Using the `raschtree` function for detecting differential item functioning in the Rasch model

Carolin Strobl
Universität Zürich

Julia Kopf
Ludwig-Maximilians-Universität München

Achim Zeileis
Universität Innsbruck

Abstract

The **psychotree** package contains the function `raschtree`, that can be used to detect differential item functioning (DIF) in the Rasch model. The DIF detection method implemented in `raschtree` is based on the model-based recursive partitioning framework of Zeileis, Hothorn, and Hornik (2008) and employs generalized M-fluctuation tests (Zeileis and Hornik 2007) for detecting differences in the item parameters between different groups of subjects. The statistical methodology behind `raschtree` is described in detail in Strobl, Kopf, and Zeileis (2010). The main advantage of this approach is that it allows to detect groups of subjects exhibiting DIF, that are not pre-specified, but are detected automatically from combinations of covariates. In this vignette, the practical usage of `raschtree` is illustrated.

Keywords: Item response theory, IRT, Rasch model, differential item functioning, DIF, structural change, multidimensionality.

1. Differential item functioning in the Rasch model

A key assumption of the Rasch model is that the item parameter estimates should not depend on the person sample (and vice versa). This assumption may be violated if certain items are easier or harder to solve for certain groups of subjects – regardless of their true ability – in which case we speak of differential item functioning (DIF).

In order to detect DIF with the `raschtree` function, the item responses and all covariates that should be tested for DIF need to be handed over to the method, as described below. Then the following steps are conducted:

1. At first, one joint Rasch model is fit for all subjects.

2. Then it is tested statistically whether the item parameters differ along any of the covariates.

3. In that case the sample is split along that covariate and two separate Rasch models are estimated.

4. This process is repeated as long as there is further DIF (and the subsample is still large enough).

For details on the underlying statistical framework implemented in `raschtree` see Strobl *et al.* (2010).

The main advantage of the Rasch tree approach is that DIF can be detected between groups of subjects created by more than one covariate. For example, certain items may be easier for male subjects over the age of 40 as opposed to all other subjects. In this case DIF is associated with an interaction of the variables gender and age, rather than any one variable alone.

Moreover, with this approach it is not necessary to pre-define cutpoints in continuous variables, as would be the standard approach when using, e.g., a likelihood ratio or Wald test: Usually, age groups are pre-specified, for example by means of splitting at the median. However, the median may not be where the actual parameter change occurs – it could be that only very young or very old subjects find certain items particularly easy or hard. By splitting at the median this effect may be disguised. Therefore, the Rasch tree method searches for the value corresponding to the strongest parameter change and splits the sample at that value. Certain statistical techniques are necessary for doing this in a statistically sound way, as described in detail in Strobl *et al.* (2010).

Now the practical application of `raschtree` is outlined, starting with the data preparation.

2. Data preparation

When using `raschtree` for the first time, the **psychotree** package needs to be installed first:

```
> install.packages("psychotree")
```

After this, the package is permanently installed on the computer, but needs to be made available at the start of every new R session:

```
> library("psychotree")
```

The package contains a data example for illustrating the Rasch trees, that can be loaded with:

```
> data("SPISA", package = "psychotree")
```

The data set `SPISA` consists of the item responses and covariate values of 1075 subjects. It is a subsample of a larger data set from an online quiz, that was carried out by the German weekly news magazine SPIEGEL in 2009 via the online version of the magazine SPIEGEL Online (SPON). The quiz was designed for testing one's general knowledge and consisted of a total of 45 items from five different topics: politics, history, economy, culture and natural sciences. A thorough analysis and discussion of the original data set is provided in Trepte and Verbeet (2010).

The data are structured in the following way: The variable `spisa` contains the 0/1-responses of all subjects to all test items (i.e., `spisa` is only a single variable but contains a matrix of responses). In addition to that, covariates like age and gender are available for each subject:

Item reponses											Covariates				
spisa											gender	age	semester	elite	spon
1	0	0	1	1	⋯	0	1	1	1	1	female	21	3	no	1–3/month
0	1	0	1	1	⋯	1	1	1	1	1	male	20	1	no	4–5/week
0	0	0	1	0	⋯	0	1	1	1	1	female	25	9	no	1–3/month
0	0	1	1	1	⋯	1	1	0	1	1	male	27	10	no	never
1	1	1	1	1	⋯	0	0	1	1	1	male	24	8	no	1/week
1	0	0	1	0	⋯	1	1	0	1	1	male	20	1	yes	1–3/month
					⋮						⋮	⋮	⋮	⋮	⋮

If your own data set, termed for example `mydata`, is in a different format, it is easy to change it into the right format for `raschtree`. For example, if the item responses are coded as individual variables like this:

Item reponses					Covariates		
item1	item2	item3	item4	item5	gender	age	semester
1	0	0	1	1	female	21	3
0	1	0	1	1	male	20	1
0	0	0	1	0	female	25	9
0	0	1	1	1	male	27	10
1	1	1	1	1	male	24	8

You can bring them into more convenient format by first defining a new variable `resp` that contains the matrix of item responses (i.e., first five columns of `mydata`):

```
> mydata$resp <- as.matrix(mydata[ , 1:5])
```

Then you can omit the original separate item response variables from the data set

```
> mydata <- mydata[ , -(1:5)]
```

The data set then contains both the complete matrix of item responses – termed `resp` – and the covariates as individual columns, so that later it is easier to address the complete matrix of item responses in the function call.

If the item responses include cases where all observed item responses are 0 or all observed item responses are 1, these cases need to be excluded prior to model fitting. (These cases do not contribute to the Rasch model anyway, because they do not contain any information on which items are easier or harder to solve, and the corresponding person parameters are not identified.) For example, the cases indicated by arrows in the example below would need to be excluded, because all observed item responses are either 0 or 1:

	Item reponses									
	resp									
	1	NA	0	1	1	1	0	1	1	1
⇒	1	1	1	1	NA	1	1	1	1	1
⇒	0	0	NA	0	0	0	0	0	NA	0
	0	0	1	1	1	0	1	0	1	0
⇒	1	1	1	1	1	1	1	1	1	1

To exclude rows where all observed item responses are either 0 or 1, we select only the subset of cases for which the proportion of correct item responses is strictly between 0 and 1 for futher analysis.

```
> mydata <- subset(mydata, rowMeans(resp, na.rm = TRUE) > 0 &
+   rowMeans(resp, na.rm = TRUE) < 1)
```

Now the data preparation is done and we can fit a Rasch tree.

3. Model fitting, plotting and extraction of parameter values

The idea of Rasch trees is to model differences in the Rasch model for the item responses by means of the covariates. This idea translates intuitively into the formula interface that is commonly used in R functions, such as `lm` for linear models: In a linear model, where the response variable `y` is modeled by the covariates `x1` and `x2`, the formula in R looks like this:

```
y ~ x1 + x2
```

Very similarly, in the Rasch tree for our `SPISA` data, where the item responses `spisa` are modeled by the covariates `age`, `gender`, `semester`, `elite` and `spon`, the formula used in `raschtree` looks like this:

```
spisa ~ age + gender + semester + elite + spon
```

The complete call is

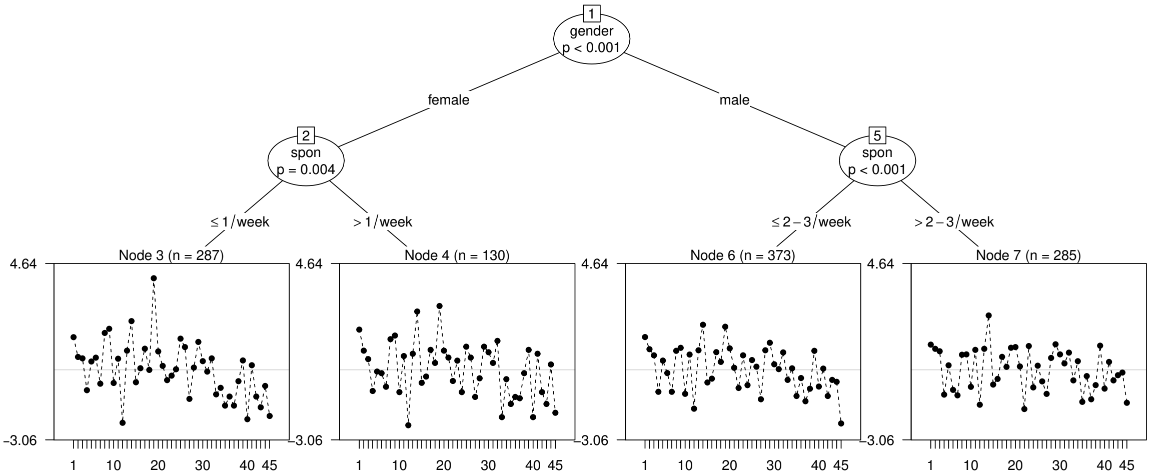
```
> my_first_raschtree <- raschtree(spisa ~ age + gender +
+   semester + elite + spon, data = SPISA)
```

Note that the model is not only fitted, but also saved under the name `my_first_raschtree`, so that we can later extract information from the fitted model object and plot the Rasch tree.

As a shortcut, when all other variables in the data set are to be used as covariates, as in our example, the covariates do not have to be listed explicitly in the formula but can replaced by a dot, as in `raschtree(spisa ~ ., data = SPISA)` (leading to equivalent output as the call above). Moreover, if you want to see the process of the Rasch tree fitting, including the computation of the p-values and corresponding split decisions in each step, you can use the `verbose` option, as in `raschtree(spisa ~ ., data = SPISA, verbose = TRUE)`. The `verbose` option also has the advantage that you can see something happening on your screen when `raschtree` takes a while to complete – which may be the case if there are many variables with DIF and if these variables offer many possible cutpoints, like continuous variables and factors with many categories.

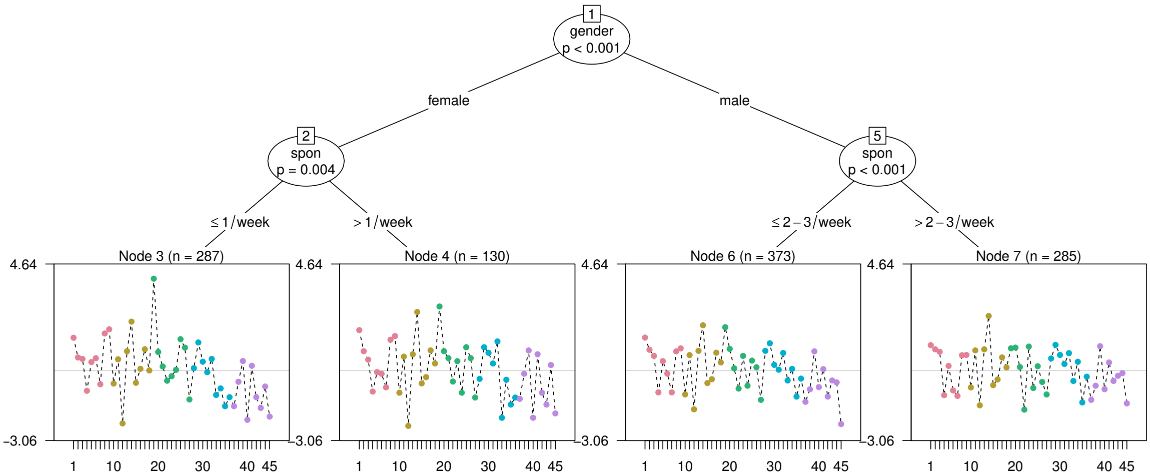
The resulting Rasch tree can then be plotted with the generic `plot` call:

```
> plot(my_first_raschtree)
```



The plot function also accepts many options for standard plot functions, including coloring. Here, a set of "rainbow" colors (from package **colorspace**, see Zeileis, Hornik, and Murrell 2009) is employed to indicate the blocks of nine items from each of the five different topics covered in the quiz: politics, history, economy, culture and natural sciences:

```
> library("colorspace")
> plot(my_first_raschtree,
+        col = rep(rainbow_hcl(5, c = 65, l = 65), each = 9))
```



For extracting the estimated item parameters for each group, there are two different calls corresponding to the two different ways to scale the item parameters: The parameters of a Rasch model are unique only up to linear transformations. In particular, the origin of the scale is not fixed but chosen arbitrarily. There are two common ways to choose the origin: setting one item parameter to zero or setting the sum of all item parameters to zero. Accordingly, there are two calls to extract the item parameters from `raschtree` one way or the other:

```
> coef(my_first_raschtree, node = 4)

    spisa2     spisa3     spisa4     spisa5     spisa6     spisa7     spisa8
-0.9187137 -1.2874521 -2.6805353 -1.8312493 -1.9026320 -2.4951461 -0.4162699
    spisa9    spisa10    spisa11    spisa12    spisa13    spisa14    spisa15
-0.2581010 -2.7296693 -1.1543021 -4.1769262 -1.0539421  0.7916895 -2.3241647
   spisa16    spisa17    spisa18    spisa19    spisa20    spisa21    spisa22
-2.0495272 -0.8845425 -1.4541652  1.0321505 -0.9187137 -1.2208998 -2.2428880
   spisa23    spisa24    spisa25    spisa26    spisa27    spisa28    spisa29
-1.3540086 -2.7296693 -0.7458437 -1.2208998 -2.9403761 -2.1253688 -0.7458437
   spisa30    spisa31    spisa32    spisa33    spisa34    spisa35    spisa36
-0.9865644 -1.4541652 -0.4922363 -3.8232693 -2.1640318 -3.2469996 -2.9403761
   spisa37    spisa38    spisa39    spisa40    spisa41    spisa42    spisa43
-2.9972355 -1.9026320 -0.8845425 -3.8232693 -1.0539421 -2.7296693 -3.2469996
   spisa44    spisa45
-1.5213581 -3.6324888
```

where the parameter for the first item is set to zero and therefore not displayed (the call is termed `coef`, because that is the name of the call extracting the estimated parameters, or coefficients, from standard regression models generated, e.g., with the `lm` function) and

```
> itempar(my_first_raschtree, node = 4)

     spisa1      spisa2      spisa3      spisa4      spisa5      spisa6
 1.75417311  0.83545944  0.46672105 -0.92636220 -0.07707618 -0.14845889
     spisa7      spisa8      spisa9     spisa10     spisa11     spisa12
-0.74097296  1.33790319  1.49607208 -0.97549614  0.59987100 -2.42275309
    spisa13     spisa14     spisa15     spisa16     spisa17     spisa18
 0.70023103  2.54586259 -0.56999156 -0.29535409  0.86963065  0.30000788
    spisa19     spisa20     spisa21     spisa22     spisa23     spisa24
 2.78632359  0.83545944  0.53327329 -0.48871486  0.40016448 -0.97549614
    spisa25     spisa26     spisa27     spisa28     spisa29     spisa30
 1.00832941  0.53327329 -1.18620295 -0.37119569  1.00832941  0.76760868
    spisa31     spisa32     spisa33     spisa34     spisa35     spisa36
 0.30000788  1.26193682 -2.06909621 -0.40985866 -1.49282653 -1.18620295
    spisa37     spisa38     spisa39     spisa40     spisa41     spisa42
-1.24306243 -0.14845889  0.86963065 -2.06909621  0.70023103 -0.97549614
    spisa43     spisa44     spisa45
-1.49282653  0.23281497 -1.87831571
```

where the item parameters by default sum to zero (other restrictions can be specified as well).

Here the item parameters have been displayed only for the subjects in node number 4 (representing female students who access the online magazine more than once per week) to save space. The item parameters for all groups can be displayed by omitting the `node` argument.

4. Interpretation

Ideally, if none of the items showed DIF, we would find a tree with only one single node. In this case, one joint, unidimensional Rasch model would be appropriate to describe the entire data set.

If however, the Rasch tree shows at least one split, DIF is present and it is not appropriate to compare the different groups of subjects with the test. The DIF may be caused by certain characteristics of the items, such as their wording, but may also be an indicator of multidimensionality:

If, for example, certain groups of subjects are disadvantaged by the wording or content of certain items, it would be unfair to compare the different groups with the test including those items. In practice, items showing DIF will then be excluded from the test before rating the subjects' performance. Sometimes it is also possible to rephrase the items, for example when DIF is present only for subjects who are not native speakers of the test language.

If, however, in a general knowledge quiz, e.g., all history items are easier for a particular group of subjects, this may indicate that history knowledge should be considered as a sub-dimension of general knowledge (in which the particular group happens to outperform the others). In this case, a multidimensional Rasch model would be called for (that is unfortunately not available in R yet).

Note in particular that when one joint, unidimensional Rasch model is not appropriate to describe the test, this also means that a ranking of the subjects based on the raw scores (i.e., the number of items that each subject answered correctly) is not appropriate either, because this would also assume that the test is unidimensional.

5. Outlook

We are currently working on functionality for facilitating the interpretation of the Rasch trees by means of summarizing in tables which items show the strongest DIF with respect to which groups, and on generalizations of the method to, e.g., the partial credit model for items with more than two response categories.

Acknowledgements

The work of Carolin Strobl was supported by grant STR1142/1-1 ("Methods to Account for Subject-Covariates in IRT-Models") from the German Research Foundation (Deutsche Forschungsgemeinschaft). The authors would like to thank Reinhold Hatzinger for important insights stimulated by conversations and the R package **eRm** (Mair and Hatzinger 2007; Mair, Hatzinger, and Maier 2010).

References

Mair P, Hatzinger R (2007). "Extended Rasch Modeling: The **eRm** Package for the Application of IRT Models in R." *Journal of Statistical Software*, **20**(9), 1–20. URL http://www.jstatsoft.org/v20/i09/.

Mair P, Hatzinger R, Maier M (2010). ***eRm**: Extended Rasch Modeling.* R package version 0.13-0, URL http://CRAN.R-project.org/package=eRm.

Strobl C, Kopf J, Zeileis A (2010). "A New Method for Detecting Differential Item Functioning in the Rasch Model." *Technical Report 92*, Department of Statistics, Ludwig-Maximilians-Universität München. URL http://epub.ub.uni-muenchen.de/11915/.

Trepte S, Verbeet M (eds.) (2010). *Allgemeinbildung in Deutschland – Erkenntnisse aus dem SPIEGEL Studentenpisa-Test.* VS Verlag, Wiesbaden.

Zeileis A, Hornik K (2007). "Generalized M-Fluctuation Tests for Parameter Instability." *Statistica Neerlandica*, **61**(4), 488–508.

Zeileis A, Hornik K, Murrell P (2009). "Escaping RGBland: Selecting Colors for Statistical Graphics." *Computational Statistics & Data Analysis*, **53**(9), 3259–3270. doi:10.1016/j.csda.2008.11.033.

Zeileis A, Hothorn T, Hornik K (2008). "Model-Based Recursive Partitioning." *Journal of Computational and Graphical Statistics*, **17**(2), 492–514.

Affiliation:

Carolin Strobl
Department of Psychology
Universität Zürich
Binzmühlestr. 14
8050 Zürich, Switzerland
E-mail: Carolin.Strobl@psychologie.uzh.ch
URL: http://www.psychologie.uzh.ch/fachrichtungen/methoden.html

Julia Kopf
Department of Statistics
Ludwig-Maximilians-Universität München
Ludwigstraße 33
80539 München, Germany
E-mail: Julia.Kopf@stat.uni-muenchen.de

Achim Zeileis
Department of Statistics
Faculty of Economics and Statistics
Universität Innsbruck
Universitätsstr. 15
6020 Innsbruck, Austria
E-mail: Achim.Zeileis@R-project.org