107.330 Statistische Simulation und computerintensive Methoden

Bootstrap Methods and Jackknife

Alexandra Posekany

WS 2020

Bootstrap sampling

Given a data set x of size n, the general strategy for non-parametric bootstrapping is:

1. Identify the parameters of interest θ.
2. Compute the parameters for the given data set, denoted $\hat{\theta}$.
3. Generate m "data sets'' x^i of length n by resampling with replacement from x. x^i is often called the bootstrap sample.
4. Compute for each data set x^i the corresponding bootstrap estimate $\hat{\theta}^i$, $i = 1, \ldots, m$.

The choice of m depends then mainly of what needs to done using the bootstrap estimates.

The jackknife

Quenouille developed the jackknife approach.

Tukey expanded the method and proposed its name as “jack knife” because it is ‘a rough-and-ready tool that can improvise a solution for a variety of problems even though specific problems may be more efficiently solved with a purpose-designed tool’.

Notion of the jackknife

Another resampling method is the jackknife which is actually older than bootstrapping.

The jackknife is basically a "leave-one-out'' approach. For a data sample $x = (x_1, \ldots, x_n)$ with parameter of interest θ, the idea is to:

1. Denote $x^{(i)} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ as the data set which has omitted x_i, for all $i = 1, \ldots, n$.
2. Denote $\hat{\theta}^{(i)}$ as the parameter estimate based on $x^{(i)}$, for all $i = 1, \ldots, n$.

Note that $\hat{\theta}$ and $\hat{\theta}^{(i)}$ are based on different sample sizes!

Nowadays the jackknife is often used together with the bootstrap.

For the jackknife, $\hat{\theta}$ should be a smooth estimate in the sense that small changes in the data cause small changes in the estimate. In this sense the sample mean is smooth but not the sample median.

Jackknife standard error estimation

To estimated the standard error of (a univariate) $\hat{\theta}$ using the jackknife is:

$$\widehat{se}_j(\hat{\theta}) = \sqrt{\frac{n-1}{n}\sum_{i=1}^{n}\left(\hat{\theta}^{(i)} - \bar{\hat{\theta}}_{(\cdot)}\right)^2},$$

where $\bar{\hat{\theta}}_{(\cdot)} = \frac{1}{n}\sum_{i=1}^{n}\hat{\theta}^{(i)}$.

The factor $(n-1)/n$ is needed to make the estimate unbiased.

Unbiasedness of the Jackknife

The mean of Jackknife samples is identical to the sample mean.

$$\underbrace{\bar{x}}_{\text{sample mean}} = \underbrace{\frac{1}{n}\sum_{i=1}^{n}\bar{x}_{(i)}}_{\text{Jackknife mean}} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n-1}\sum_{j=1}^{n-1}x_j^{(-i)} = \frac{1}{n}\sum_{i=1}^{n}\frac{(n-1)\cdot x_i}{n-1}$$

It holds true that the relation between sample standard error and Jackknife standard error is

$$\mathrm{sd}(\bar{x}) = \underbrace{\sqrt{\frac{n-1}{n}\sum_{i=1}^{n}(\bar{x}_i-\bar{x})^2}}_{\text{Jackknife standard error } \hat{se}_j(\hat{\theta})} = \underbrace{\sqrt{\frac{1}{n(n-1)}\sum_{i=1}^{n}(x_i-\bar{x})^2}}_{\text{sample standard error } \hat{s}_n/\sqrt{n}}.$$

Jackknife standard errors example

```
> set.seed(123456)
> n <- 50
> x <- rnorm(n)
> mean(x)
> 1/sqrt(n)
> sd(x)/sqrt(n)
> J.MEANS <- numeric(n)
> for (i in 1:n) {J.MEANS[i] <- mean(x[-i])}
> sqrt((n-1)*mean((J.MEANS-mean(J.MEANS))^2))
```

```
a sample of size 50  with mean 0.117362532834952
and standard error 0.147288076217546 sampled from N(0,1)
which would imply a theoretical mean of 0
and standard error of 0.14142135623731
jackknife with mean 0.117362532834952
and standard error 0.147288076217546
```

Bias estimation with Jackknife

An early motivation for the jackknife was actually to estimate the bias of an estimate: $E(\hat{\theta}) - \theta$.

As θ is usually unknown this has to be solved using resampling. Because of their unbiasedness, jackknife techniques are used to estimate the bias of an estimator calculated over the entire sample.

We look for example at the estimator $\hat{\theta}_{(.)} = \frac{1}{n}\sum_{i=1}^{n}\hat{\theta}_{(i)}$

Then the jackknife estimator for the bias based on "leave-one-out" is

$$\widehat{\text{bias}}_{(\theta)} = (n-1)(\hat{\theta}_{(.)} - \hat{\theta})$$

Based on this we construct the bias-corrected jackknife estimate of θ as $\hat{\theta}_{\text{jack}} = \hat{\theta} - \widehat{\text{bias}}_{(\theta)} = n\hat{\theta} - (n-1)\hat{\theta}_{(.)}$

Jackknife after Bootstrap

Nowadays bootstrap is much more popular than jackknife. However bootstrap estimates are still random variables and therefore also these estimates have bias and standard errors and so on.

If one is interested in these quantities the two procedures can be connected.

In case of the standard error, denote $J(i)$ as all the indices of bootstrap sample which do not contain observation x_i and denote as $m(i)$ the number of bootstrap samples that do not contain x_i. Then to get the jackknife estimate for the bootstrap estimate by using the jackknife replicates which leave out the m-m(i) sample which contain x_i.

Jackknife after Bootstrap for the standard error

The jackknife estimate for the bootstrap estimate of the standard error is then basically

$$\widehat{se}_j(\widehat{se}_{m(1)}, \ldots, \widehat{se}_{m(n)}),$$

where

$$\widehat{se}_{m(i)} = \sqrt{\frac{1}{m(i)} \sum_{i \in J(i)} \left(\hat{\theta}_{(i)} - \bar{\hat{\theta}}_{(i)}\right)^2}$$

and

$$\bar{\hat{\theta}}_{(j)} = \frac{1}{m(i)} \sum_{i \in J(i)} \hat{\theta}_{(i)}.$$

Bias estimation with bootstrap

Assume then again having m bootstrap samples and corresponding estimates $\hat{\theta}^i$ and their mean $\bar{\hat{\theta}} = \frac{1}{m}\sum_{i=1}^{m}\hat{\theta}^i$. Then

$$\widehat{bias} = \sum_{i=1}^{m} \bar{\hat{\theta}} - \hat{\theta}^i.$$

And a bias corrected estimate is

$$\widehat{\theta}_B = \hat{\theta} - \widehat{bias}.$$

Bootstrapping in R

A very good book about bootstrapping is:

- Davison, A. C. and Hinkley, D. V. (1997): Bootstrap Methods and Their Applications.

The functions and datasets from the book are implemented in the package `boot` which is a recommended package.

The package is easy to use when one has a function which takes as input the data and a index vector.

Example using the boot package I

```
> set.seed(234)
> n <- 50;  m <- 200
> x <- rnorm(n)
> mean(x)
[1] -0.06297809
> my.mean <- function(x,i) mean(x[i])
> library(boot)
> BOOTMEAN <- boot(x,my.mean,m)
```

Example using the boot package II

```
> BOOTMEAN

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = x, statistic = my.mean, R = m)


Bootstrap Statistics :
       original       bias    std. error
t1* -0.06297809 0.008982377   0.1293942
```

Example using the boot package III

```
> boot.ci(BOOTMEAN, type=c("norm", "perc", "bca"))
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 200 bootstrap replicates

CALL :
boot.ci(boot.out = BOOTMEAN, type = c("norm", "perc", "bca"))

Intervals :
Level      Normal              Percentile            BCa
95%   (-0.3256,  0.1816 )   (-0.2980,  0.1916 )   (-0.3306,  0.1649 )
Calculations and Intervals on Original Scale
Some percentile intervals may be unstable
Some BCa intervals may be unstable
```

Needs to adapt the bootstrap

The parametric and non-parametric bootstrap works usually quite easily for iid data. There the resampling strategy is always clear.

What however in the context of

- hypothesis testing: the distribution of the data under the null hypothesis is of interest.
- regression data: the data is not iid as the "mean'' of the response is a function of the covariates.
- dependent data: time series or spatial data for example are not iid.

What to do then?

Bootstrapping hypothesis tests

Assume we have the testing problem:

$$H_0 : \theta = \theta_0 \text{ and } H_1 : \theta \neq \theta_0$$

and a corresponding test statistic T.

The key in bootstrapping hypothesis tests is that the sampling should reflect the distribution under H_0. Even if the data at hand was actually not following it!

Parametric bootstrap of hypothesis tests

Under a parametric model this is usually quite simple, as the null hypothesis suggests usually a concrete distribution to sample from.

Assume a sample X_n of size n yielding the test statistic $T = T(X_n)$, the parametric bootstrap is

1. Sample using the null model m samples X_n^* of size n.
2. Compute the m test statistics $T^* = T(X_n^*)$.
3. The bootstrap p-value is then

$$\frac{\#(T^* \text{ more extreme than } T) + 1}{m + 1}$$

Bootstrap from student's t-test

student's t test

$$H_0: \quad \mu = \mu_0 \qquad \text{or} \ \geq \mu_0 \ \text{or} \ \leq \mu_0$$

$$H_A: \quad \mu = \mu_A \ \text{or} \ \neq \mu_0 \ \text{or} \ < \mu_0 \ \text{or} \ > \mu_0$$

- assumption und the H_0: data follow (approximately) a normal distribution $\mathcal{N}(\mu_0, \sigma \approx s_n)$
- test statistics $T(\bar{x}_n) = \frac{\overline{x}_n - \mu_0}{s_n}$ and bootstrap test statistics $T^*(\bar{x}_n^i) = \frac{\overline{x}_n^i - \mu_0}{s_n^i}$ of bootstrap sample $i = 1, \ldots, m$
- theoretically the resulting distribution of the test statistic t is a **student's t** distribution
  with parametric bootstrapping we obtain the bootstrapping distribution under the H_0 and from this the p-value
  $p - value = \frac{\sum_{i=1}^{n} \mathbb{I}(|T^*(\bar{x}_n^i)| \geq |T(\bar{x}_n)|) + 1}{m+1}$

non-parametric bootstrap of hypothesis tests

In non-parametric bootstrapping we again have only the ecdf available. This should be modified in such a way that the null hypothesis is true.

For example if the hypothesis is that in the one sample location problem

$$H_0 : \mu = \mu_0 \text{ vs } H_1 : \mu \neq \mu_0$$

and the estimator is the sample mean.

Here, we can center the data using $\bar{x}$ and add the null hypothesis value μ_0. Then, we resample from these "pseudo" data.

non-parametric bootstrap of hypothesis tests II

Assume you have a scheme to sample under the null then non-parametric bootstrapping for hypothesis testing has the same steps as parametric bootstrapping.

1. Sample using the null model m samples X_n^* of size n.
2. Compute the m test statistics $T^* = T(X_n^*)$.
3. The bootstrap p-value is then

$$\frac{\#(T^* \text{ more extreme than } T) + 1}{m + 1}$$

Hypothesis testing using bootstrap example I

Consider the hypothesis test for $H_0 : \mu_0 = 2$ vs. $H_1 : \mu_0 \neq 2$

```
> set.seed(1)
> # the data
> x <- rnorm(100, mean=1.8)
> mu0 <- 2
> xbar <- mean(x)
> xbar
[1] 1.908887
>
> Tstatistic <- function(x, mu0) (mean(x) - mu0) / sd(x)
> # note I can drop dividing by sqrt(n) as that
> # will be the same for all test statistics
> TX <- Tstatistic(x, mu0=mu0)
```

parametric bootstrap vs. t-test

```
> Tstarparam<-replicate(500,Tstatistic(
+    x = rnorm(length(x),mean = mu0,sd = sd(x)), mu0=mu0))
> (sum(abs(Tstarparam) > abs(TX)) + 1) / 501
[1] 0.3153693
> t.test(x, mu=mu0)

    One Sample t-test

data:  x
t = -1.0144, df = 99, p-value = 0.3129
alternative hypothesis: true mean is not equal to 2
95 percent confidence interval:
 1.730665 2.087110
sample estimates:
mean of x
 1.908887
```

Non-parametric Bootstrap vs. t-test

```
> xNull <- x-xbar+mu0
> Tstarnonparam <- replicate(500, Tstatistic(
+   x=sample(xNull, replace=TRUE), mu0=mu0))
> (sum(abs(Tstarnonparam) > abs(TX)) + 1) / 501
[1] 0.3473054
> t.test(x, mu=mu0)

    One Sample t-test

data:  x
t = -1.0144, df = 99, p-value = 0.3129
alternative hypothesis: true mean is not equal to 2
95 percent confidence interval:
 1.730665 2.087110
sample estimates:
mean of x
 1.908887
```

Sign and permutation hypothesis tests

Resampling methods have actually a long tradition in hypothesis testing.

In so-called sign or permutation tests usually under the null either signs of observations can be "permuted" or other features can be permuted, depending on the null hypothesis. Then new samples are created using the appropriate permutation strategy and again the test statistic from the sample is compared to the test statistics from the permuted data.

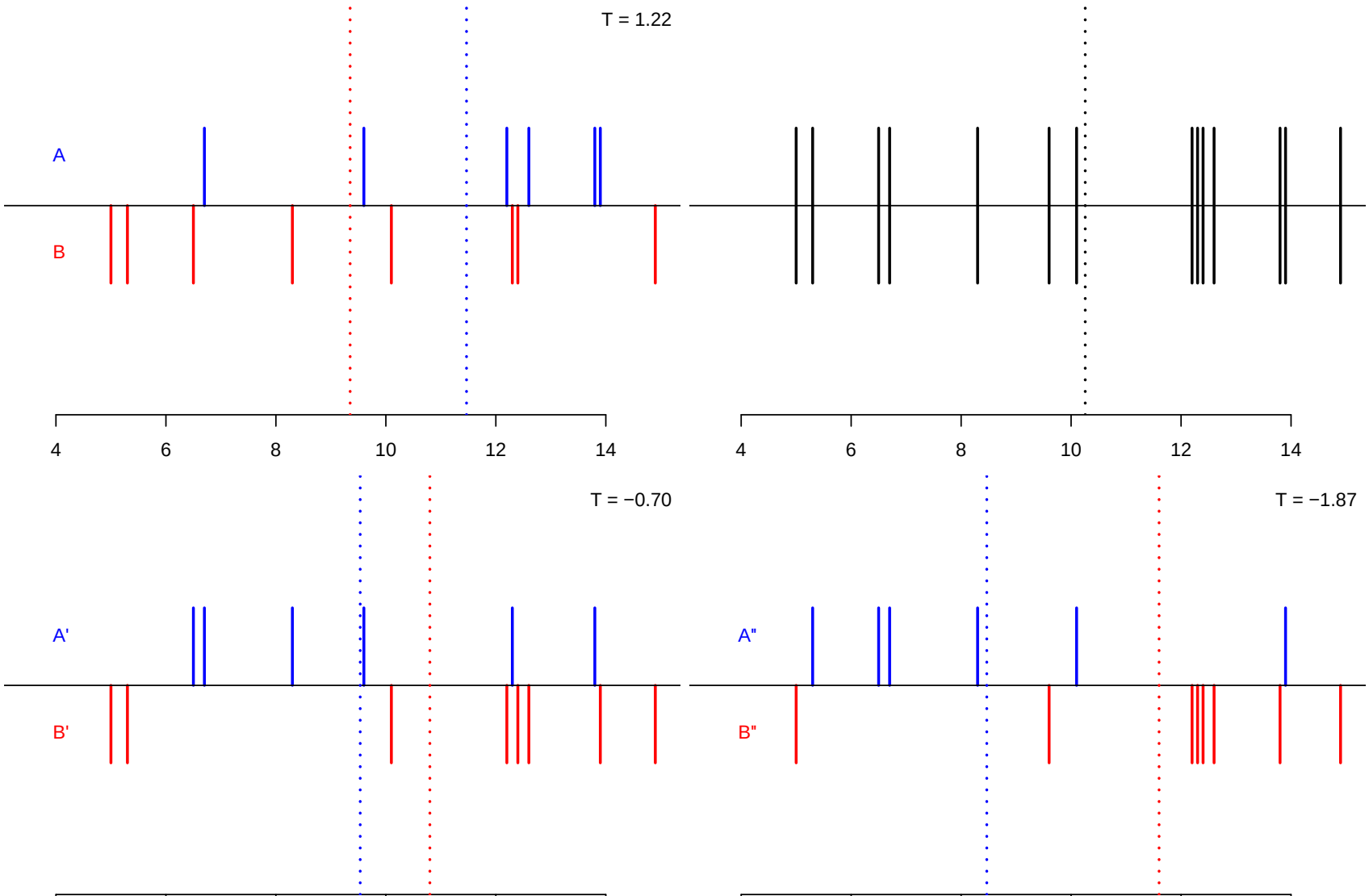
Example of a sign permutation version of the above t test

Assume the previous data and say the null hypothesis is that the symmetry center of the data is μ_0.

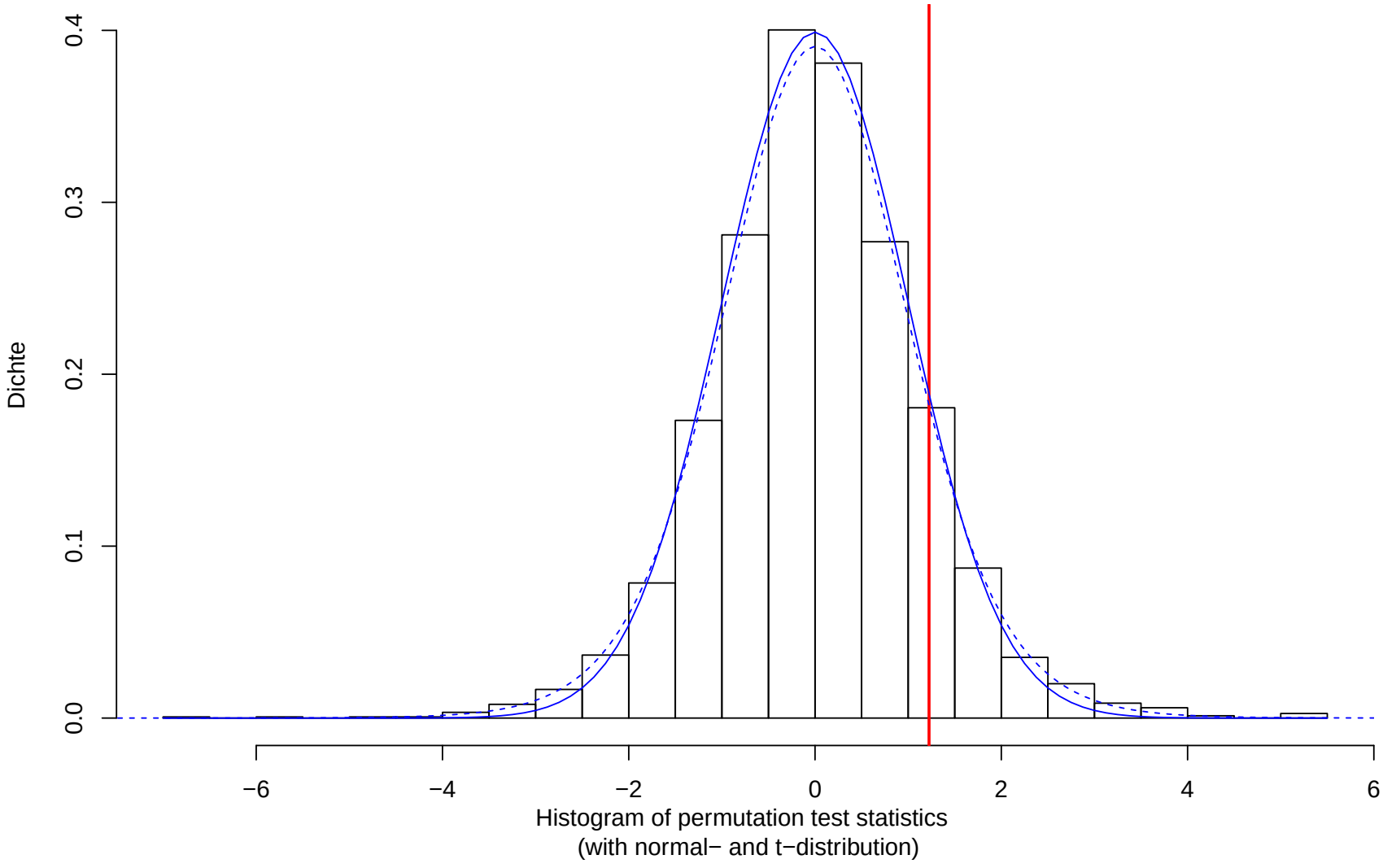
```
> n <- length(x)
> Tsign <- replicate(500, Tstatistic(sample(c(-1,1),n,
+                       replace=TRUE)*(x-mu0), mu0=0))
> (sum(abs(Tsign) > abs(TX)) + 1) / 501
[1] 0.3073852
```

Permutation test vs. 2-sample t-test

The simple idea of the permutation test is to sample from the null hypothesis $H_0 : \mu_1 = \mu_2$ by permuting the labels of the 2 samples.



Permutation test in R



Linear regression

Using the formula representation of the linear model for a dependent variable y and independent explanatory variables x_i corresponding to the model $n \times p$ matrix $(x_1, \ldots, x_p)$, then the regression model for the observations $(x_{1,i}, \ldots, x_{p,i}, y_i)$ for $i = 1, \ldots, n$

$$y_i = \beta_0 + \beta_1 x_{1,i} + \ldots + \beta_p x_{p,i} + \varepsilon_i,$$

models the regression hyperplane

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

Linear Regression Assumptions

The following model assumptions are made:

(A1): $\mathbf{X}$ is nonstochastic, $\mathbf{X}$ has full rank, i.e. the explanatory variables x_i are mutually indepedent.

(A2): The model has no systematic bias, $\mathbf{E}(\varepsilon) = 0$.

(A3): The residual errors are uncorrelated, $cov(\varepsilon_i, \varepsilon_j) = 0$ for $i \neq j$.

(A4): The residual errors are homoscedastic, $cov(\varepsilon) = \sigma^2 \mathbf{I}_n$.

(A5): The residual errors are normally distributed, $\varepsilon \sim N_n(\mathbf{0}, \sigma^2 \mathbf{I}_n)$.

Linear regression II

Which means that under (A1)-(A4):

$$E(y_i) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p \quad \text{and} \quad Var(y_i) = \sigma^2$$

and under (A1)-(A5):

$$y_i \sim N(\beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p, \sigma^2)$$

Based on these two variants we have several options to resample such data.

Bootstrapping linear regression

The iid part of linear repression are actually the errors ε_i, $i = 1, \ldots, n$.

And they are estimated as the residuals of the data:

$$\hat{\varepsilon}_i = y_i - \hat{y}_i$$

Hence, one can resample from these residuals and construct new y values from these.

Parametric Bootstrapping linear regression

The strategy is:

1. Fit a regression model and get estimates for the residuals.
2. Estimate the residual variance $\hat{\sigma}^2$.
3. Sample m times n residuals from $N(0, \hat{\sigma}^2)$.
4. Create the new bootstrap data sets $(\hat{y}_i + \varepsilon_i^*, x_{1,i}, \ldots, x_{p,i})$.
5. Fit the same model for each of the bootstrap data sets.

non-parametric Bootstrapping linear regression

The strategy is:

1. Fit a regression model and get estimates for the residuals.
2. Sample m times n residuals by sampling with replacement from the estimated residuals.
3. Create the new bootstrap data sets $(\hat{y}_i + \varepsilon_i^*, x_{1,i}, \ldots, x_{p,i})$.
4. Fit the same model for each of the bootstrap data sets.

The bootstrapping covariance of parameter estimates

Let $\hat{\boldsymbol{\beta}}$ be the coefficient vector of the parameter estimates of the model estimated from the true sample.

Let $\hat{\boldsymbol{\beta}}_j^*$ be the corresponding coefficient estimate based on the jth bootstrap sample and denote $\bar{\boldsymbol{\beta}} = \frac{1}{m}\sum_{i=j}^{m} \hat{\boldsymbol{\beta}}_j^*$, then the covariance matrix of the bootstrap estimator $\hat{\boldsymbol{\beta}}$ is estimated as

$$\boldsymbol{\Sigma}^* = \frac{1}{m-1}\sum_{i=j}^{m}(\hat{\boldsymbol{\beta}}_j^* - \hat{\boldsymbol{\beta}})(\hat{\boldsymbol{\beta}}_j^* - \hat{\boldsymbol{\beta}})^\top.$$

Bootstrapping pairs in linear regression

An alternative to the residual bootstrapping strategy is simply to resample with replacement the original data points $(y_i, \boldsymbol{x}_i)$, treating them simply as if they were iid.

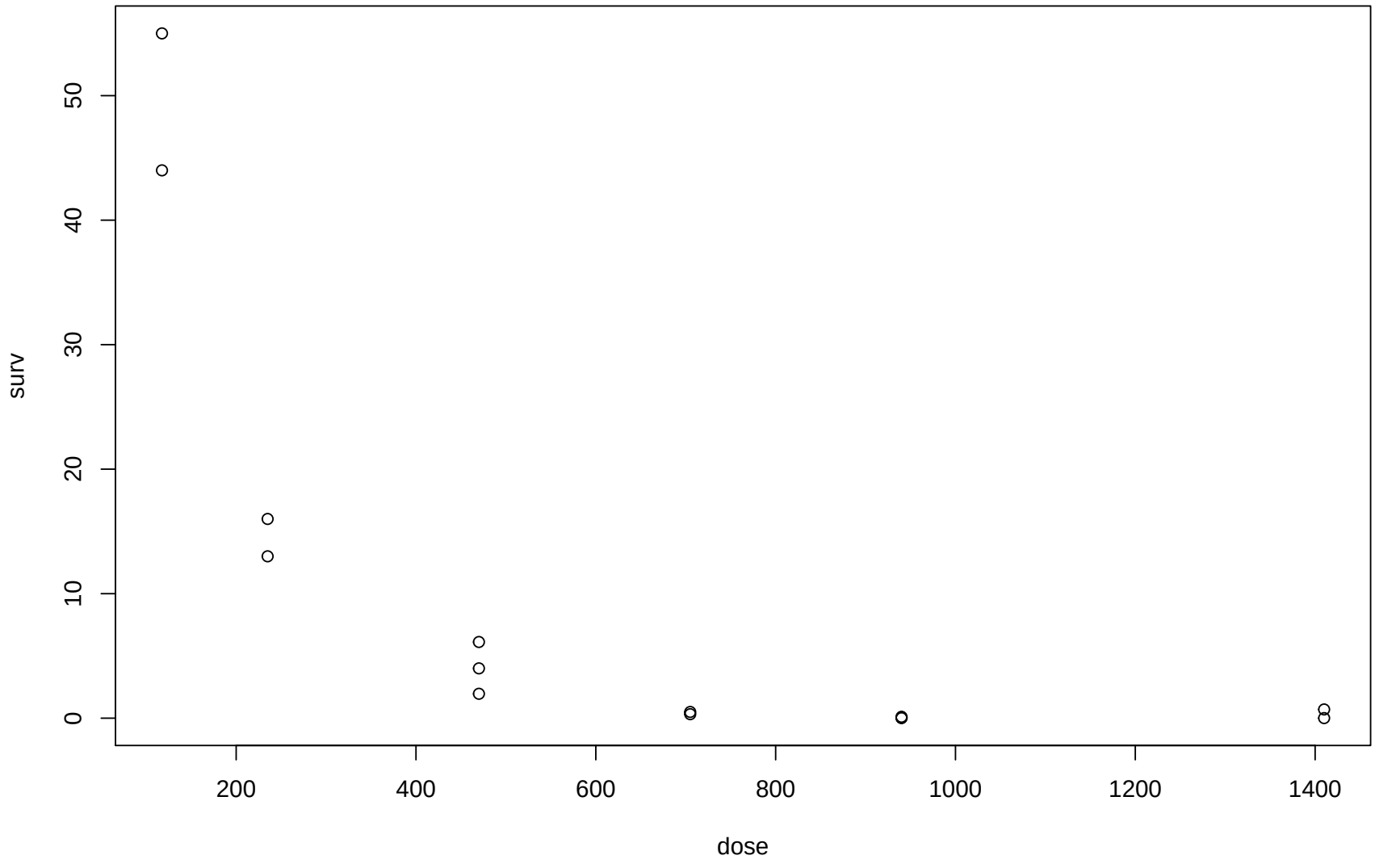
The opinions about the pro's and con's here differ, some claims are:

- asymptotically they are equivalent when the model is correctly specified.
- they perform quite differently for small samples and for the correct model the residual version is more efficient.
- the pairs strategy is often considered more robust when the model is misspecified.

For the heteroscedastic case there is a strategy called the wild bootstrap, not covered here.

Example

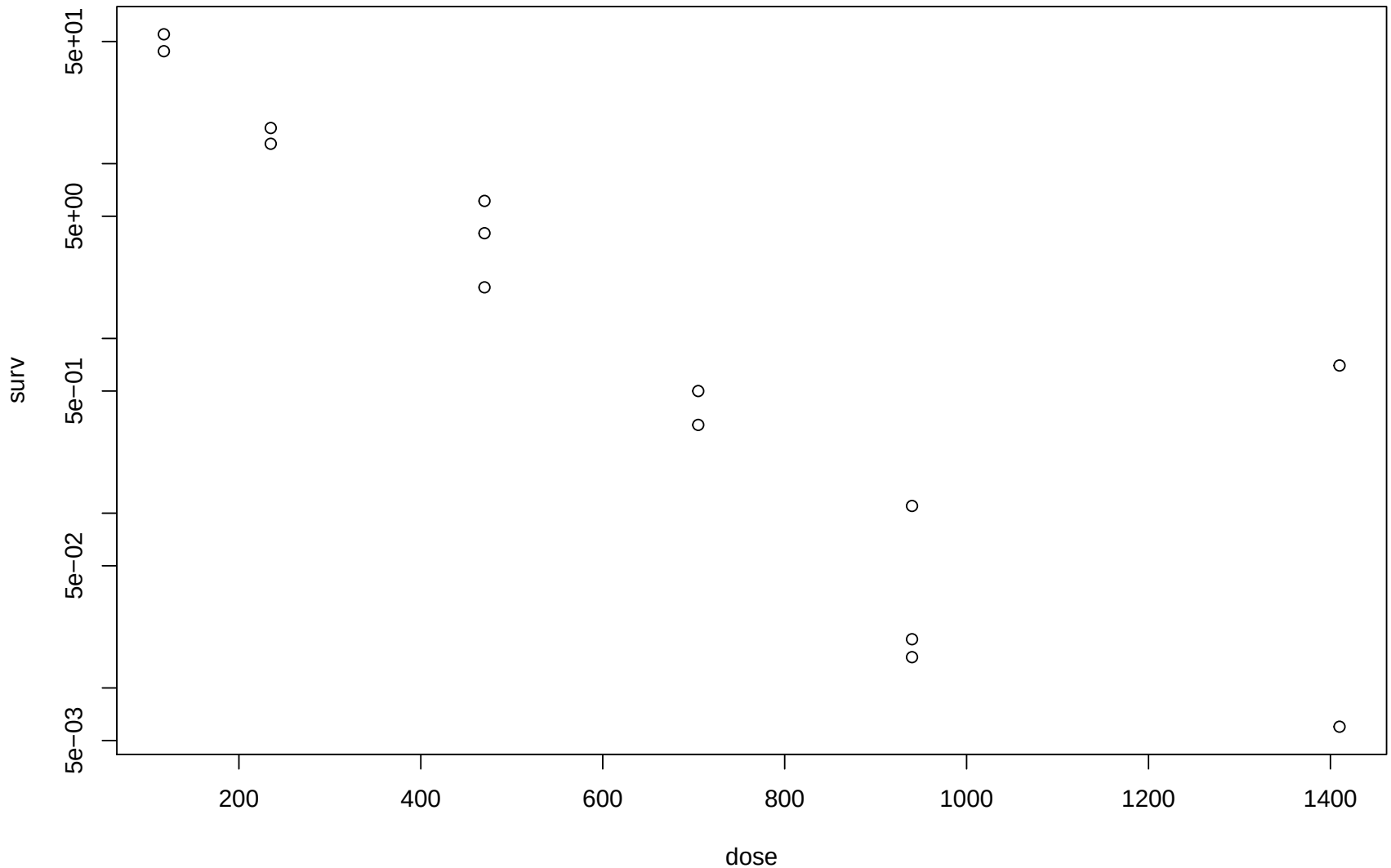
```
> library(boot); data(survival)
> with(survival, plot(dose, surv))
```



Example Log transformation

Watch out that for the whole example y data are log transformed.

```
> with(survival, plot(dose, surv, log="y"))
```

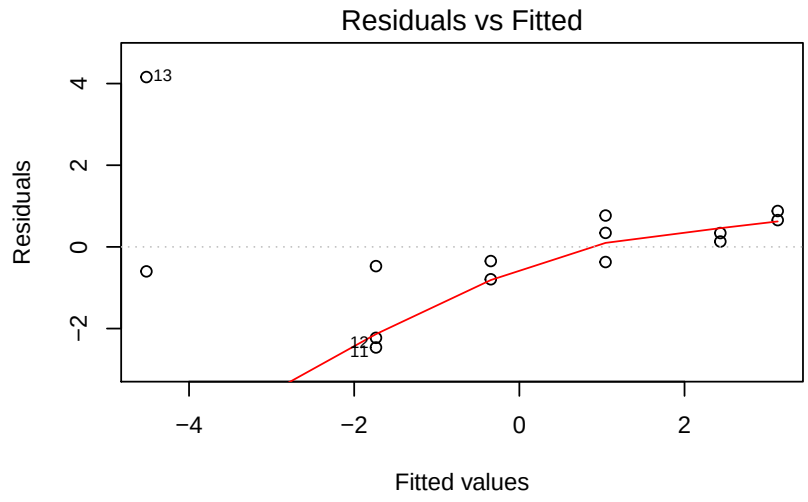


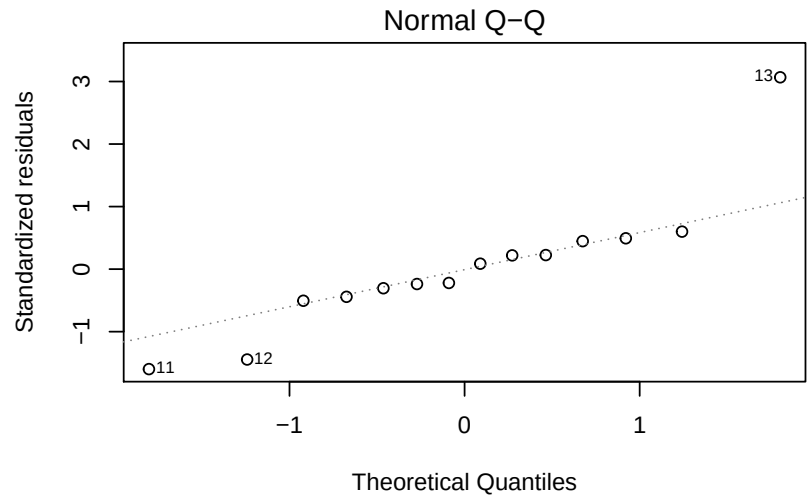
Example Linear Model Summary

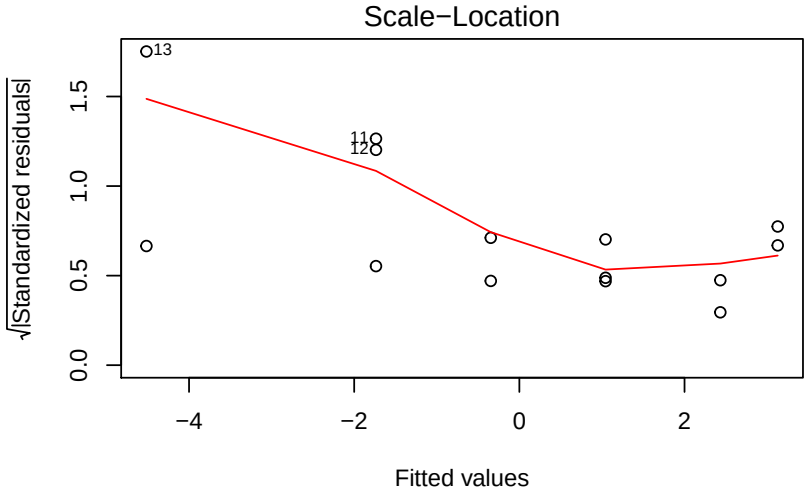
```
> fit <- lm(log(surv)~dose, data=survival)
> summary(fit)

Call:
lm(formula = log(surv) ~ dose, data = survival)

Residuals:
    Min      1Q  Median      3Q     Max
-2.4637 -0.5679 -0.1079  0.5772  4.1592

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.823648   0.811788   4.710 0.000505 ***
dose        -0.005915   0.001047  -5.651 0.000107 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.629 on 12 degrees of freedom
Multiple R-squared:  0.7269,    Adjusted R-squared:  0.7041
F-statistic: 31.93 on 1 and 12 DF,  p-value: 0.0001071
```

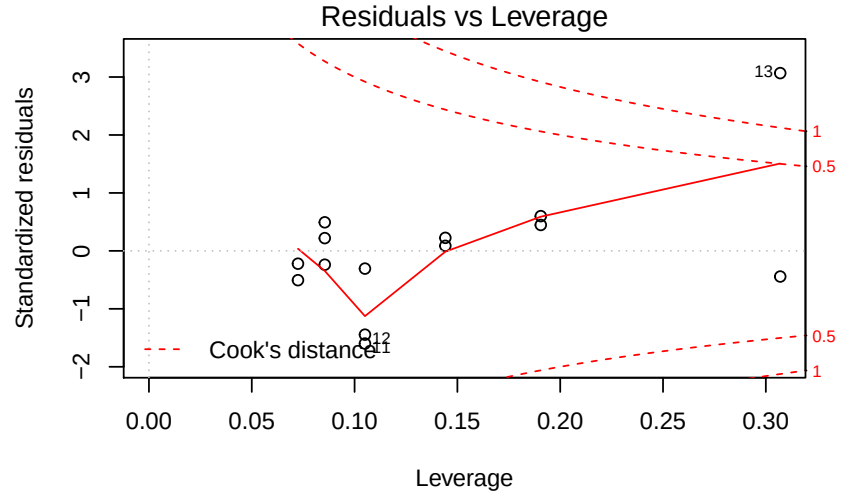
Example Residual Diagnostic Plots

```
> par(mfrow=c(2,2)); plot(fit); par(mfrow=c(1,1))
```

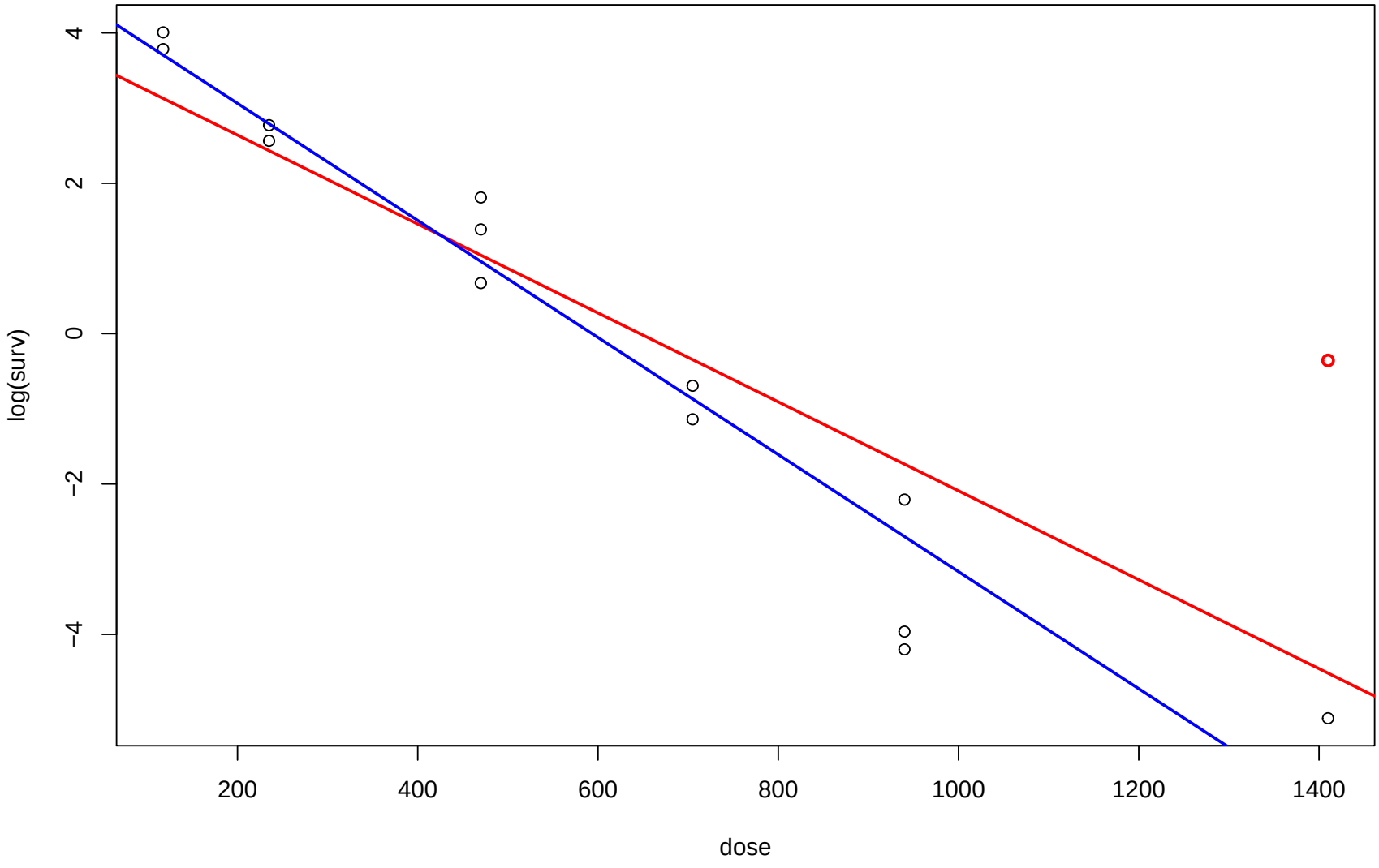








Example Leverage Point



Example Bootstrap Sample of Points

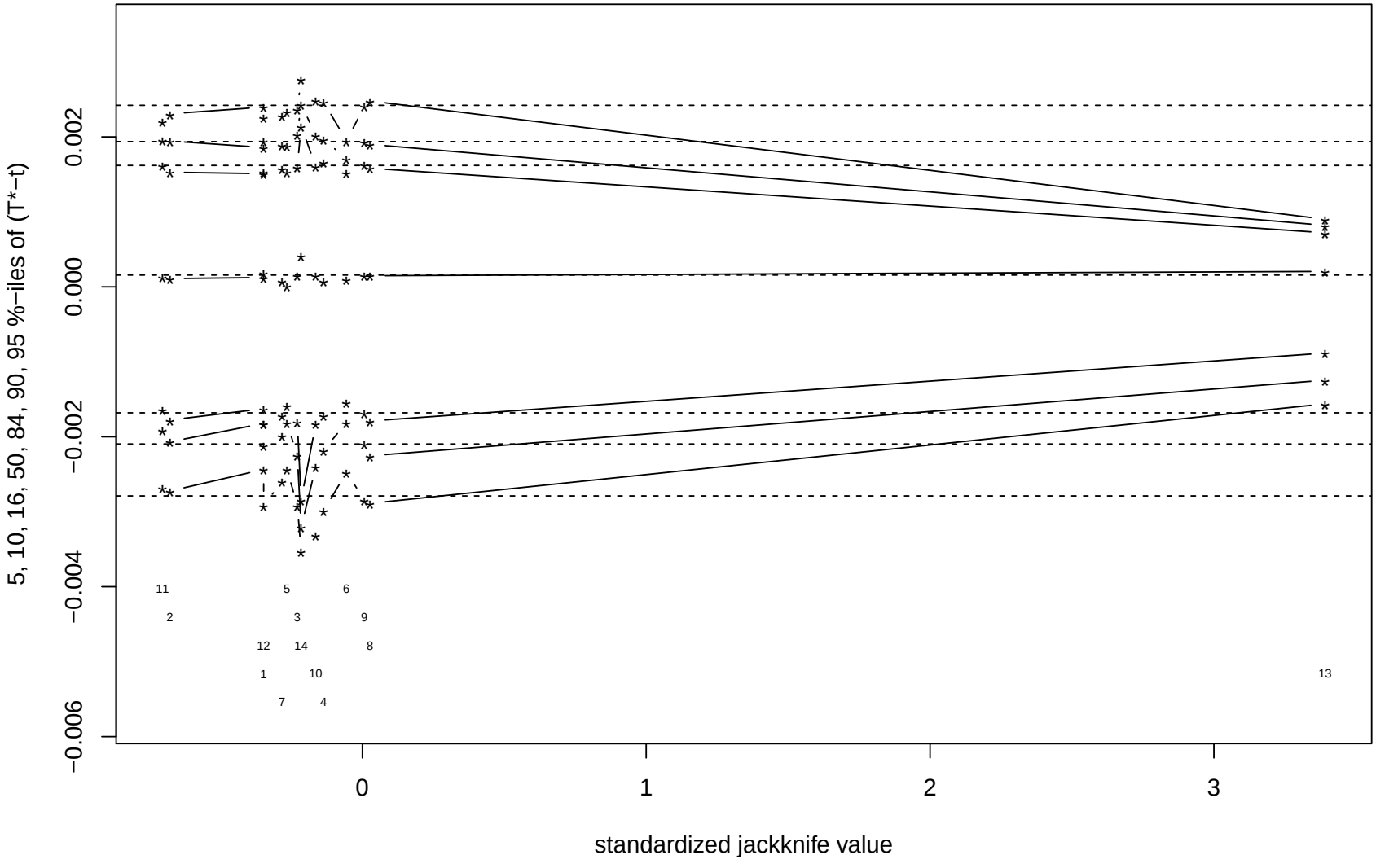
```
> set.seed(1)
> reg.fun <- function(x, i)
+ {
+   x.i <- x[i,]
+   x.i.reg <- lm(log(surv)~dose, data=x.i)
+   c(coef(x.i.reg))
+ }
> surv.boot <- boot(survival, reg.fun, R=1000)
```

Example Non-parametric Bootstrap of Model

```
> surv.boot

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = survival, statistic = reg.fun, R = 1000)


Bootstrap Statistics :
       original         bias    std. error
t1*  3.82364794  0.0435261916 0.756957587
t2* -0.00591454 -0.0001362111 0.001581196
```

Example Jackknife reveals Leverage Point

```
> jack.after.boot(surv.boot, index=2)
```



Example Parametric Bootstrap of the Residuals

```
> fit <- lm(log(surv)~dose, data=survival)
> res <- resid(fit)
> yhat <- fitted(fit)
> surv2 <- data.frame(yhat, res)
> surv2.fun <- function(x)
+ {coef(lm(log(surv)~dose, data=x))}
> surv2.sim <- function(x, resi)
+ {
+   x$surv <- exp(resi$yhat + sample(resi$res,
+                                    replace=TRUE))
+   x
+ }
> fit.boot <- boot(survival, surv2.fun, R=1000,
+                  sim="parametric", ran.gen=surv2.sim,
+                  mle = surv2)
```

Example Parametric Bootstrap of the Residuals

```
> fit.boot

PARAMETRIC BOOTSTRAP


Call:
boot(data = survival, statistic = surv2.fun, R = 1000, sim
    ran.gen = surv2.sim, mle = surv2)


Bootstrap Statistics :
       original         bias     std. error
t1*  3.82364794  2.023075e-02 0.7400203149
t2* -0.00591454 -3.630009e-05 0.0009686601
```