Transformation Models: The tram Package

Torsten Hothorn
Universität Zürich

Abstract

The **tram** package allows a range of stratified linear transformation models to be fitted using standard formula-based R interfaces. Functions for the estimation of potentially stratified Cox models, several survival regression models including log-normal or log-logistic models, models for ordered categorical responses (proportional odds, proportional hazards, probit) are available. Likelihood-based inference, also in the presense of random left-, right-, and interval-censoring and arbitrary forms of truncation, is performed using infrastructure from package **mlt**. More complex models, allowing non-linear and interaction functions of covariates, can be estimated using corresponding transformation trees and forests in package **trtf** with the same simple user interface.

Keywords: Linear model, Cox model, survival regression, ordered regression, censoring, truncation.

1. Introduction

The **tram** package offers standard formula-based R interfaces for stratified linear transformation models. The package uses general infrastructure for likelihood-based inference in conditional transformation models provided by package **mlt** (Hothorn 2018, 2025a). The underlying theory is presented by Hothorn *et al.* (2018) and an introduction to the **mlt** package is given by Hothorn (2018). An interface to package **trtf** (Hothorn 2025b) also allows more complex models to be fitted by recursive partitioning and random forest technology (Hothorn and Zeileis 2017).

In a nutshell, the model class covered by package **tram** consists of transformation models of the form

$$\mathbb{P}(Y \le y \mid \boldsymbol{S} = \boldsymbol{s}, \boldsymbol{X} = \boldsymbol{x}) = F_Z(h_Y(y \mid \boldsymbol{s}) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta} + \text{offset}) \tag{1}$$

where Y is an at least ordered univariate response variable, $\boldsymbol{S}$ are stratum variables and $\boldsymbol{X}$ covariates with observations $y, \boldsymbol{s}$, and $\boldsymbol{x}$, respectively. We use the term 'stratum' in a little more general sense and allow factors and also numeric variables in $\boldsymbol{S}$. The P-vector $\tilde{\boldsymbol{x}}^\top$ is a row of the design matrix corresponding to the observed covariate status $\boldsymbol{x}$. The package allows different choices of the 'link' function F_Z and the monotone increasing (in its y argument) 'baseline transformation' h_Y and, consequently, the estimation of a rather broad class of models.

Please cite this document as: Torsten Hothorn (2025) Transformation Models: The **tram** Package. R package vignette version 1.3-0, URL https://doi.org/10.32614/CRAN.package.tram.

A little more specifically, the monotone increasing baseline stratum-specific transformation h_Y is of the form

$$h_Y(y \mid \boldsymbol{s}) = \tilde{\boldsymbol{s}}^\top \boldsymbol{\alpha}(y)$$

with J-vector $\tilde{\boldsymbol{s}}^\top$ being the row of the design matrix corresponding to the observed strata $\boldsymbol{s}$. Each element of the parameter vector $\boldsymbol{\alpha}(y) = (\boldsymbol{a}(y)^\top \boldsymbol{\vartheta}_1, \dots, \boldsymbol{a}(y)^\top \boldsymbol{\vartheta}_J) \in \mathbb{R}^J$ is parameterised as $\boldsymbol{a}(y)^\top \boldsymbol{\vartheta}_j, j = 1, \dots, J$. Thus, the 'response-varying' coefficients of $\tilde{\boldsymbol{s}}^\top$ depend on the response y. The key part is a basis transformation $\boldsymbol{a}(y)$ of the response. Different choices of this basis allow the model class to handle different types of response variables and models of different complexity. In the absence of any stratum variables, we have

$$h_Y(y) = \boldsymbol{a}(y)^\top \boldsymbol{\vartheta}_1$$

and this function can be interpreted as an intercept function. With a single factor coding J strata, we obtain

$$h_Y(y \mid \boldsymbol{s} = j) = \boldsymbol{a}(y)^\top \boldsymbol{\vartheta}_j.$$

In this case, one intercept function is fitted for level j. We treat numeric variables with response-varying coefficients in a similar way. With $\boldsymbol{s} = s \in \mathbb{R}$, the baseline transformation

$$h_Y(y \mid \boldsymbol{s}) = \alpha_1(y) + s\alpha_2(y) = \boldsymbol{a}(y)^\top \boldsymbol{\vartheta}_1 + s\boldsymbol{a}(y)^\top \boldsymbol{\vartheta}_2$$

consists of an intercept function $\alpha_1(y)$ and a response-varying effect $\alpha_2(y)$ of s. The latter function is called 'time-varying' in survival analysis and some people use the more general term 'distribution regression' when refering to models with response-varying coefficients. Because the intercept function is contained in h_Y, the linear predictor $\tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}$ must not contain an intercept. The design row $\tilde{\boldsymbol{s}}^\top$, however, is expected to include an intercept term, explicitly or implicitly.

All model interfaces implemented in the **tram** package expect models to be specified by calls of the form

```
R> tram(y | s ~ x, ...)
```

where `y` is a variable containing the observed response (possibly under all forms random censoring and truncation), `s` specifies the design row $\tilde{\boldsymbol{s}}^\top$ and `x` the row $\tilde{\boldsymbol{x}}^\top$ in the design matrix using standard R formula language. Specific modelling functions for normal linear regression models (`Lm()`), non-normal linear regression models (`BoxCox()`, `Colr()`), ordinal linear regression models (`Polr()`), and survival regression models (`Survreg()`, `Coxph()`) implement transformation models tailored to these specific domains. The corresponding user interfaces resemble the interfaces of existing modeling functions (such as `lm()`, `polr()`, `survreg()`, or `coxph()`) are closely as possible.

This document describes the underlying models and illustrates the application of the **tram** package for the estimation of specific stratified linear transformation models.

2. Normal Linear Regression Models

The normal linear regression model

$$Y = \tilde{\alpha} + \tilde{\boldsymbol{x}}^\top \tilde{\boldsymbol{\beta}} + \varepsilon, \quad \varepsilon \sim \mathrm{N}(0, \sigma^2)$$

is a transformation model of the general form (1) because we can rewrite it in the form

$$\begin{aligned} \mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) &= \Phi\left(\frac{y - \tilde{\alpha} - \tilde{\boldsymbol{x}}^\top \tilde{\boldsymbol{\beta}}}{\sigma}\right) \\ &= \Phi(\vartheta_1 + \vartheta_2 y - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}). \end{aligned} \tag{2}$$

With $\vartheta_1 = -\tilde{\alpha}/\sigma, \vartheta_2 = 1/\sigma$ and $\boldsymbol{\beta} = \tilde{\boldsymbol{\beta}}/\sigma$ we see that the model is of the form

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = F_Z(h_Y(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$$

with distribution function $F_Z = \Phi$ and linear transformation $h_Y(y) = \vartheta_1 + \vartheta_2 y$ such that

$$\mathbb{E}(h_Y(Y) \mid \boldsymbol{X} = \boldsymbol{x}) = \mathbb{E}(\vartheta_1 + \vartheta_2 Y \mid \boldsymbol{X} = \boldsymbol{x}) = \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}.$$

The Boston Housing data are a prominent test-bed for parametric and non-parametric alternatives to a normal linear regression model. Assuming a conditional normal distribution for the median value of owner-occupied homes (medv, in USD 1000's, we use the corrected version) in the normal linear model with constant variance

$$\text{medv} \mid \boldsymbol{X} = \boldsymbol{x} \sim \text{N}(\tilde{\alpha} + \tilde{\boldsymbol{x}}^\top \tilde{\boldsymbol{\beta}}, \sigma^2)$$

we can fit this model using the `lm()` function:

```
R> data("BostonHousing2", package = "mlbench")
R> lm_BH <- lm(cmedv ~ crim + zn + indus + chas + nox + rm + age + dis +
+               rad + tax + ptratio + b + lstat, data = BostonHousing2)
```

The **tram** package implements a function `Lm()` for fitting the normal linear regression model in the parameterisation (2)

```
R> Lm_BH_1 <- Lm(cmedv ~ crim + zn + indus + chas + nox + rm + age + dis +
+                 rad + tax + ptratio + b + lstat, data = BostonHousing2)
```

The model fit is the same

```
R> logLik(lm_BH)

'log Lik.' -1494 (df=15)

R> logLik(Lm_BH_1)

'log Lik.' -1494 (df=15)
```

so why would one want to set `lm()` aside in favour of `Lm()`? The parameterisation in (2) seems a little odd, because the parameters are $\vartheta_1 = -\tilde{\alpha}/\sigma$ and $\boldsymbol{\beta} = \tilde{\boldsymbol{\beta}}/\sigma$ instead of the mean $\tilde{\alpha}$ and the regression coefficients $\tilde{\boldsymbol{\beta}}$. The parameters on the scale of $\tilde{\boldsymbol{\beta}}$, including the intercept $\tilde{\alpha}$ can be obtained via

```
R> coef(lm_BH)

(Intercept)        crim          zn       indus       chas1         nox
  3.637e+01  -1.062e-01   4.772e-02   2.325e-02   2.692e+00  -1.774e+01
         rm         age         dis         rad         tax     ptratio
  3.789e+00   5.749e-04  -1.502e+00   3.038e-01  -1.270e-02  -9.239e-01
          b       lstat
  9.228e-03  -5.307e-01

R> coef(Lm_BH_1, as.lm = TRUE)

(Intercept)        crim          zn       indus       chas1         nox
  3.637e+01  -1.062e-01   4.772e-02   2.324e-02   2.692e+00  -1.774e+01
         rm         age         dis         rad         tax     ptratio
  3.790e+00   5.731e-04  -1.502e+00   3.037e-01  -1.270e-02  -9.239e-01
          b       lstat
  9.229e-03  -5.307e-01
attr(,"scale")
cmedv
4.637
```

The standard deviation is the inverse interaction term with the response ϑ_2^{-1}

```
R> summary(lm_BH)$sigma

[1] 4.703

R> 1 / coef(Lm_BH_1, with_baseline = TRUE)["cmedv"]

cmedv
4.637
```

The latter estimate is the maximum-likelihood estimator of $\hat{\sigma}$ and not the usual REML estimate reported by `lm()`.

One subtle difficulty with the observed response is that median housing values larger than 50 are actually right-censored. This fact was ignored in both model fits. In contrast to `lm()`, `Lm()` is able to deal with this situation

```
R> BostonHousing2$y <- with(BostonHousing2, Surv(cmedv, cmedv < 50))
R> Lm_BH_2 <- Lm(y ~ crim + zn + indus + chas + nox +
+                 rm + age + dis + rad + tax + ptratio + b + lstat,
+                 data = BostonHousing2)
R> logLik(Lm_BH_2)

'log Lik.' -1496 (df=15)
```

Why is the extention to censored responses such a big deal? `lm` estimates the regression coefficients $\tilde{\boldsymbol{\beta}}$ by least-squares $(y - \tilde{\boldsymbol{x}}^\top \tilde{\boldsymbol{\beta}})^2$. In the presence of censoring (here at `cmedv` > 50), the likelihood contribution is

$$\mathbb{P}(\text{medv} > 50 \mid \boldsymbol{X} = \boldsymbol{x}) = 1 - F_Z(\vartheta_1 + \vartheta_2 50 - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$$

and one cannot treat the inverse standard deviation $\sigma = \vartheta_2^{-1}$ as a nuisance parameter. Thus, the variance needs to be estimated simultaneously with the shift parameters. As we will see later, this model can also be estimated by `survreg()`, for example.

One of the variables (`chas`) gives us information about proximity to the Charles River. The model above only allows for mean changes in this variable, but how about models that allow different variances? This is simple in `Lm()` because we can use `chas` as a stratum variable. This means that we estimate ϑ_1 and ϑ_2 separately for each of the two groups in the model

```
R> Lm_BH_3 <- Lm(y | 0 + chas ~ crim + zn + indus + nox +
+                rm + age + dis + rad + tax + ptratio + b + lstat,
+                data = BostonHousing2)
R> logLik(Lm_BH_3)
```

```
'log Lik.' -1478 (df=16)
```

Here, it seems the standard deviation is almost twice as large in areas without access to the Charles River. Because the stratum-specific inverse standard deviations are parameters in our model,

```
R> 1 / coef(Lm_BH_3, with_baseline = TRUE)[c(2, 4)]
```

```
y:chas0 y:chas1 
  4.133   7.157 
```

we can construct a test for the null of equal variances as a test for a linear hypothesis

```
R> summary(glht(as.mlt(Lm_BH_3), linfct = c("y:chas0 - y:chas1 = 0")))
```

```

	 Simultaneous Tests for General Linear Hypotheses

Fit: Lm(formula = y | 0 + chas ~ crim + zn + indus + nox + rm + age + 
    dis + rad + tax + ptratio + b + lstat, data = BostonHousing2)

Linear Hypotheses:
                       Estimate Std. Error z value Pr(>|z|)    
y:chas0 - y:chas1 == 0    0.102      0.015    6.83  8.5e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)
```

We could go one step further and include an interaction term between each covariate and the response by treating all covariates as strata

```
R> Lm_BH_4 <- Lm(y | 0 + chas + crim + zn + indus + nox +
+                  rm + age + dis + rad + tax + ptratio + b + lstat ~ 0,
+                  data = BostonHousing2)
R> logLik(Lm_BH_4)

'log Lik.' -1310 (df=28)
```

For each variable x (or factor level) in the model, we obtain two parameters, *i.e.* $\vartheta_1 x + \vartheta_2 yx = (\vartheta_1 + \vartheta_2 y)x$. Thus, each regression coefficient for x may vary with the response in a linear way. Technically, this extention of the notion of a stratum to numeric covariates leads to the simplest form of 'distribution regression', that is, a model with linear response-varying regression coefficients. Because our transformation function is still linear in y, the class of conditional distributions of our response $Y \mid \boldsymbol{X} = \boldsymbol{x}$ is still normal. The next section discusses how we can fit models without such an restrictive assumption.

3. Box-Cox Models

Maybe the question causing most headaches in normal linear regression is 'Is my response conditionally normal?'. In their seminal paper, Box and Cox (1964) suggested to transform the response to normality. They used a rather simple function, today known as the Box-Cox transformation, whereas the **tram** package uses rather flexible Bernstein polynomials for the 'baseline transformation' h_Y. Although the technical details are different, the spirit is the same and we thus refer to the model

$$\mathbb{P}(Y \le y \mid \boldsymbol{S} = \boldsymbol{s}, \boldsymbol{X} = \boldsymbol{x}) = \Phi(h_Y(y \mid \boldsymbol{s}) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta} + \text{offset})$$

with smooth (with respect to y) baseline transformation h_Y as a 'Box-Cox' model. For the Boston Housing data, we first fit the model

$$\mathbb{P}(\text{medv} \le y \mid \boldsymbol{X} = \boldsymbol{x}) = \Phi(h_Y(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$$

```
R> BC_BH_1 <- BoxCox(y ~ chas + crim + zn + indus + nox +
+                      rm + age + dis + rad + tax + ptratio + b + lstat,
+                      data = BostonHousing2)
R> logLik(BC_BH_1)

'log Lik.' -1320 (df=20)
```

Inspection of the fitted baseline transformation $\hat{h}_Y(y)$ is a simple device for detecting deviations from normality. An approximately linear function $\hat{h}_Y(y)$ suggests that the normal assumption is appropriate whereas deviations from linearity correspond to deviations from this assumption. Figure 1 indicates that there is a deviation from normality in the upper tail.

When we add proximity to Charles River as stratum in the model

```
R> nd <- model.frame(BC_BH_1)[1,-1,drop = FALSE]
R> plot(BC_BH_1, which = "baseline only", newdata = nd, col = col,
+       confidence = "interval", fill = fill, lwd = 2,
+       xlab = "Median Value", ylab = expression(h[Y]))
```

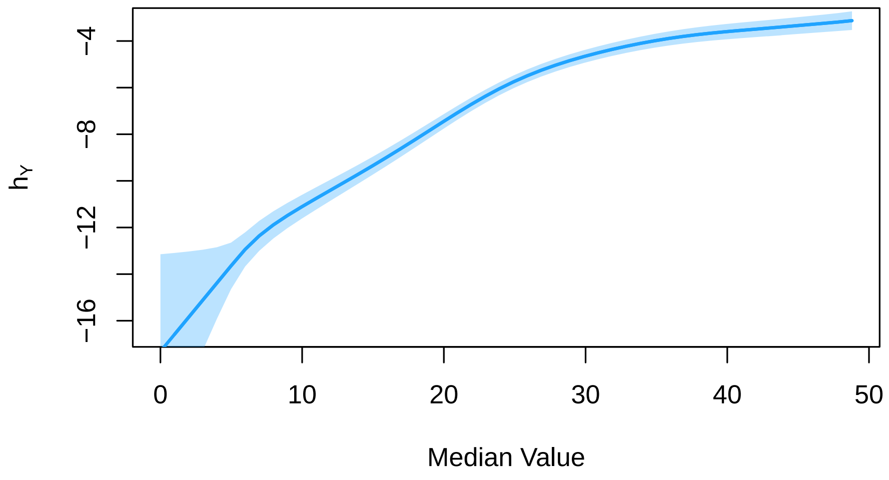

Figure 1: Boston Housing: Baseline transformation h_Y in model `BC_BH_1` with pointwise confidence intervals.

```
R> BC_BH_2 <- BoxCox(y | 0 + chas ~ crim + zn + indus + nox +
+                    rm + age + dis + rad + tax + ptratio + b + lstat,
+                    data = BostonHousing2)
R> logLik(BC_BH_2)

'log Lik.' -1316 (df=26)
```

We see, in Figure 2, that the two baseline transformations differ only for median values smaller than USD $15,000$ with a large variance in houses near Charles River (indicating that there are hardly any cheap houses in this area).

The heterogeneous variances in our model `Lm_BH_3` where caused by the same effect and Figure 3 shows that allowing the model to deviate from normality leads to better model interpretation.

We could now look at a model with smooth response-varying effects

```
R> BoxCox(y | 0 + chas + crim + zn + indus + nox +
+         rm + age + dis + rad + tax + ptratio + b + lstat ~ 0,
+         data = BostonHousing2)
```

but save this exercise for Section 5.

```
R> nd <- model.frame(BC_BH_2)[1:2, -1]
R> nd$chas <- factor(c("0", "1"))
R> plot(BC_BH_2, which = "baseline only", newdata = nd, col = col,
+      confidence = "interval", fill = fill, lwd = 2,
+      xlab = "Median Value", ylab = expression(h[Y]))
R> legend("bottomright", lty = 1, col = col,
+         title = "Near Charles River", legend = c("no", "yes"), bty = "n")
```

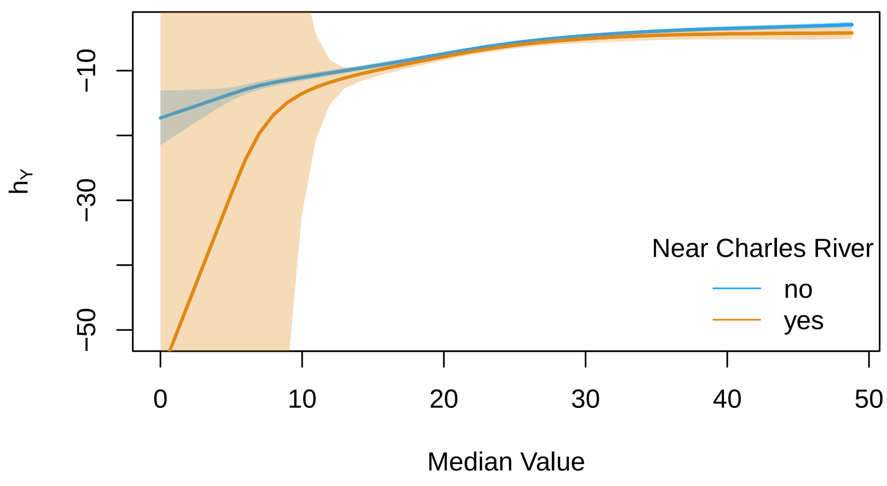

Figure 2: Boston Housing: Stratified baseline transformation h_Y in model `BC_BH_2` with pointwise confidence intervals.

4. Continuous Outcome Logistic Regression

One problem with the above discussed Box-Cox-type linear regression models is that we loose the nice interpretation of the regression coefficients $\tilde{\boldsymbol{\beta}}$ we all enjoy in normal linear regression models. The only way of interpreting the linear predictor in the model

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = \Phi(h_Y(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$$

with smooth baseline transformation h_Y is based on the relationship $\mathbb{E}(h_Y(Y) \mid \boldsymbol{X} = \boldsymbol{x}) = \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}$. That means that the conditional expectation of the *transformed* response Y is given by the linear predictor. We can still judge whether a relationship is positive or negative by the sign of the regression coefficients $\boldsymbol{\beta}$ but the nice way of saying 'a one-unit increase in x corresponds to an increase of β in the conditional mean of Y (all other covariables being fix)' just goes away. Better interpretability guides the choice of models in this section.

Consider the linear transformation model (we skip strata and offset here, for simplicity)

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = F_{\text{SL}}(h_Y(y) + \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})) = \frac{\exp(h_Y(y) + \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})}{1 + \exp(h_Y(y) + \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})}$$

```
R> plot(Lm_BH_3, which = "baseline only", newdata = nd, col = col,
+       confidence = "interval", fill = fill, lwd = 2)
R> legend("bottomright", lty = 1, col = col,
+         title = "Near Charles River", legend = c("no", "yes"), bty = "n")
```

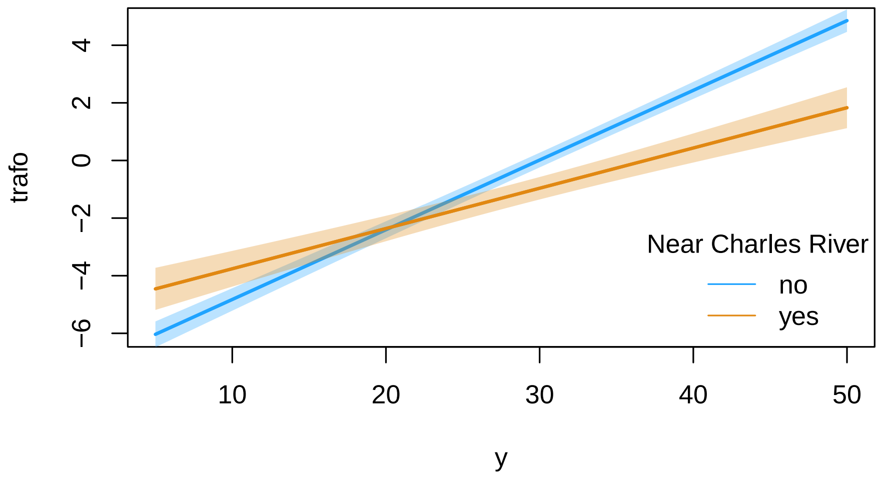

Figure 3: Boston Housing: Stratified linear baseline transformation h_Y in model `Lm_BH_2` with pointwise confidence intervals.

where we replace the cumulative distribution function of the standard normal Φ with the cumulative distribution function of the standard logistic. Note that we add the linear predictor instead of substracting it (as in the Cox model, see Section 6). In this model, the conditional odds is given by

$$\frac{\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x})}{\mathbb{P}(Y > y \mid \boldsymbol{X} = \boldsymbol{x})} = \exp(h_Y(y)) \exp(\tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}).$$

The odds ratio between a subject with covariate status $\boldsymbol{x}$ and a subject with covariates such that $\tilde{\boldsymbol{x}}^\top \boldsymbol{\beta} = 0$ is simply $\exp(\tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$ (and not $\exp(-\tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$, therefore a positive shift makes sense in this model). The response-varying baseline transformations conveniently cancel out in this odds ratio. We can interpret β as log-odds ratios, simultaneously for all possible binary logistic regression models with cutpoint y. The baseline transformation $h_Y(y)$ is the intercept in the logistic regression model corresponding to cutpoint y. Lohse *et al.* (2017) introduced the name ‘Continuous Outcome Logistic Regression’ (COLR) for this formulation of the model.

Using proximity to Charles River as single stratum variable $\boldsymbol{s}$, we can fit this model with smooth baseline transformation $h_Y(y \mid \boldsymbol{s})$

```
R> Colr_BH_1 <- Colr(y | 0 + chas ~ crim + zn + indus + nox +
+                      rm + age + dis + rad + tax + ptratio + b + lstat,
+                     data = BostonHousing2)
R> logLik(Colr_BH_1)

'log Lik.' -1291 (df=26)
```

In comparison to the BoxCox-type model `BC_BH_2` with the exact same structure of the model (except the link function) with likelihood -1315.97, the continuous outcome logistic regression improved the model fit.

The estimated odds-ratios, along with likelihood-based confidence intervals, can be computed as

```
R> round(cbind(exp(coef(Colr_BH_1)), exp(confint(Colr_BH_1))), 3)

                  2.5 %    97.5 %
crim       1.084  1.055     1.115
zn         0.990  0.980     0.999
indus      0.976  0.936     1.017
nox     1162.327 63.090 21414.046
rm         0.218  0.142     0.334
age        1.010  1.000     1.020
dis        1.660  1.425     1.935
rad        0.884  0.842     0.928
tax        1.007  1.004     1.009
ptratio    1.495  1.360     1.643
b          0.994  0.992     0.996
lstat      1.322  1.256     1.393
```

A display of the model and observed median housing values is given in Figure 4. For each observation, the linear predictor is plotted against the observed value and this scatterplot is overlayed with the conditional decile curves.

5. Connection to Quantile Regression

There is a strong relationship between quantile regression models and transformation models. In essence, quantile regression (as the name suggests) models the conditional quantile function whereas transformation models describe the conditional distribution function (that is, the inverse conditional quantile function). Under which circumstances can we expect to obtain similar results?

To answer this question, consider a linear quantile regression model for the pth conditional quantile $Q_Y(p \mid \boldsymbol{X} = \boldsymbol{x})$ of Y:

$$Q_Y(p \mid \boldsymbol{X} = \boldsymbol{x}) = \tilde{\alpha}(p) + \tilde{\boldsymbol{x}}^\top \tilde{\boldsymbol{\beta}}(p).$$

Typically (for example in package **quantreg**), separate models are fitted for varying probabilities $p \in (0, 1)$, potentially causing a 'quantile crossing' problem.

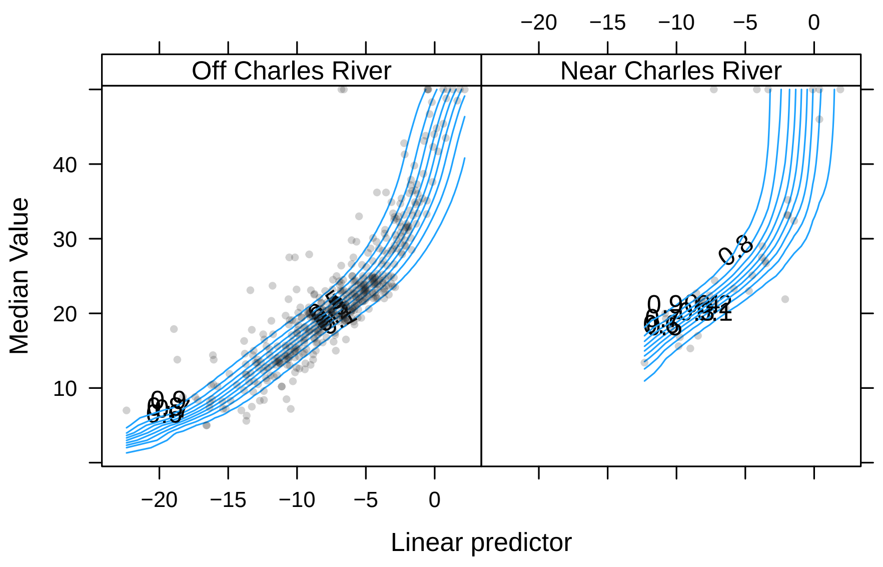

Figure 4: Boston Housing: Scatterplot of linear predictors vs. observed median values obtained from the Continuous Outcome Logistic Regression model `Colr_BH_1`. The observations are overlayed with lines representing conditional quantiles of median value given the linear predictor.

One nice feature of this model is its invariance wrt. monotone transformations, say h_Y, because we have

$$h_Y(Q_Y(p \mid \boldsymbol{X} = \boldsymbol{x})) = Q_{h_Y(Y)}(p \mid \boldsymbol{X} = \boldsymbol{x}).$$

With $y = Q_Y(p \mid \boldsymbol{X} = \boldsymbol{x})$ and a linear transformation model we obtain the relationship

$$\begin{aligned} & p = \mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = F_Z(h_Y(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}) \\ \iff \quad & h_Y(Q_Y(p \mid \boldsymbol{X} = \boldsymbol{x})) = F_Z^{-1}(p) + \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}. \end{aligned}$$

This means that a linear transformation model is a model for a transformed quantile with constant regression effects $\boldsymbol{\beta}$ (only the intercept depends on p). When we allow for response-varying effects $\boldsymbol{\beta}(y)$ in our transformation model, the relationship

$$\begin{aligned} & p = \mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = F_Z(h_Y(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}(y)) \\ \iff \quad & h_Y(Q_Y(p \mid \boldsymbol{X} = \boldsymbol{x})) = F_Z^{-1}(p) + \tilde{\boldsymbol{x}}^\top \tilde{\boldsymbol{\beta}}(p) \end{aligned}$$

with $\tilde{\boldsymbol{\beta}}(p) = \boldsymbol{\beta}(Q_Y(p \mid \boldsymbol{X} = \boldsymbol{x}))$ shows that a transformation model with response-varying effects (aka 'distribution regression') is a linear quantile regression models for a transformed quantile. In this sense, quantile regression and transformation models assume additivity of the effects on a different scale, but are otherwise comparable (also in terms of model complexity). A big advantage of transformation models is that we can estimate one model for all effects simultaneously and tedious post-processing to re-order crossing quantiles is not necessary.

But how well do the two approaches coincide for the Boston Housing data? We first fit a number of linear quantile regressions for a grid of probabilities $p \in \{.1, \dots, .9\}$ and, second, estimate a continuous outcome logistic regression with response-varying effects (restricting ourselves to very smooth regression coefficient functions parameterised as Bernstein polynomials with three parameters by the choice `order = 2`):

```
R> tau <- 2:18 / 20
R> fm <- cmedv ~ crim + zn + indus + chas + nox + rm + age + dis +
+                rad + tax + ptratio + b + lstat
R> rq_BH_1 <- lapply(tau, function(p) rq(fm, data = BostonHousing2, tau = p))
R> Colr_BH_2 <- Colr(cmedv | crim + zn + indus + chas + nox + rm + age + dis +
+                    rad + tax + ptratio + b + lstat ~ 0,
+                    data = BostonHousing2, order = 2)
```

For nine selected observations from the dataset, we then plot the estimated conditional distribution function obtained from both approaches. Figure 5 shows good agreement between the two models. Overall, the transformation models gives smoother conditional distribution functions and for some observations, for example for number 153, the estimated conditional quantiles are very rough and, in fact, not monotone.

6. Survival Analysis

Yet another choice of F_Z is motivated by the desire to obtain simple model interpretation. In survival analysis, we are interested in the conditional survivor function (omitting $\boldsymbol{s}$ and offset again)

$$S(y \mid \boldsymbol{X} = \boldsymbol{x}) = 1 - \mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = \exp(-\Lambda(y \mid \boldsymbol{X} = \boldsymbol{x}))$$

with conditional cumulative hazard function $\Lambda(y \mid \boldsymbol{X} = \boldsymbol{x})$. Because Λ is always positive, one can parameterise Λ via the log-cumulative hazard function

$$\Lambda(y \mid \boldsymbol{X} = \boldsymbol{x}) = \exp(h_Y(y) - \tilde{\boldsymbol{x}}^\top\boldsymbol{\beta}) = \exp(h_Y(y))\exp(-\tilde{\boldsymbol{x}}^\top\boldsymbol{\beta})$$

Here, $\exp(-\tilde{\boldsymbol{x}}^\top\boldsymbol{\beta})$ is the hazard ratio of the cumulative hazard between a subject with covariate status $\boldsymbol{x}$ and a subject with $\tilde{\boldsymbol{x}}^\top\boldsymbol{\beta} = 0$. The hazard function is

$$\lambda(y \mid \boldsymbol{X} = \boldsymbol{x}) = \frac{\partial\Lambda(y \mid \boldsymbol{X} = \boldsymbol{x})}{\partial y} = \exp(h_Y(y))\frac{\partial h_Y(y)}{\partial y}\exp(-\tilde{\boldsymbol{x}}^\top\boldsymbol{\beta}) = \lambda(y)\exp(-\tilde{\boldsymbol{x}}^\top\boldsymbol{\beta})$$

with 'baseline hazard function' λ. The ratio of two hazard functions is again $\exp(-\tilde{\boldsymbol{x}}^\top\boldsymbol{\beta})$. Putting everything together we see that we formulate the conditional distribution function as

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = 1 - \exp(-\exp(h_Y(y) - \tilde{\boldsymbol{x}}^\top\boldsymbol{\beta})$$

which corresponds to our linear transformation model (1) with $F_Z() = 1 - \exp(-\exp())$ being the cumulative distribution function of the standard minimum extreme value distribution.

Like in the normal case, special choices of h_Y correspond to simple parametric models. The Weibull linear regression model features a linear baseline transformation h_Y of $\log(y)$

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = 1 - \exp(-\exp(\vartheta_1 + \vartheta_2\log(y) - \tilde{\boldsymbol{x}}^\top\boldsymbol{\beta})), \vartheta_2 > 0$$

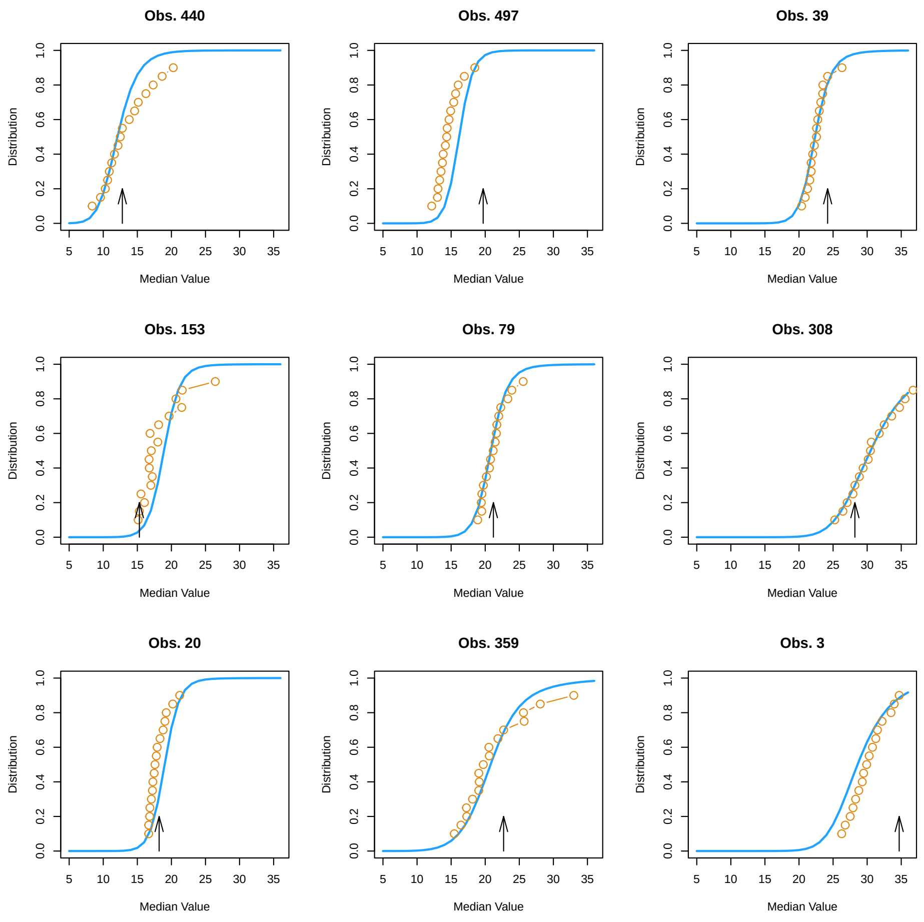

Figure 5: Boston Housing: Comparison of distribution regression via a transformation model (model `Colr_BH_2`, solid lines) and quantile regression (model `rq_BH_1`, dots). The plot shows the estimated conditional distribution functions along with the actually observed median value (arrows).

which, with $\vartheta_2 = 1$, simplifies to an exponential distribution

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = 1 - \exp(-y \exp(\vartheta_1 - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}))$$

with parameter $\lambda = \exp(\vartheta_1 - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$. With $\vartheta = 2$, the distribution is known as Rayleigh distribution.

`survreg()` (Therneau 2024) uses a different parameterisation of the same model with scale parameter σ of the form

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = 1 - \exp\left(-\exp\left(\frac{\tilde{\alpha} + \log(y) - \tilde{\boldsymbol{x}}^\top \tilde{\boldsymbol{\beta}}}{\sigma}\right)\right).$$

As an example, consider time-to-death data from the German Breast Cancer Study Group 2 randomised clinical trial on hormonal treatment of breast cancer (`horTh` being the treatment indicator). The Weibull model can be fitted by these two functions

```
R> data("GBSG2", package = "TH.data")
R> Survreg_GBSG2_1 <- Survreg(Surv(time, cens) ~ horTh, data = GBSG2)
R> logLik(Survreg_GBSG2_1)

'log Lik.' -2632 (df=3)

R> survreg_GBSG2_1 <- survreg(Surv(time, cens) ~ horTh, data = GBSG2)
R> logLik(survreg_GBSG2_1)

'log Lik.' -2632 (df=3)
```

The results are equivalent, but the parameters differ. To obtain the log-hazard ratio, we need to scale the estimate from `survreg()`

```
R> c(coef(Survreg_GBSG2_1),
+    coef(survreg_GBSG2_1)["horThyes"] / survreg_GBSG2_1$scale)

horThyes horThyes
  0.3932   0.3932
```

We get the log-hazard ratio $\beta = 0.39$ as the log-hazard ratio between treated and untreated patients. The hazard ratio $\exp(-\beta) = 0.67$ means that the hazard of a treated patient is 67% the hazard of an untreated patient. The corresponding confidence interval is

```
R> exp(-rev(confint(Survreg_GBSG2_1)))

[1] 0.5284 0.8619
```

The assumption of 'proportional hazards', *i.e.* the belief that the conditional hazard functions given treatment differ by a time-constant parameter β, might be questionable. We might want to ask 'How well do the corresponding conditional survivor functions (Figure 6, obtained under proportional hazards) reflect what is going on in the data?'.

The question is simple to answer when we use the treatment as a stratum and, in essence, fit two unconditional models (for each treatment arm) at once

```
R> Survreg_GBSG2_2 <- Survreg(Surv(time, cens) | 0 + horTh ~ 1, data = GBSG2)
R> logLik(Survreg_GBSG2_2)

'log Lik.' -2632 (df=4)

R> survreg_GBSG2_2 <- survreg(Surv(time, cens) ~ strata(horTh) + horTh - 1,
+                              data = GBSG2)
R> logLik(survreg_GBSG2_2)

'log Lik.' -2632 (df=4)
```

```
R> nd <- data.frame(horTh = factor(c("no", "yes")))
R> plot(Survreg_GBSG2_1, newdata = nd, which = "distribution",
+       type = "survivor", confidence = "interval", fill = fill,
+       col = col, ylab = "Probability", xlab = "Survival Time")
R> legend("bottomleft", lty = 1, title = "Hormonal Therapy",
+         legend = levels(nd$horTh), bty = "n", col = col)
```

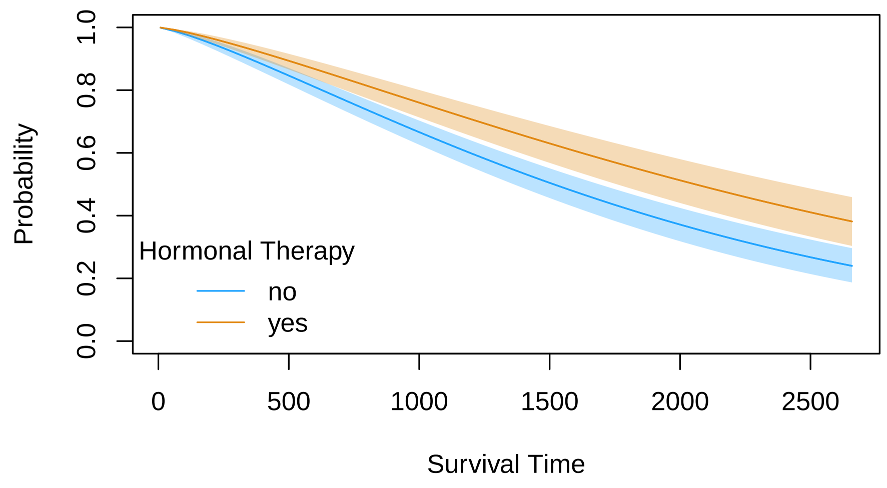

Figure 6: German Breast Cancer Study Group 2: Conditional survivor curves of treated and control patients obtained from the Weibull model `Survreg_GBSG2_1` under proportional hazards assumption.

```
R> coef(Survreg_GBSG2_2, with_baseline = TRUE)

           (Intercept):horThno  log(Surv(time, cens)):horThno
                        -9.736                          1.279
          (Intercept):horThyes log(Surv(time, cens)):horThyes
                       -10.274                          1.299

R> c(1 / survreg_GBSG2_2$scale, -coef(survreg_GBSG2_2) /
+                                 survreg_GBSG2_2$scale)

      no      yes  horThno horThyes
   1.279    1.299   -9.736  -10.274
```

It should be noted that the `strata` command in `survreg()` only leads to stratum-specific scale parameters, so we added a treatment specific intercept to make the two models comparable. The log-likelihood is roughly identical to the log-likelihood of the proportional hazards model and we can treat the assumption as reasonable.

As in the normal linear model, we might want to allow deviations from the strong distributional assumptions inherit in a Weibull model. In our framework, this simply means we allow for more flexible baseline log-hazard functions h_Y. The model featuring a smooth function h_Y

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = 1 - \exp(-\exp(h_Y(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}))$$

with $\Lambda(y \mid \boldsymbol{X} = \boldsymbol{x}) = \exp(h_Y(y)) \exp(-\tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$ and thus baseline cumulative hazard $\Lambda_Y(y) = \exp(h_Y(y))$ is called 'Cox proportional hazards model'. The term $\exp(-\tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$ is the hazard ratio. Typically (for example in `coxph()` and our `Coxph()`), the model is parameterised as

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = 1 - \exp(-\exp(h_Y(y) + \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}))$$

with positive linear predictors being related to larger hazards (and smaller means). Thus, the hazard ratio is $\exp(\tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$. The classical (and not so classical) literature very often associates the Cox model with an unspecified baseline hazard function and the corresponding partial likelihood estimator for $\boldsymbol{\beta}$. The **tram** package uses a parametric yet very flexible form for h_Y, leading to simpler inference and smooth estimated conditional survivor curves.

For the breast cancer data, we get with

```
R> Coxph_GBSG2_1 <- Coxph(Surv(time, cens) ~ horTh, data = GBSG2)
R> logLik(Coxph_GBSG2_1)

'log Lik.' -2606 (df=8)

R> coef(Coxph_GBSG2_1)

horThyes
 -0.3657
```

a model very similar to `Survreg_GBSG2_1` and stratification with respect to treatment does not improve the fit by much

```
R> Coxph_GBSG2_2 <- Coxph(Surv(time, cens) | 0 + horTh ~ 1 , data = GBSG2)
R> logLik(Coxph_GBSG2_2)

'log Lik.' -2604 (df=14)
```

When we compare the survivor curves from these two models with the Kaplan-Meier curves in Figure 7 we essentially get a smooth interpolation of this nonparametric maximum likelihood estimator by the two parametric Cox models.

Survival models with conditional log-logistic distribution are transformation models with $F_Z = \text{expit}$ and linear baseline transformation of a log-transformed response:

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = \frac{\exp(\vartheta_1 + \vartheta_2 \log(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})}{1 + \exp(\vartheta_1 + \vartheta_2 \log(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})}, \vartheta_2 > 0$$

```
R> plot(survfit(Surv(time, cens) ~ horTh, data = GBSG2), col = col,
+       ylab = "Probability", xlab = "Survival Time")
R> plot(Coxph_GBSG2_1, newdata = nd, which = "distribution",
+       type = "survivor", col = col, add = TRUE, lty = 1)
R> plot(Coxph_GBSG2_2, newdata = nd, which = "distribution",
+       type = "survivor", col = col, add = TRUE, lty = 2)
R> legend("bottomleft", lty = 1, title = "Hormonal Therapy",
+         legend = levels(nd$horTh), bty = "n", col = col)
```

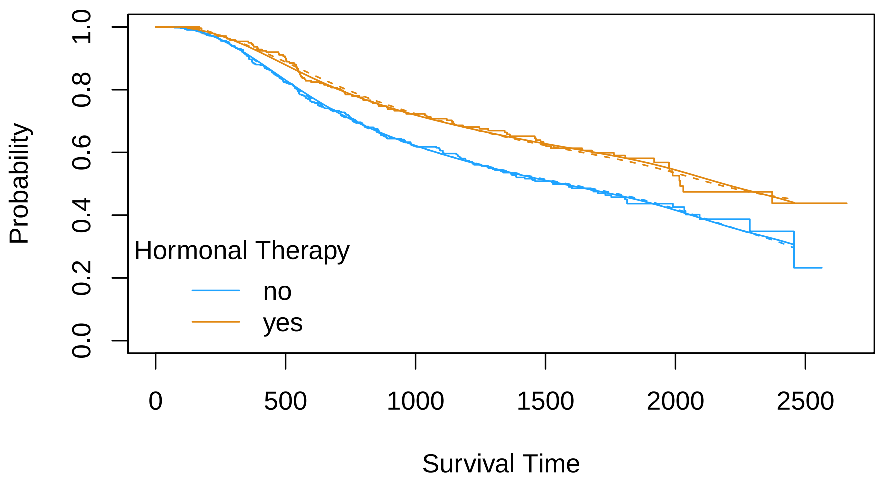

Figure 7: German Breast Cancer Study Group 2: Conditional survivor curves of treated and control patients obtained from the Kaplan-Meier estimator, model `Coxph_GBSG2_1` under proportional hazards assumption, and `Coxph_GBSG2_2` under non-proportional hazards.

Similar to continuous outcome logistic regression (Section 4), the parameters $\boldsymbol{\beta}$ can be interpreted as log-odds ratios because the conditional odds

$$\frac{\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x})}{\mathbb{P}(Y > y \mid \boldsymbol{X} = \boldsymbol{x})} = \frac{\frac{\exp(\vartheta_1 + \vartheta_2 \log(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})}{1+\exp(\vartheta_1 + \vartheta_2 \log(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})}}{\frac{1}{1+\exp(\vartheta_1 + \vartheta_2 \log(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})}} = \exp(\vartheta_1 + \vartheta_2 \log(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$$

consists of the baseline odds $\exp(\vartheta_1 + \vartheta_2 \log(y))$ and the odds ratio $\exp(-\tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})$. This model can be fitted using `Survreg()` by the choice `distribution = "loglogistic"`.

Switching to a conditional log-normal distribution (`distribution = "lognormal"`) simply means using $F_Z = \Phi$ in the model

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = \Phi(\vartheta_1 + \vartheta_2 \log(y) - \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}))$$

so that we get $\log(Y) \mid \boldsymbol{X} = \boldsymbol{x} \sim \mathrm{N}((-\vartheta_1 + \tilde{\boldsymbol{x}}^\top \boldsymbol{\beta})/\vartheta_2, 1/\vartheta^2)$.

It should be noted that both `Survreg()` and `Coxph()` are able to fit models in the presence of random censoring (left, right and interval) as well as truncation of any form (left, right, interval, each of these possibly subject-specific).

7. Ordinal Regression

For ordinal responses $Y \in \{y_1, \dots, y_K\}$ at K levels, the baseline transformation attaches one parameter to each level (except the last one) in the parameterisation

$$h_Y(y_k) = \vartheta_k \in \mathbb{R}, \quad 1 \leq k < K \quad \vartheta_1 < \dots, \vartheta_{K-1}.$$

With $F_Z = \text{expit}$, our linear transformation model is known as proportional odds model and can be fitted by the three implementations from packages **ordinal** (Christensen 2024), **MASS** (Ripley and Venables 2025) and **tram** as follows (using the wine tasting data as example)

```
R> data("wine", package = "ordinal")
R> polr_wine <- polr(rating ~ temp + contact, data = wine)
R> logLik(polr_wine)

'log Lik.' -86.49 (df=6)

R> coef(polr_wine)

  tempwarm contactyes
     2.503      1.528

R> clm_wine_1 <- clm(rating ~ temp + contact, data = wine)
R> logLik(clm_wine_1)

'log Lik.' -86.49 (df=6)

R> coef(clm_wine_1)

       1|2        2|3        3|4        4|5   tempwarm contactyes
    -1.344      1.251      3.467      5.006      2.503      1.528

R> Polr_wine_1 <- Polr(rating ~ temp + contact, data = wine)
R> logLik(Polr_wine_1)

'log Lik.' -86.49 (df=6)

R> coef(Polr_wine_1, with_baseline = TRUE)

   rating1    rating2    rating3    rating4   tempwarm contactyes
    -1.344      1.251      3.467      5.006      2.503      1.528
```

with identical results. Treating one of the variables as stratum is possible in `clm()` and `Polr()` (we use treatment contrasts for $\tilde{s}$ here)

```
R> clm_wine_2 <- clm(rating ~ temp, nominal = ~ contact, data = wine)
R> logLik(clm_wine_2)

'log Lik.' -86.21 (df=9)

R> coef(clm_wine_2)

1|2.(Intercept) 2|3.(Intercept) 3|4.(Intercept) 4|5.(Intercept)
         -1.323           1.246           3.550           4.660
 1|2.contactyes  2|3.contactyes  3|4.contactyes  4|5.contactyes
         -1.615          -1.512          -1.675          -1.051
       tempwarm
          2.519

R> Polr_wine_2 <- Polr(rating | 1 + contact ~ temp, data = wine)
R> logLik(Polr_wine_2)

'log Lik.' -86.21 (df=9)

R> coef(Polr_wine_2, with_baseline = TRUE)

rating1:(Intercept) rating2:(Intercept) rating3:(Intercept)
             -1.323               1.246               3.550
rating4:(Intercept)  rating1:contactyes  rating2:contactyes
              4.660              -1.615              -1.512
 rating3:contactyes  rating4:contactyes            tempwarm
             -1.675              -1.051               2.519
```

and again the fit is equivalent. Changing the link function F_Z from expit to Φ is also possible in both functions

```
R> clm_wine_3 <- clm(rating ~ temp, nominal = ~ contact, data = wine,
+                    link = "probit")
R> logLik(clm_wine_3)

'log Lik.' -85.33 (df=9)

R> coef(clm_wine_3)

1|2.(Intercept) 2|3.(Intercept) 3|4.(Intercept) 4|5.(Intercept)
        -0.7829          0.7521          2.1323          2.7544
 1|2.contactyes  2|3.contactyes  3|4.contactyes  4|5.contactyes
        -0.8229         -0.8892         -1.0094         -0.5818
       tempwarm
         1.5114
```

```
R> Polr_wine_3 <- Polr(rating | 1 + contact ~ temp, data = wine,
+                       method = "probit")
R> logLik(Polr_wine_3)

'log Lik.' -85.33 (df=9)

R> coef(clm_wine_3)

1|2.(Intercept) 2|3.(Intercept) 3|4.(Intercept) 4|5.(Intercept)
        -0.7829          0.7521          2.1323          2.7544
 1|2.contactyes  2|3.contactyes  3|4.contactyes  4|5.contactyes
        -0.8229         -0.8892         -1.0094         -0.5818
       tempwarm
         1.5114
```

The identical results obtain from **ordinal** and **tram** increase our empirical evidence that the implementation of linear transformation models for ordinal responses in both packages are correct. Let us look at one situation where **tram** can fit a model that would be hard to obtain (for me!) in **ordinal**. Suppose on of the wine tasters (judge 9, say) had a bad day and didn't feel comfortable to deliver her ratings accurately. Instead, she choose to check-off a range of three scores (maybe '2–4' instead of just '3'). We can represent this discrete interval-censored version of the response using the `R()` function as follows

```
R> erating <- wine$rating
R> lrating <- erating
R> rrating <- erating
R> l9 <- lrating[wine$judge == 9]
R> l9[l9 > 1] <- levels(l9)[unclass(l9[l9 > 1]) - 1]
R> r9 <- rrating[wine$judge == 9]
R> r9[r9 < 5] <- levels(r9)[unclass(r9[r9 < 5]) + 1]
R> lrating[wine$judge != 9] <- rrating[wine$judge != 9] <- NA
R> erating[wine$judge == 9] <- NA
R> lrating[wine$judge == 9] <- l9
R> rrating[wine$judge == 9] <- r9
R> which(wine$judge == 9)

[1] 65 66 67 68 69 70 71 72

R> (wine$crating <- R(erating, cleft = lrating, cright = rrating))

 [1] (1, 2]  (2, 3]  (2, 3]  (3, 4]  (3, 4]  (3, 4]  (4, NA] (4, NA]
 [9] (NA, 1] (1, 2]  (NA, 1] (2, 3]  (1, 2]  (2, 3]  (4, NA] (3, 4]
[17] (1, 2]  (2, 3]  (2, 3]  (1, 2]  (4, NA] (4, NA] (3, 4]  (3, 4]
[25] (2, 3]  (1, 2]  (2, 3]  (1, 2]  (2, 3]  (1, 2]  (4, NA] (2, 3]
[33] (1, 2]  (2, 3]  (3, 4]  (2, 3]  (2, 3]  (2, 3]  (2, 3]  (2, 3]
[41] (2, 3]  (1, 2]  (2, 3]  (1, 2]  (1, 2]  (3, 4]  (4, NA] (3, 4]
[49] (NA, 1] (NA, 1] (1, 2]  (1, 2]  (1, 2]  (2, 3]  (1, 2]  (2, 3]
[57] (1, 2]  (1, 2]  (1, 2]  (2, 3]  (2, 3]  (2, 3]  (2, 3]  (3, 4]
[65] (1, 2]  (1, 3]  (2, 4]  (1, 3]  (2, 4]  (1, 3]  (3, NA] (3, NA]
```

`Polr()` is able to deal with such types of response variables and the model can be fitted in the very same way as before

```
R> Polr_wine_4 <- Polr(crating | contact ~ temp, data = wine,
+                      method = "probit")
R> logLik(Polr_wine_4)

'log Lik.' -80.26 (df=9)

R> coef(Polr_wine_4)

tempwarm
   1.515
```

As expected, the difference is not very large, but more extreme forms of censoring (or truncation) might very well lead to substantially different models.

8. Transformation Trees and Forests

Models involving a linear predictor $\tilde{\boldsymbol{x}}^\top \boldsymbol{\beta}$ suggest a simple linear impact of the covariates on the distribution of the response. This assumption might be questioned by allowing more flexible regression relationships in the transformation model

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}) = F_Z(\boldsymbol{a}(y)^\top \boldsymbol{\vartheta}(\boldsymbol{x}))$$

where the parameters of the baseline transformation can be unspecified functions of $\boldsymbol{x}$. When we assume a simple tree structure, the model can be fitted using function `trafotree()` from **trtf**.

A transformation tree for the Boston Housing data (taking censoring in the median housing values into account) consists of two steps. First, we set-up an unconditional transformation model. We choose a model with smooth and flexible transformation function

```
R> BC_BH_0 <- BoxCox(y ~ 1, data = BostonHousing2)
R> logLik(BC_BH_0)

'log Lik.' -1739 (df=7)
```

This model is then plugged into `trafotree()`, along with a formula describing the response and partitioning ($\boldsymbol{x}$) variables

```
R> library("trtf")
R> BC_BH_4 <- trafotree(BC_BH_0,
+      formula = y ~ chas + crim + zn + indus + nox +
+                 rm + age + dis + rad + tax + ptratio + b + lstat, data =
+      BostonHousing2, control = ctree_control(minbucket = 30))
R> logLik(BC_BH_4)
```

```
'log Lik.' -1249 (df=84)
```

In each terminal node of the corresponding tree (depicted in Figure 8), a density representing the conditional distribution of median housing values given the splits in the partitioning variables is given. Location, scale and shape of these distributions vary across the tree.

One could go one step further and estimate a whole forest of such transformation trees by the `traforest()` function

```
R> BC_BH_5 <- traforest(BC_BH_0,
+      formula = y ~ chas + crim + zn + indus + nox +
+                rm + age + dis + rad + tax + ptratio + b + lstat, data =
+      BostonHousing2)
```

Another interesting option is to partition more complex models by `trafotree()`. Suppose we are interested in the identification of potential treatment effect modifiers in a Cox model of the form

$$\mathbb{P}(Y \le y \mid \boldsymbol{X} = \boldsymbol{x}, \text{horTh}) = 1 - \exp(-\exp(\boldsymbol{a}(y)^\top \boldsymbol{\vartheta}(\boldsymbol{x}) + I(\text{horTh})\beta(\boldsymbol{x})))$$

(details on tree-based subgroup analyses are given in Seibold *et al.* 2016). Here we are interested in an $\boldsymbol{x}$-varying log-hazard ratio $\beta(\boldsymbol{x})$, *i.e.* , the dependency of the success (or failure) of hormonal therapy on baseline variables $\boldsymbol{x}$. We start with the model

```
R> Coxph_GBSG2_1 <- Coxph(Surv(time, cens) ~ horTh, data = GBSG2)
R> logLik(Coxph_GBSG2_1)

'log Lik.' -2606 (df=8)

R> coef(Coxph_GBSG2_1)

horThyes
 -0.3657
```

This model estimates a global treatment effect, *i.e.* , a treatment effect which applies uniformly to all patients. We now try to find out if this model can be improved by a transformation tree (where splits are looked for with respect to the parameters defining the log-cumulative baseline hazard function, *i.e.* the intercept, and the treatment effect parameter):

```
R> Coxph_GBSG2_3 <- trafotree(Coxph_GBSG2_1,
+      formula = Surv(time, cens) ~ horTh | age + menostat + tsize +
+                                   tgrade + pnodes + progrec + estrec,
+      data = GBSG2)
R> logLik(Coxph_GBSG2_3)

'log Lik.' -2557 (df=24)

R> coef(Coxph_GBSG2_3)[, "horThyes"]
```

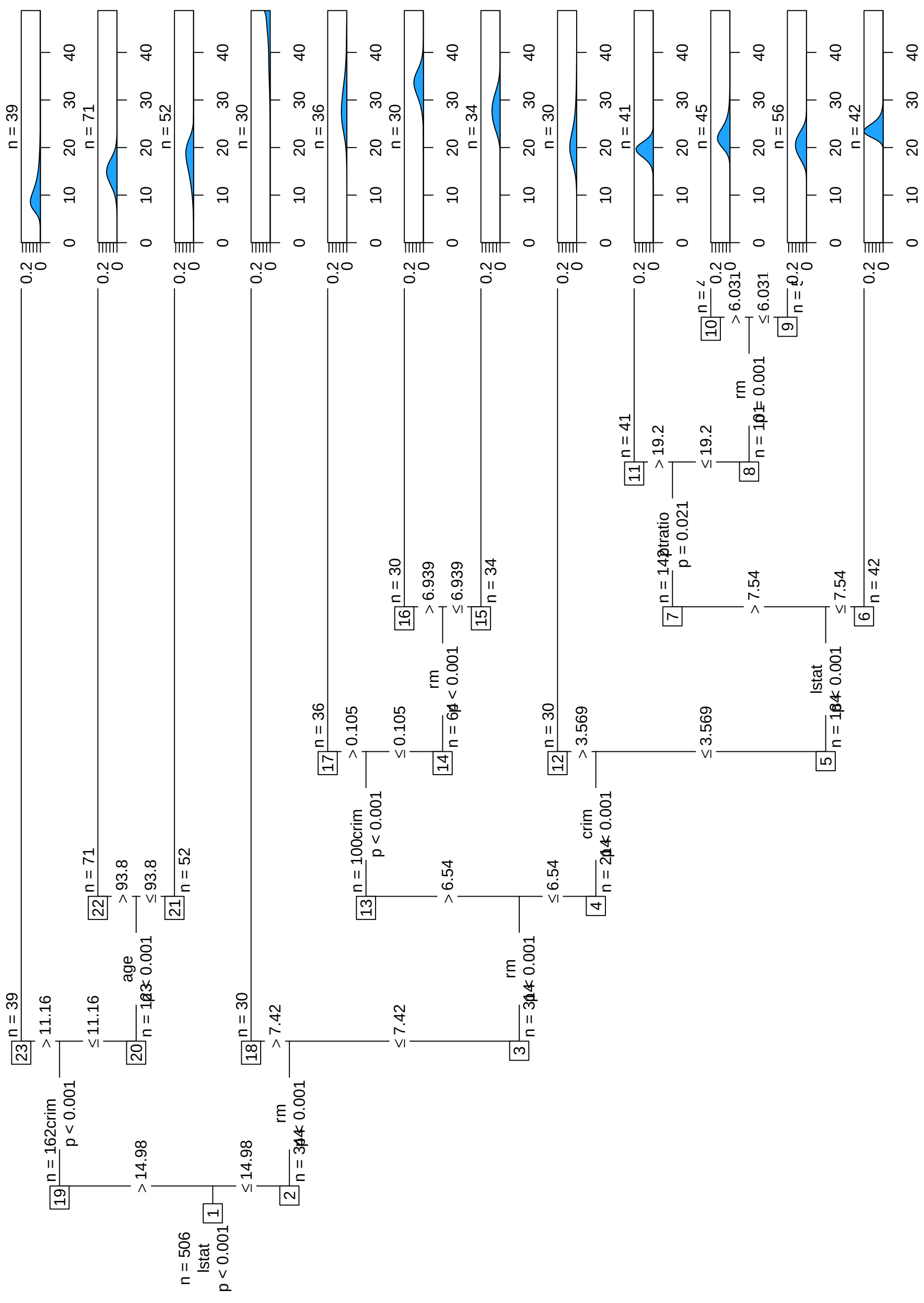

Figure 8: Boston Housing: Transformation tree with conditional transformation models in each terminal node depicted via the corresponding density function.

```
R> nd <- data.frame(horTh = sort(unique(GBSG2$horTh)))
R> plot(Coxph_GBSG2_3, newdata = nd,
+       tp_args = list(type = "survivor", col = col))
```

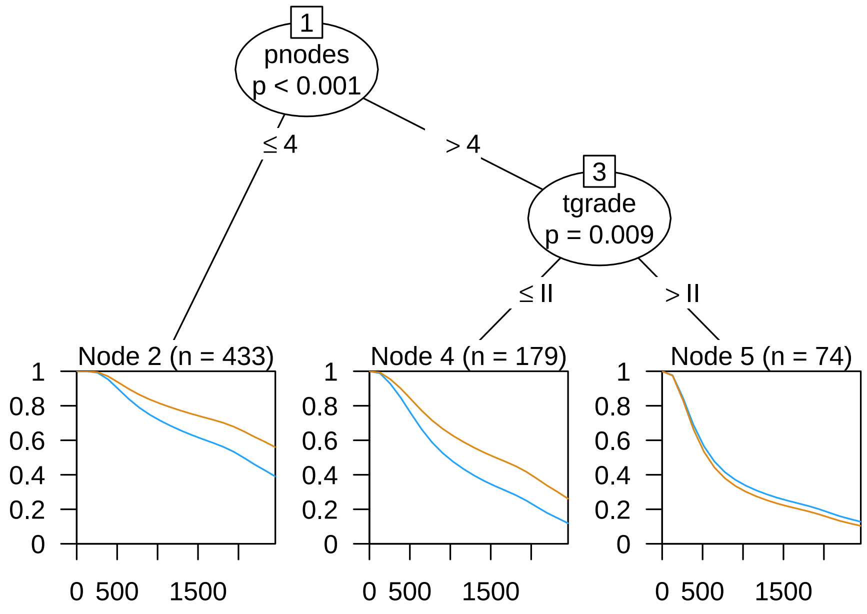

Figure 9: German Breast Cancer Study Group 2: Transformation tree with conditional survivor functions given treatment (blue is hormonal treatment, yellow refers to the control group) in each terminal node.

```
        2         4         5 
-0.48468 -0.46049  0.09648 
```

These log-hazard ratios and the tree in Figure 9 show that the global treatment effect vanishes in patients with many positive lymph nodes and a larger tumor grading.

If one is only interested in proportional hazards alternatives, one can add an explicit intercept parameter (constrained to zero) to the model such that the log-rank scores and the scores for the treatment effect parameters define the splits:

```
R> GBSG2$int <- 1
R> Coxph_GBSG2_3 <- Coxph(Surv(time, cens) ~ int + horTh, data = GBSG2,
+                         fixed = c("int" = 0))
R> (Coxph_GBSG2_4 <- trafotree(Coxph_GBSG2_3,
+      formula = Surv(time, cens) ~ int + horTh | age + menostat + tsize +
+                                    tgrade + pnodes + progrec + estrec,
+      data = GBSG2, parm = c("int", "horThyes"),
+      mltargs = list(fixed = c("int" = 0))))
```

```
Model formula:
```

```
Surv(time, cens) ~ int + horTh + (age + menostat + tsize + tgrade +
    pnodes + progrec + estrec)

Fitted party:
[1] root
|   [2] pnodes <= 3: Inf (n = 376)
|   [3] pnodes > 3
|   |   [4] progrec <= 20: 624 (n = 144)
|   |   [5] progrec > 20: 1701 (n = 166)

Number of inner nodes:    2
Number of terminal nodes: 3

R> logLik(Coxph_GBSG2_4)

'log Lik.' -2544 (df=24)

R> coef(Coxph_GBSG2_4)[, "horThyes"]

      2       4       5
-0.5559 -0.1104 -0.5553
```

9. Summary

The **tram** package simplifies model estimation, interpretation and inference for some members of the broad family of conditional transformation models. Where implementations for special cases already exist, practically equivalent results are obtained. In some aspects, such as different ways of handling stratum variables, response-varying coefficients, likelihood estimation for interval-censored or truncated data, etc., the **tram** package offers additional functionality. Users requiring more control over model specification are welcome to study facilities for model specification, estimation and inference provided by package **mlt**.

References

Box GEP, Cox DR (1964). "An Analysis of Transformations." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **26**(2), 211–252.

Christensen RHB (2024). ***ordinal**: Regression Models for Ordinal Data*. doi:10.32614/CRAN.package.ordinal. R package version 2023.12-4.1, URL https://CRAN.R-project.org/package=ordinal.

Hothorn T (2018). "Most Likely Transformations: The **mlt** Package." *Journal of Statistical Software*. Accepted for publication 2018-03-05, URL https://cran.r-project.org/web/packages/mlt.docreg/vignettes/mlt.pdf.

Hothorn T (2025a). ***mlt**: Most Likely Transformations*. R package version 1.7-2, URL http://ctm.R-forge.R-project.org.

Hothorn T (2025b). ***trtf**: Transformation Trees and Forests*. doi:10.32614/CRAN.package.trtf. R package version 0.4-3, URL https://CRAN.R-project.org/package=trtf.

Hothorn T, Möst L, Bühlmann P (2018). "Most Likely Transformations." *Scandinavian Journal of Statistics*, **45**(1), 110–134. doi:10.1111/sjos.12291.

Hothorn T, Zeileis A (2017). "Transformation Forests." *Technical report*, arXiv 1701.02110, v2. URL https://arxiv.org/abs/1701.02110.

Lohse T, Rohrmann S, Faeh D, Hothorn T (2017). "Continuous Outcome Logistic Regression for Analyzing Body Mass Index Distributions." *F1000Research*, **6**, 1933. doi:10.12688/f1000research.12934.1.

Ripley B, Venables B (2025). ***MASS**: Support Functions and Datasets for Venables and Ripley's MASS*. doi:10.32614/CRAN.package.MASS. R package version 7.3-65, URL https://CRAN.R-project.org/package=MASS.

Seibold H, Zeileis A, Hothorn T (2016). "Model-Based Recursive Partitioning for Subgroup Analyses." *International Journal of Biostatistics*, **12**(1), 45–63. doi:10.1515/ijb-2015-0032.

Therneau TM (2024). ***survival**: Survival Analysis*. doi:10.32614/CRAN.package.survival. R package version 3.8-3, URL https://CRAN.R-project.org/package=survival.

Affiliation:

Torsten Hothorn
Institut für Epidemiologie, Biostatistik und Prävention
Universität Zürich
Hirschengraben 84, CH-8001 Zürich, Switzerland
`Torsten.Hothorn@R-project.org`