# strucchange: Model-Based Testing, Monitoring, and Dating of Structural Changes in R

Achim Zeileis

http://eeecon.uibk.ac.at/~zeileis/

# Overview

- Example: Seatbelt data
- Structural change methods
  - Model frame
  - Testing
  - Monitoring
  - Dating
- Beyond the linear regression model
- Challenges/wishlist
- Summary

# Overview

**History:** Work on structural change methods since Master's thesis.

**Packages:** Methodological work is accompanied by software implemented in the R system for statistical computing in packages *strucchange* and *fxregime*. Available from the Comprehensive R Archive Network at `http://CRAN.R-project.org/`.

**Content:**

- Testing, monitoring, and dating structural changes in linear regression model.
- Score-based tests for structural change in general parametric models with M-type estimators (least squares, maximum likelihood, instrumental variables, robust M-estimation, ...).
- Testing, monitoring, and dating structural changes in Gaussian regression models (including error variance).
- Some more bits and pieces for general parametric models.

# Example: Seatbelt data

**Data:** Monthly totals of car drivers in Great Britain killed or seriously injured from 1969(1) to 1984(12).

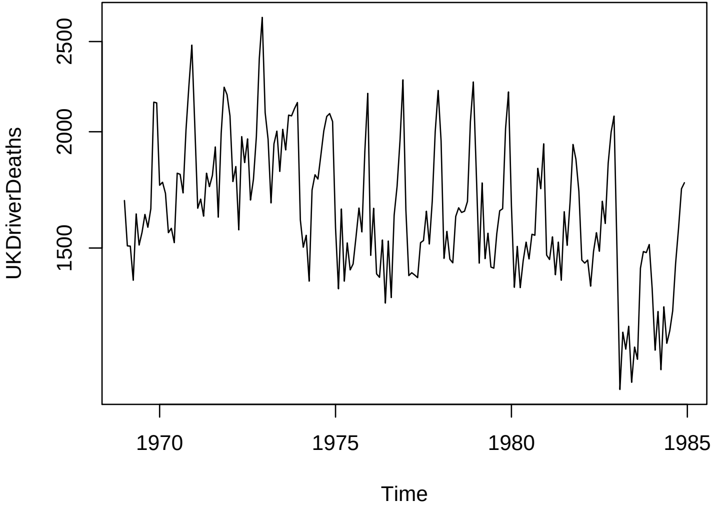
**Source:** Harvey AC, Durbin J (1986). “The Effects of Seat Belt Legislation on British Road Casualties: A Case Study in Structural Time Series Modelling.” *Journal of the Royal Statistical Society A*, **149**(3), 187–227.

**Intervention:** Compulsory wearing of seat belts was introduced on 1983-01-31.

**Here:** Employ knowledge about intervention only in monitoring illustration.

# Example: Seatbelt data

```
R> plot(UKDriverDeaths, log = "y")
```



# Model frame

**Generic idea:** Consider a regression model for $n$ ordered observations $y_i \mid x_i$ with $k$-dimensional parameter $\theta$. Ordering is typically with respect to time in time-series regressions, but could also be with respect to income, age, etc. in cross-section regressions.

**Estimation:** To fit the model to observations $i = 1, \dots, n$ an additive objective function $\Psi(y, x, \theta)$ is used such that

$$\widehat{\theta} \; = \; \underset{\theta}{\text{argmin}} \sum_{i=1}^{n} \Psi(y_i, x_i, \theta).$$

This can also be defined implicitly based on the corresponding score function (or estimating function) $\psi(y, x, \theta) = \partial \Psi(y, x, \theta) / \partial \theta$:

$$\sum_{i=1}^{n} \psi(y_i, x_i, \widehat{\theta}) \; = \; 0.$$

# Model frame

**Special cases:** (Ordinary) least squares (OLS), maximum likelihood (ML), instrumental variables, quasi-ML, robust M-estimation, etc.

**Central limit theorem:** Under parameter stability and some mild regularity conditions

$$\sqrt{n}(\widehat{\theta} - \theta_0) \xrightarrow{d} \mathcal{N}(0, V(\theta_0)),$$

where the covariance matrix is

$$V(\theta_0) = \{A(\theta_0)\}^{-1} B(\theta_0) \{A(\theta_0)\}^{-1}$$

and $A$ and $B$ are the expectation of the derivative of $\psi$ and its variance respectively.

# Model frame

**Special case:** For the standard linear regression model

$$y_i \;=\; x_i^\top \beta \;+\; \varepsilon_i$$

with coefficients $\beta$ and error variance $\sigma^2$ one can either treat $\sigma^2$ as a nuisance parameter $\theta = \beta$ or include it as $\theta = (\beta, \sigma^2)$.

In the former case, the estimating functions are $\psi = \psi_\beta$

$$\psi_\beta(y, x, \beta) \;=\; (y - x^\top \beta)\, x$$

and in the latter case, they have an additional component

$$\psi_{\sigma^2}(y, x, \beta, \sigma^2) \;=\; (y - x^\top \beta)^2 - \sigma^2.$$

and $\psi = (\psi_\beta, \psi_{\sigma^2})$. Here, focus on $\beta$.

# Model frame: Seatbelt data

**Example:** OLS regression for log-deaths with lag and seasonal lag, roughly corresponding to SARIMA(1, 0, 0)(1, 0, 0)$_{12}$ model.

```
R> dd <- log(UKDriverDeaths)
R> dd <- ts.intersect(dd = dd, dd1 = lag(dd, -1), dd12 = lag(dd, -12))
R> coeftest(lm(dd ~ dd1 + dd12, data = dd))

t test of coefficients:

            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.4205     0.3633    1.16     0.25
dd1           0.4310     0.0533    8.09  9.1e-14 ***
dd12          0.5112     0.0565    9.04  2.7e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

# Model frame: Questions

**Testing:** Given that a model with parameter $\widehat{\theta}$ has been estimated for these *n* observations, the question is whether this is appropriate or: *Are the parameters stable or did they change through the sample period* $i = 1, \dots, n$*?*

**Monitoring:** Given that a stable model could be established for these *n* observations, the question is whether it remains stable in the future or: *Are incoming observations for* $i > n$ *still consistent with the established model or do the parameters change?*

**Dating:** Given that there is evidence for a structural change in $i = 1, \dots, n$, it might be possible that stable regression relationships can be found on subsets of the data. *How many segments are in the data? Where are the breakpoints?*

# Testing

**Null hypothesis:** To assess the stability of the fitted model with $\widehat{\theta}$, we want to test

$$H_0: \ \theta_i = \theta_0 \qquad (i = 1, \dots, n)$$

against the alternative that $\theta_i$ varies over "time" $i$.

**Alternative:** Various patterns of deviation from $H_0$ are conceivable: single/multiple break(s), random walks, etc.

**Idea:** Assess fluctuation in measures of model deviation or test statistics against a (single) break alternative.

# Testing

**Testing procedure:**

- Empirical fluctuation processes captures fluctuation in (partial sums of)
  - residuals (e.g., OLS, recursive),
  - scores,
  - parameter estimates (e.g., recursive, rolling), or
  - test statistics for a (single) break alternative.
- Theoretical limiting process is obtained through functional central limit theorem (typically functional of Brownian motion/bridge).
- Choose boundaries which are crossed by the limiting process (or some transformation of it) only with a known probability $\alpha$.
- If the empirical fluctuation process crosses the theoretical boundaries the fluctuation is improbably large $\Rightarrow$ reject the null hypothesis.

# Testing: Software

**For the linear regression model:**

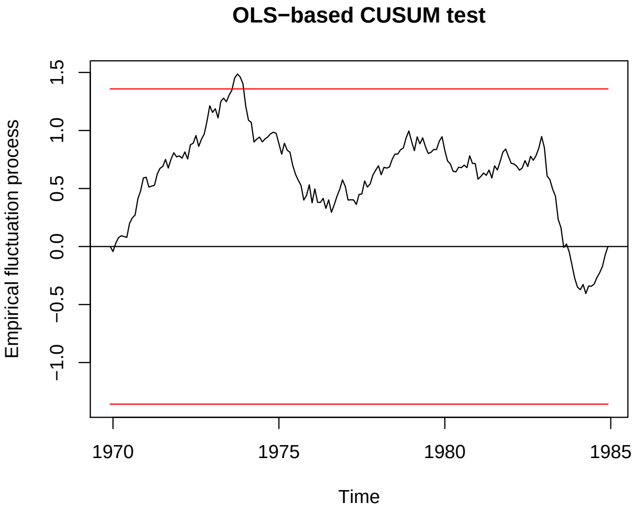
- `efp()` computes various CUSUM or MOSUM processes based on recursive or OLS residuals, parameter estimates, or scores.
- `Fstats()` compute the sequence of $F$ statistics (LR/Wald) for all single break alternatives (given trimming).
- Significance tests can be performed graphically by `plot()` method while statistic and $p$ value are computed by `sctest()` method.

**For general models:** Object-oriented implementation.

- `gefp()` computes CUSUM process from scores of model object.
- Relies on `estfun()` method (from *sandwich* package) for extracting the empirical scores (aka estimating functions).
- `efpFunctional()` simulates critical values for functionals of Brownian bridges and set up visualization functions.
- Methods for `plot()` and `sctest()` perform the significance tests.

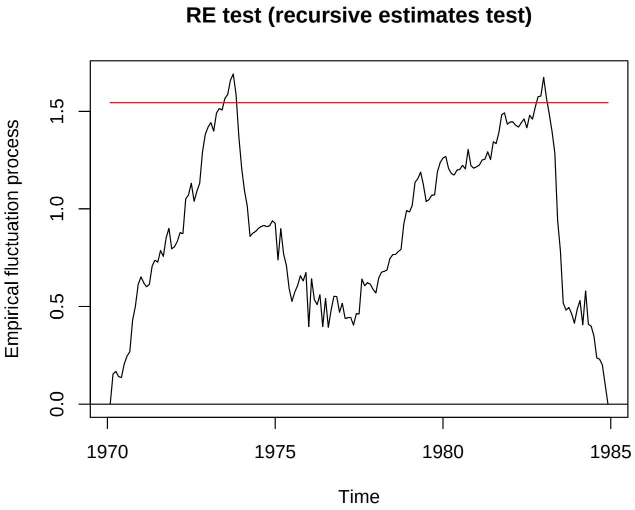
# Testing: Seatbelt data

```
R> ocus <- efp(dd ~ dd1 + dd12, data = dd, type = "OLS-CUSUM")
R> plot(ocus)
```



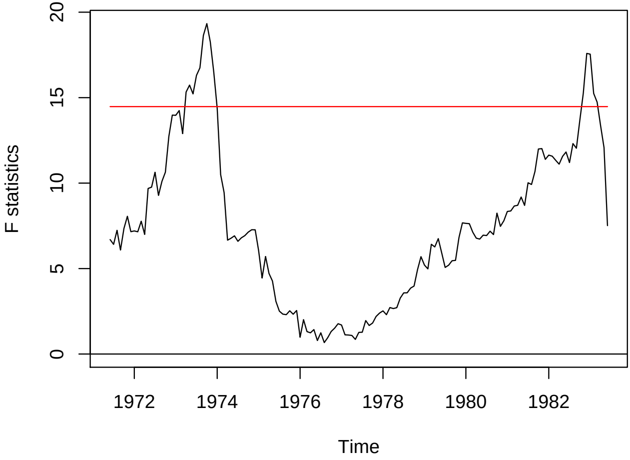
# Testing: Seatbelt data

```
R> re <- efp(dd ~ dd1 + dd12, data = dd, type = "RE")
R> plot(re)
```



# Testing: Seatbelt data

```
R> fs <- Fstats(dd ~ dd1 + dd12, data = dd, from = 0.1)
R> plot(fs)
```



# Testing: Seatbelt data

```
R> sctest(ocus)

OLS-based CUSUM test

data:  ocus
S0 = 1.487, p-value = 0.02407

R> sctest(re)

RE test (recursive estimates test)

data:  re
RE = 1.691, p-value = 0.01956

R> sctest(fs)

supF test

data:  fs
sup.F = 19.33, p-value = 0.006721
```

# Monitoring

**Idea:** Fluctuation tests can be applied sequentially to monitor models.

**More formally:** Sequentially test the null hypothesis

$$H_0: \ \theta_i = \theta_0 \qquad (i > n)$$

against the alternative that $\theta_i$ changes at some time in the future $i > n$.

**Basic assumption:** The model parameters are stable $\theta_i = \theta_0$ in the history period $i = 1, \dots, n$.

**Test statistics:** Update the fluctuation process and re-compute the associated test statistic in the monitoring period $i > n$.

**Critical values:** For sequential testing not only a single critical value is needed, but a full boundary function. This can direct power to early or late changes or try to spread the power evenly.

# Monitoring: Software

**For the linear regression model:**

- `mefp()` initializes a monitoring fluctuation process based on various types of CUSUM or MOSUM for recursive or OLS residuals or parameter estimates.
- `monitor()` conducts monitoring as new data becomes available.
- Results can be inspected by `print()` or `plot()` methods.
- `fxmonitor()` from *fxregime* computes CUSUM process of scores (including error variance), again accompanied by suitable methods.

**For general models:** Object-oriented implementation.

- Various general techniques available in literature.
- None implemented yet in *strucchange*.

# Monitoring: Seatbelt data

**Initialization:** Select 1976(1) until 1982(12) as the history period, fit OLS regression, and compute MOSUM process of OLS residuals (with bandwidth $n/4$).

```
R> mdd <- window(dd, start = c(1976, 1), end = c(1982, 12))
R> mcus <- mefp(dd ~ dd1 + dd12, data = mdd,
+     type = "OLS-MOSUM", h = 0.25)
```

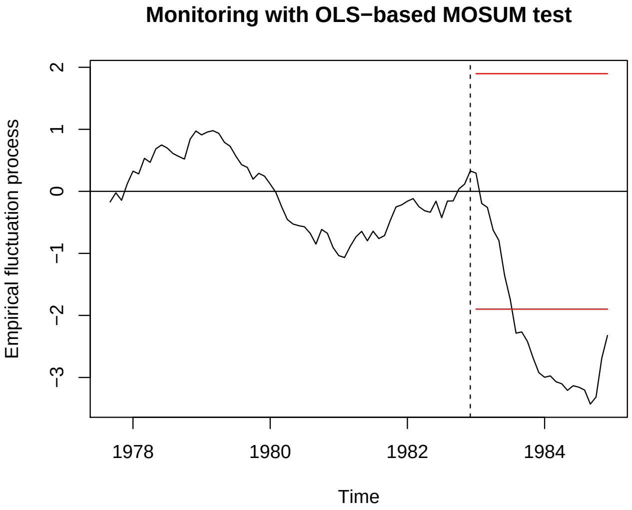
**Monitoring:** Make monitoring period data available, i.e., all data since 1976(1) until 1984(12) and conduct monitoring.

```
R> mdd <- window(dd, start = c(1976, 1))
R> mcus <- monitor(mcus)
```

```
Break detected at observation # 92
```

# Monitoring: Seatbelt data

```
R> plot(mcus, functional = NULL)
```



# Monitoring: Seatbelt data

```
R> mcus

Monitoring with OLS-based MOSUM test

Initial call:
  mefp.formula(formula = dd ~ dd1 + dd12, type = "OLS-MOSUM", data = mdd

Last call:
  monitor(obj = mcus)

Significance level   :  0.05
Critical value       :  1.342
History size         :  84
Last point evaluated :  108
Structural break at  :  92

Parameter estimate on history :
(Intercept)         dd1        dd12
     1.1451      0.1317      0.7134
```

# Dating

**Segmented regression model:** A stable model with parameter vector $\theta^{(j)}$ holds for the observations in $i = i_{j-1} + 1, \dots, i_j$. The segment index is $j = 1, \dots, m + 1$.

**Estimation:** Given the number of breakpoints $m$, these can be estimated by minimizing the segmented objective function

$$\sum_{j=1}^{m+1} \sum_{i=i_{j-1}+1}^{i_j} \Psi(y_i, x_i, \widehat{\theta}^{(j)}).$$

with respect to $i_1, \dots, i_m$. $\widehat{\theta}^{(j)}$ is the segment-specific estimate of the parameters and $i_0 = 0$, $i_{m+1} = n$

**Model selection:** If $m$ is unknown, it can be selected by means of information criteria (AIC, BIC, LWZ, MDL, etc.) or sequential tests.

# Dating: Software

**For the linear regression model:**

- `breakpoints()` minimizes residual sum of squares for all *m* using dynamic programming algorithm (exploiting recursive residuals).
- `plot()`, `summary()`, `AIC()` methods for selection of *m*.
- `breakpoints()` and `breakdates()` methods can extract estimated breakpoints (for any *m*).
- `confint()` computes the associated confidence intervals.
- `coef()` extracts estimated regression coefficients (for any *m*) or `breakfactor()` can be leveraged for reestimation.

**For general models:** Object-oriented implementation.

- `fxregimes()` in *fxregime* optimizes Gaussian negative log-likelihood of linear regression model (i.e., including variance).
- Employs unexported `gbreakpoints()` for optimizing additive objective functions via dynamic programming (extremely slow).

# Dating: Seatbelt data

```
R> bp <- breakpoints(dd ~ dd1 + dd12, data = dd, h = 0.1, breaks = 5)
R> summary(bp)

 Optimal (m+1)-segment partition:

Call:
breakpoints.formula(formula = dd ~ dd1 + dd12, h = 0.1, breaks = 5,
    data = dd)

Breakpoints at observation number:

m = 1   46
m = 2   46              157
m = 3   46 70           157
m = 4   46 70 108       157
m = 5   46 70 120 141 160
```

# Dating: Seatbelt data

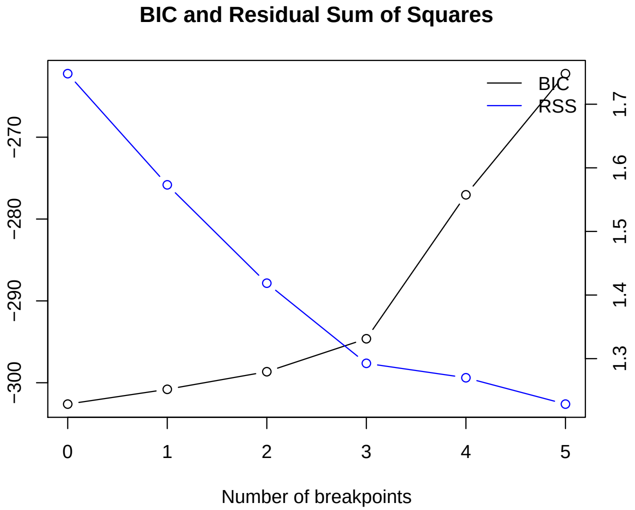
```
Corresponding to breakdates:

m = 1   1973(10)
m = 2   1973(10)                                  1983(1)
m = 3   1973(10) 1975(10)                         1983(1)
m = 4   1973(10) 1975(10) 1978(12)                1983(1)
m = 5   1973(10) 1975(10) 1979(12) 1981(9) 1983(4)

Fit:

m   0        1        2        3        4        5
RSS    1.748    1.573    1.419    1.293    1.270    1.229
BIC -302.609 -300.802 -298.652 -294.626 -277.039 -262.236
```

```
R> coef(bp, breaks = 2)
```

```
                   (Intercept)    dd1   dd12
1970(1) - 1973(10)       1.458 0.1173 0.6945
1973(11) - 1983(1)       1.534 0.2182 0.5723
1983(2) - 1984(12)       1.687 0.5486 0.2142
```

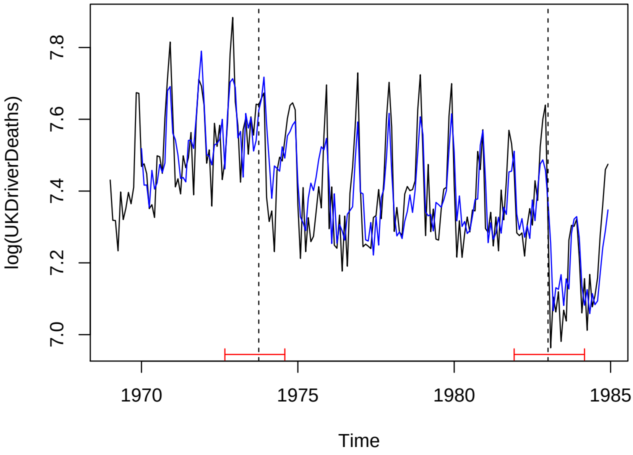
# Dating: Seatbelt data

```
R> plot(bp)
```



# Dating: Seatbelt data

```
R> plot(log(UKDriverDeaths))
R> lines(fitted(bp, breaks = 2), col = 4)
R> lines(confint(bp, breaks = 2))
```



# Beyond the linear regression model

**Question:** Why all this fuzz about object orientation?

**Answer:** Many possible models of interest (e.g., GLMs or other ML models). Avoid recoding of workhorse functions.

**Example:** Cross-section data fitted by ML model. Assess parameter stability along ordering by a numeric covariate.

**Here:** Bradley-Terry model for paired comparison data.

# Topmodel data






**Questions:** Which of these women is more attractive? How does the answer depend on the viewer's age? (And gender and the familiarity with the associated TV show Germany's Next Topmodel?)

# Topmodel data

**Data:** Paired comparisons of attractiveness from 192 survey participants for *Germany's Next Topmodel 2007* finalists: Barbara, Anni, Hana, Fiona, Mandy, Anja.

**Model:** Bradley-Terry paired comparison $P(i > j) = a_i/(a_i + a_j)$.

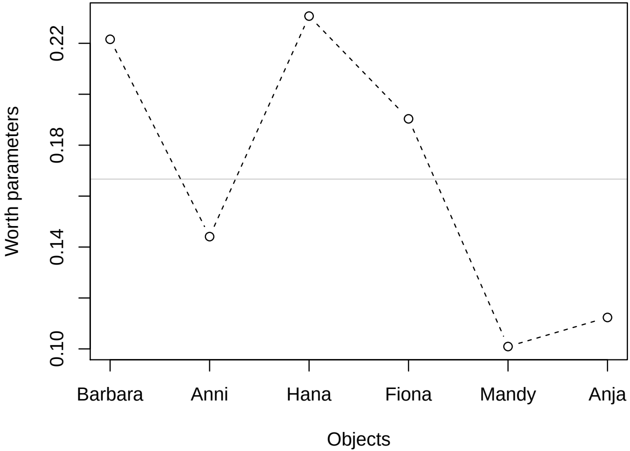
**Task:** Assess stability of attractiveness parameters from Bradley-Terry model along the age of the respondents.

**In R:** Load data, break ties randomly, set up simple formula interface.

```
R> library("psychotree")
R> data("Topmodel2007", package = "psychotree")
R> set.seed(2007)
R> tm <- transform(Topmodel2007,
+    age2 = age + runif(length(age), -0.1, 0.1))
R> names(tm)[1] <- "pref"
R> bt <- function(formula, data, ...)
+    btReg.fit(model.response(model.frame(formula, data, ...)))
```

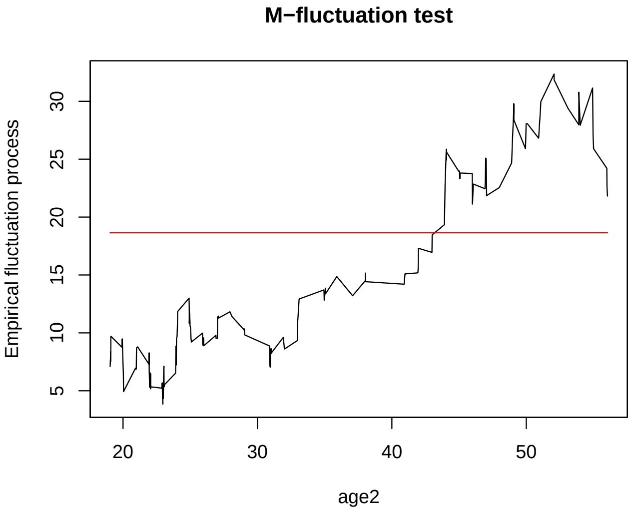
# Topmodel data

```
R> m <- bt(pref ~ 1, data = tm)
R> plot(m)
```



# Topmodel data

```
R> scus <- gefp(pref ~ 1, data = tm, fit = bt, order.by = ~ age2)
R> plot(scus, functional = supLM(0.1))
```

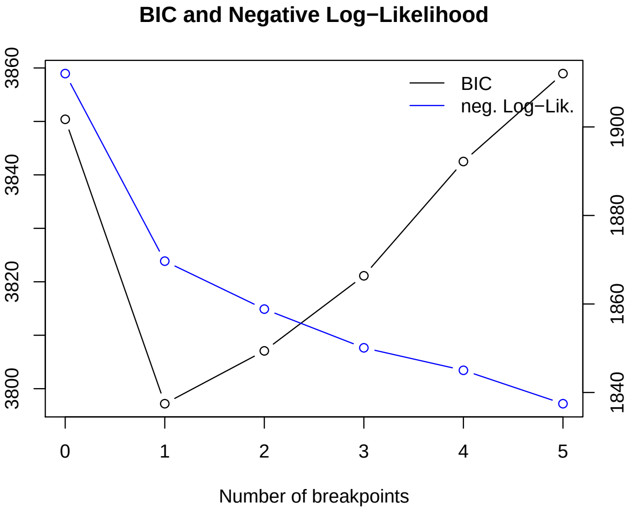


# Topmodel data

```
R> sctest(scus, functional = supLM(0.1))

M-fluctuation test

data:  scus
f(efp) = 32.36, p-value = 0.0001607

R> gbp <- fxregime::gbreakpoints(pref ~ 1, data = tm,
+    fit = bt, order.by = tm$age2, ic = "BIC")
R> breakpoints(gbp)

 Optimal 2-segment partition for `bt' fit:

Call:
breakpoints.gbreakpointsfull(obj = gbp)

Breakpoints at observation number:
161

Corresponding to breakdates:
52.0700112714432
```

# Topmodel data

```
R> plot(gbp)
```



# Topmodel data

**Segmented model:** Manually refit the Bradley-Terry model for each segment.

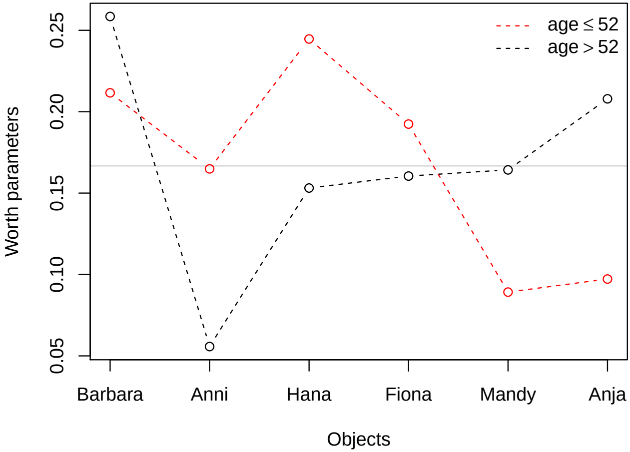
```
R> m1 <- bt(pref ~ 1, data = tm, subset = age <= 52)
R> m2 <- bt(pref ~ 1, data = tm, subset = age >  52)
```

**Alternatively:** Recursively repeat the procedure in each segment. Include further covariates gender and three questions (yes/no) that assess familiarity with the TV show.

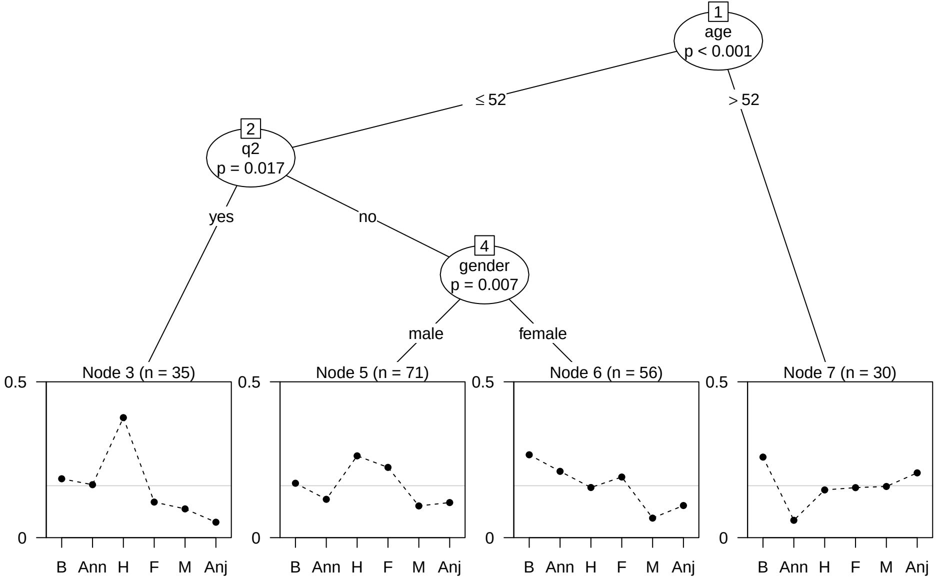
```
R> mb <- bttree(preference ~ gender + age + q1 + q2 + q3,
+    data = Topmodel2007)
```

# Topmodel data

```
R> plot(m2)
R> lines(worth(m1), col = 2, lty = 2, type = "b")
R> legend("topright", legend = c(expression(age <= 52),
+     expression(age > 52)), lty = 2, col = 2:1, bty = "n")
```



# Topmodel data



# Topmodel data

```
R> sctest(mb, node = 1)

            gender       age      q1        q2     q3
statistic 17.08798 3.236e+01 12.6320 19.839222 6.7586
p.value    0.02168 7.915e-04  0.1283  0.006698 0.7452

R> sctest(mb, node = 7)

          gender    age     q1 q2     q3
statistic 3.3498 7.8686 8.0524  0 4.7728
p.value   0.9843 0.9593 0.4862 NA 0.9046
```

# Challenges/wishlist

**Basic building blocks:**

- Distributions (`p`/`q` functions) for functionals of (multivariate) Brownian motions/bridges.
- Faster optimizers for (penalized) additive objective functions.

**Object orientation:**

- More infrastructure for general orderings (in particular "`zoo`", "`xts`", etc.).
- More tests, e.g., LR- or Wald-based tests.
- Sequential monitoring techniques.
- Better interface to dating algorithm.

# Summary

- Extensive toolbox for testing, monitoring, and dating structural changes in linear regression models.
- Object-oriented implementation of score-based structural change tests for general models and arbitrary orderings.
- Emphasis on visualization along with formal modeling.
- Capture workflow by suite of methods to generic functions.
- More object-oriented tools desirable for general models, especially monitoring and (better) dating functions.

# Summary: ***strucchange***

Classical structural change tools for OLS regression:

- Time ordering: Regular (via “`ts`”).
- Testing: `efp()`, `Fstats()`, `sctest()`.
- Monitoring: `mefp()`, `monitor()`.
- Dating: `breakpoints()`.
- Vignette: `"strucchange-intro"`.

Object-oriented structural change tools:

- Time ordering: Arbitrary (via “`zoo`”).
- Testing: `gefp()`, `efpFunctional()`.
- Monitoring: Still to do.
- Dating: Some currently unexported support in `gbreakpoints()` in *fxregime*.
- Vignette: None, but CSDA paper.

# Summary: *fxregime*

Structural change tools for Gaussian regression estimated by (quasi-)ML, specifically for exchange rate regression:

- Time ordering: “`zoo`”.
- Data: `FXRatesCHF` (“`zoo`” series with US Federal Reserve exchange rates in CHF for various currencies).
- Preprocessing: `fxreturns()`.
- Model fitting: `fxlm()`.
- Testing: `gefp()` from *strucchange*.
- Monitoring: `fxmonitor()`.
- Dating: `fxregimes()` based on currently unexported `gbreakpoints()`; `refit()` method for fitting segmented regression.
- Vignettes: `"CNY"`, `"INR"`.

# References: Methods


Zeileis A, Shah A, Patnaik I (2010). "Testing, Monitoring, and Dating Structural Changes in Exchange Rate Regimes." *Computational Statistics & Data Analysis*, **54**(6), 1696–1706. `doi:10.1016/j.csda.2009.12.005`.

Zeileis A, Hothorn T, Hornik K (2008). "Model-Based Recursive Partitioning." *Journal of Computational and Graphical Statistics*, **17**(2), 492–514. `doi:10.1198/106186008X319331`

Zeileis A, Hornik K (2007). "Generalized M-Fluctuation Tests for Parameter Instability." *Statistica Neerlandica*, **61**(4), 488–508. `doi:10.1111/j.1467-9574.2007.00371.x`

Zeileis A (2005). "A Unified Approach to Structural Change Tests Based on ML Scores, *F* Statistics, and OLS Residuals." *Econometric Reviews*, **24**(4), 445–466. `doi:10.1080/07474930500406053`

Zeileis A, Leisch F, Kleiber C, Hornik K (2005). "Monitoring Structural Change in Dynamic Econometric Models." *Journal of Applied Econometrics*, **20**(1), 99–121. `doi:10.1002/jae.776`

# References: Software


Zeileis A (2006). "Implementing a Class of Structural Change Tests: An Econometric Computing Approach". *Computational Statistics & Data Analysis*, **50**(11), 2987–3008. `doi:10.1016/j.csda.2005.07.001`

Zeileis A (2006). "Object-Oriented Computation of Sandwich Estimators." *Journal of Statistical Software*, **16**(9), 1–16. URL `http://www.jstatsoft.org/v16/i09/`.

Zeileis A, Grothendieck G (2005). "*zoo*: S3 Infrastructure for Regular and Irregular Time Series." *Journal of Statistical Software*, **14**(6), 1–27. URL `http://www.jstatsoft.org/v14/i06/`.

Zeileis A, Kleiber C, Krämer W, Hornik K, (2003). "Testing and Dating of Structural Changes in Practice." *Computational Statistics & Data Analysis*, **44**(1–2), 109–123. `doi:10.1016/S0167-9473(03)00030-6`

Zeileis A, Leisch F, Hornik K, Kleiber C (2002). "*strucchange*: An R Package for Testing for Structural Change in Linear Regression Models." *Journal of Statistical Software*, **7**(2), 1–38. URL `http://www.jstatsoft.org/v07/i02/`.