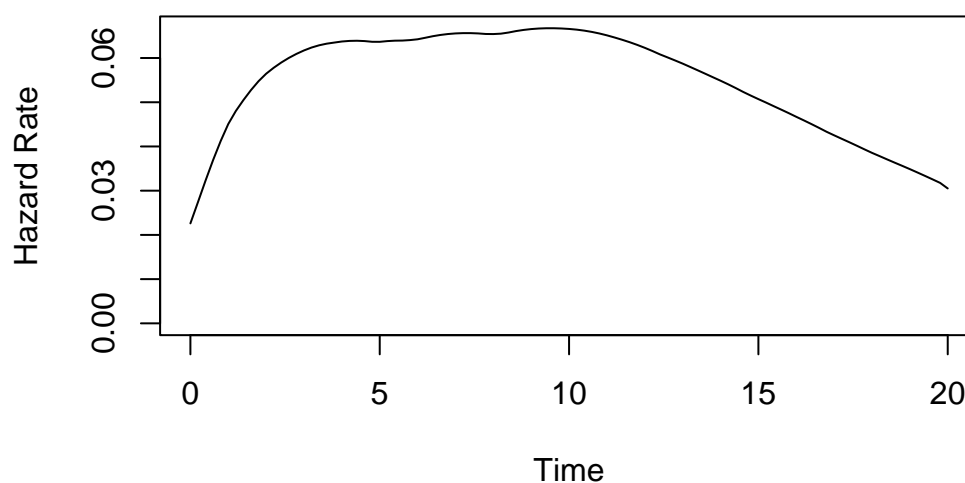


Figure 6: Smooth Hazard with Bandwidth 20

```
# Multiple peaks could indicate:  
# 1. Different risk periods in disease progression  
# 2. Treatment effects wearing off  
# 3. Heterogeneity in patient populations  
# 4. Artifacts from smoothing procedure
```

#### 8.2.4 Exercise 3.4

Compare left-truncated vs. non-truncated estimates and discuss potential bias.

```
# This comparison shows how ignoring left truncation can lead to biased estimates  
# The non-truncated estimate may underestimate survival at younger ages  
# because individuals who died young were never observed in the study
```

---

[← Return to Course Materials](#)