JOURNAL OF REPRODUCIBLE STATISTICS
FAIR Press Journals · jrepstat.fairpressjournals.com · ISSN 3043-2790
Vol. 1 (2026), Art. 12
10.68365/jrepstat.2026.12

SOFTWARE · FOUNDING ARTICLE

Fitting segmented mixed models with random changepoints in R

Vito M.R. Muggeo*

Dip. Scienze Economiche, Aziendali & Statistiche - Universita' di Palermo, Italy
*Corresponding author: vito.muggeo@unipa.it

ARTICLE INFO

Keywords
segmented regression
mixed models
random changepoints
R package
piecewise-linear models
longitudinal data

History
Received 1 Oct 2026
Final version 18 Nov 2026
Accepted 18 Nov 2026
Version 1

Handling editor
Matthias Templ

ABSTRACT

We illustrate and discuss R code to fit segmented mixed models with random changepoints in a likelihood based framework. Some examples are presented to show capability of the methodology and of the relevant functions in the R package segmented.

Reproduced verified 2026-11-18 REPRODUCIBILITY RECORD

Result	Reported results reproduced. Table 1 reproduces exactly from 300 replicates (B set to 300). Examples 1 and 2 agree, four printed values differ in their last digit.	✓ *measured*
Entry point	source("Rcode.R"), no command. The author's code is `code/Rcode.R` in the package.	*declared*
Environment	macOS arm64 native, R 4.5.2, segmented 2.3-0 from the author's tarball, 27 minutes	✓ *measured*
Determinism	Seeded.	*declared*
Archive	`fitting-segmented-mixed-models-with-random-changepoints-in-r-v1-repro.zip` · reviewed bundle SHA-256 `ab8da100777d1cbc5d6cff97a99a5f5d95367079bb4de738889c154c7ff5314e`	✓ *computed*
AI use	None.	*declared*

1 Introduction

In segmented regression, the effect of a numeric covariate is expressed via a piecewise linear relationship, namely two straight lines with different slopes connected at some unknown value within the covariate range, the so-called breakpoint (Hardy, 2001; Muggeo, 2003). *Segmented mixed models* extend simple segmented regression to the mixed model framework by putting random effects in each model parameter, including the breakpoint (Cudeck & Klebe, 2002; Morrell et al., 1995; Muggeo et al., 2014). Random effects are popular to model longitudinal or clustered data when repeated measurements for each unit have been collected, see Fitzmaurice et al. (2004) for instance.

Assuming numeric data y_{ij}, with observational times t_{ij} ($j = 1, ..., n_i$ and $i = 1, ..., N$), the model reads as

CITE AS Muggeo, V. M. R. (2026). Fitting segmented mixed models with random changepoints in R. Journal of Reproducible Statistics, 1, 12. https://doi.org/10.68365/jrepstat.2026.12

$$y_{ij} = \beta_{0i} + \beta_{1i} t_{ij} + \delta_i (t_{ij} - \psi_i)_+ + \epsilon_{ij}, \tag{1}$$

where the ϵ_{ij} are the usual zero-mean stochastic component. Note that, according to parametrization above, the left and right slopes are β_{1i} and $\beta_{1i} + \delta_i$ respectively. Each parameter of model ((1)) is given by the sum of fixed effects, possibly depending on covariates, and the random component. Thus for the intercept and the left slope it is possible to write

$$\beta_{0i} = \beta_{00} + x_{0i}^T \beta_0 + b_{0i} \qquad \beta_{1i} = \beta_{10} + x_{1i}^T \beta_1 + b_{1i}, \tag{2}$$

while for the slope-difference and the changepoint parameters we have

$$\delta_i = \delta_0 + x_{di}^T \delta + d_i \qquad \psi_i = \kappa_0 + z_i^T \kappa + k_i. \tag{3}$$

The x_0, x_1, x_d and z are, possibly overlapped, covariate vectors describing the subject features or experimental conditions, i.e. with values fixed within the same subject. Note that for the changepoint parameter, an alternative parameterization is $\psi_i = \frac{a_1 + a_2 \exp\{\kappa_0 + z_i^T \kappa + k_i\}}{1 + \exp\{\kappa_0 + z_i^T \kappa + k_i\}}$, where $a_1 = min\{t_{ij}\}$ and $a_2 = max\{t_{ij}\}$. This logistic parameterization guarantees values of ψ_i always in the admissible range (a_1, a_2).

As usual, a multinormal distribution with zero mean and specified covariance matrix, is assumed for the random effects, i.e.

$$\begin{bmatrix} b_{0i} \\ b_{1i} \\ d_i \\ k_i \end{bmatrix} \sim \mathcal{N}_4 \left(\left[\begin{pmatrix} \text{JRSAUG h= v= a=c 0} \\ 0 \\ 0 \\ 0 \end{pmatrix} \right], \begin{bmatrix} \sigma^2_{b_0} & . & . & . \\ \sigma_{b_0 b_1} & \sigma^2_{b_1} & . & . \\ \sigma_{b_0 d} & \sigma_{b_1 d} & \sigma^2_d & . \\ \sigma_{b_0 k} & \sigma_{b_1 k} & \sigma_{dk} & \sigma^2_k \end{bmatrix} \right). \tag{4}$$

The model has been discussed by several authors, and estimation has been illustrated mostly in a Bayesian paradigm, see Lange et al. (1992); Kiuchi et al. (1995); Jacqmin-Gadda et al. (2006). Muggeo et al. (2014) proposed estimation in a likelihood framework, i.e. without need to specify any prior for parameters.

We illustrate how to fit the model and to display the relevant results via specific functions of the R package segmented (Muggeo, 2008), version 2.3-0. Full information on the operating system, the R version, and on additional packages used, is reported at the end of section 3.

The rest of the paper is organized as follows: section 2 introduces the functions by describing their main arguments, and section 3 reports some analyses on real-data as well as on a simulated dataset. Finally, section 4 is devoted to conclusions and discussion.

2 Structure of the functions

The main function is `segmented.lme` shipped within package segmented for R (R Core Team, 2026), with main arguments

```
segmented.lme(obj, seg.Z, z.psi =~1, x.diff =~1, random,
              psi.link = c("identity", "logit"))
```

`obj` is a classical '`lme`' object returned by `lme()` from the nlme package (Pinheiro & Bates, 2000) that is automatically loaded along with segmented; `seg.Z` is one-sided formula to specify the 'segmented' covariate having the piecewise linear relationship with the response, namely the t_{ij} values in equation ((1)). `x.diff` and `z.psi` are one-sided formulas to indicate the possible covariates for the slope difference δ_i and for the breakpoint ψ_i, i.e. respectively the x_d's and the z's in equations ((3)). Both of them default to `~1`, meaning no covariate, namely $\delta = 0$ and $\kappa = 0$. Dependence on covariates of the linear parameters β_{0i} and β_{1i} can be modelled via appropriate main effects or interaction terms to be included in the starting `obj`. `psi.link` specifies which sub-regression model should be employed for the breakpoint, linear or logistic, as discussed in the previous section.

`random` allows to set the random structure of the model, namely the covariance matrix in equation ((4)). It should be a list with a formula for the clustering variable, typically the subject identifier. The formula can include any linear variable in the model and, more importantly, the specific terms `U` and `G0` to set random effects, respectively for the difference-in-slope (the d_i's), and for breakpoint (the k_i's). Some possible specifications of the random effects covariance in ((4)) using the constructors of positive-definite matrix classes in the package `nlme`, are reported below. We refer to model ((1)), with the t_{ij} values represented by the variable `time`, and clustering variable `id`.

i `random=list(id=pdDiag(~1+time+U+G0))`: independent random effects in each model parameter, namely the covariance matrix is diagonal;
ii `random=list(id=pdBlocked(list(pdDiag(~1+time), pdSymm(~U+G0-1))))`: a block diagonal covariance matrix, where the second block assumes correlated random effects, namely all covariances set to zero but σ_{dk};
iii `random=list(id=pdBlocked(list(pdSymm(~1+time), pdSymm(~U+G0-1))))`: a block diagonal covariance matrix, where both the blocks assume correlated random effects, namely all covariances set to zero but σ_{dk} and $\sigma_{b_0 b_1}$;
iv `random=list(id=pdSymm(~1+time+U+G0))`: non-independent random effects, i.e. the full covariance matrix is estimated.

The four specifications above are listed according their complexity: the first one is the simplest and, probably, it could represent the first option when one is interested in assessing if heterogeneity is supported by data. For this very reason, `segmented.lme()` simplifies that by also accepting in `random` a simple formula; in other words, `random=id~time+U+G0` will set independent random effects without specifying `pdDiag`. In item iv, we want to estimate an unstructured covariance matrix with 4 variances and 6 covariances. Of course, the more complex the model, the better the fit and corresponding predictions, but also the heavier the corresponding computational burden which could make model estimation more demanding: namely overfitting with unsuccessful fits. Roughly speaking, if the model being fitted is not supported by data (for instance unnecessary covariances or even variances), troubles can arise in reaching convergence. This is true also in linear mixed models, but it is enhanced in nonlinear mixed models and in the segmented, in particular, see Example 2 in sec 3.2.

The object resulting from `segmented.lme()` is a list of class 'segmented.lme', with some related methods, such as `print.segmented.lme`, `summary.segmented.lme`, `vcov.segmented.lme`, and the plot method `plot.segmented.lme`. In particular, the 'plot' method can be used to produce graphics of the resulting model, namely the fitted segmented lines along with observations. The function with its main arguments is

```
plot.segmented.lme(x, level=1, id = NULL, res = TRUE, pop = FALSE,
           yscale = 1, xscale=1,  n.plot, by=NULL, ...)
```

`x` is the fit object returned by `segmented.lme()`, and `level` refers to the level of grouping, with level zero corresponding to the population fitted values; `id` specifies which profiles should be drawn (default to `NULL` meaning all profiles); `res=TRUE` indicates if the plot should also include the observations; finally `pop=TRUE` adds the population (i.e. based on fixed effects only) estimates. The remaining arguments affect the appearance. `yscale` and `xscale` regulate the y and the x-scale of the figures: if negative values are supplied, 'individual', i.e. subject-specific limits are used, otherwise the common limits are obtained by considering the ranges of all subjects. Finally `n.plot` is a vector to be passed to `mfrow` when multiple plots have to be produced.

If the fixed-effect (i.e. `level=0`) relationship depends on some covariates (included via interactions in the starting lme fit or via the arguments `x.diff` and/or `z.psi` of `segmented.lme`), the argument `by` allows to specify for which covariate pattern the relationships has to be drawn, see examples in the next section.

3 Fitting the model in practice

Broadly speaking, maximization of log marginal likelihood is quite a difficult task in nonlinear mixed models, even under the standard Gaussian assumptions for the ϵ's in ((1)) and for the random effects in ((4)). The load is enhanced in segmented mixed models as the breakpoints make the model non regular. The approach implemented in `segmented.lme`, based on the aforementioned Gaussian assumptions, is discussed in detail in Muggeo et al. (2014) and relies on iterative fitting of simple linear mixed models via the `lme()` function of the `nlme` package. The algorithm is quite efficient but, due to possible local optima, bootstrap restarting (Wood, 2001) is also implemented to escape the local solutions where the algorithm can get stuck.

3.1 *Example 1: real data*

The dataframe `glucoseData` shipped with the package includes data on $n = 33$ patients (in two groups, Control and Obese) who took a standard glucose tolerance test. The goal was to assess the pattern of phosphate elimination (`phos`) with respect to time after introduction: therefore the response measurements are plasma inorganic phosphate (inversely related to glucose level) taken at the beginning and 0.5, 1, 1.5, 2, 3, 4, and 5 hours after introduction. Data come from Zerbe (1979) and have been analysed also in Cudeck & Klebe (2002).

A first look to data can be provided by scatterplots for each subject,

```
> plot(groupedData(phos~time|subj,data=glucoseData)) #plot not shown
```

The Phosphate level decreases immediately after administration and recovers subsequently, which suggests that a segmented mixed models could be a reasonable tool for the analysis. To fit the model, we first fit the starting linear mixed model via `lme()`, and then add the piecewise-linear relationship via `segmented.lme()`, which does not need to be called explicitly as it is a *method* function in the R philosophy. Hence

```
> o <-lme(phos~time, random=~1|subj, data=glucoseData)
> os <-segmented(o, ~time, random=subj~1+time+U+G0)
```

With respect to the starting lme object, `segmented.lme()` updates the fixed part formula by adding the slope-difference term along with the breakpoint, and replaces the `random` statement by the new one; as discussed in the previous section, here we use a simple formula to set independent random effects. When implementing the bootstrap restarting, the bootstrap samples are randomly drawn, leading to select different starting values at each run; however, the global optimum is evaluated on the original dataset, and therefore the final solution is expected to be independent of the bootstrap runs, at least when data are not excessively noisy. However, to make fully replicable the analysis on the same data, including the 'path' driving to the final solution, `segmented.lme()` uses a seed coming from the response values; as an alternative, it is also possible to set it explicitly in the call via `seg.control()`, namely `segmented(.., control=seg.control(seed=..))`.

The print method returns some basic information, just like the print method for lme objects, namely point estimates for the fixed effects and the variance components

```
> os
Segmented linear mixed-effects model fit by REML
  Data: glucoseData
  Log-restricted-likelihood (approx): -155.9955
  Bootstrap restarting on 6 samples; 1 different solution(s)

Fixed: phos ~ time + U + G0
(Intercept)        time           U          G0
  4.3050486  -0.8483528   1.2048822   1.6624618
  psi.link = identity
```

```
Random effects:
 Formula: ~time + U + G0 | subj
 Structure: Diagonal
        (Intercept)       time           U        G0 Residual
StdDev:    0.675425 0.00189322 0.003314052 0.4340041 0.365604

Number of Observations: 264
Number of Groups: 33
```

The fixed effects correspond to estimates of the intercept, left slope and slope-difference (coefficient of `U`), while the coefficient of `G0` is the changepoint estimate (i.e. $\hat{\psi} = 1.662$). The high standard deviations for the intercepts and `G0` underline the high heterogeneity of the subjects with respect to their intercepts and changepoints.

The first output of the print method (and also of summary, see later) is the log (marginal) likelihood value which comes from an approximation, as illustrated in Muggeo et al. (2014). Model fitting via maximization of the actual likelihood requires heavy computations and numerical integration that could lead to a considerable computational burden. Simulation studies presented in Muggeo et al. (2014) and in the end of section 3, show that such approximation works reasonable well in practice, while maintaining an acceptable execution times.

According to results of model `os`, all subjects share the same V-shaped pattern (quite low $\hat{\sigma}_{b_1}$ and $\hat{\sigma}_d$), although their changepoints are different. However, data refer to Control and Obese patients, therefore it could be interesting to assess whether the two V-shaped relationships are the same or not in the two groups. Therefore, we include the term `group` as main effect and interaction term in the starting lme fit, and also in the arguments `x.diff` and `z.psi`,

```
> o1<-lme(phos~time*group, random=~1|subj, data=glucoseData)
> os1<-segmented(o1, ~time, z.psi=~group, x.diff=~group,
+                random=subj~1+time+U+G0)
```

When there are interaction terms, the `summary.lme()` returns results by assembling the estimates of each parameter of the regression equation ((1)),

```
> summary(os1)
Segmented mixed-effects model fit by REML
       AIC      BIC    logLik
  327.5191 373.7076 -150.7596
 Bootstrap restarting on 6 samples;  1 different solution(s)

Random effects:
 Formula: ~time + U + G0 | subj
 Structure: Diagonal
        (Intercept)       time           U        G0  Residual
StdDev:   0.5619773 0.05488536 0.003517548 0.2958991 0.3511334

Fixed effects:
                  Value Std.Error  DF t-value p-value
(Intercept)      4.0349    0.1788 225  22.567 0.00000
groupObese       0.5113    0.2271  31   2.252 0.03155
-- leftS:
time            -1.2796    0.1415 225  -9.045 0.00000
time:groupObese  0.5460    0.1521 225   3.589 0.00041
-- diffS:
U                1.6481    0.1446 225  11.397
U.groupObese    -0.5350    0.1611 225  -3.322 0.00104
-- break:
G0               1.1086    0.1170 225
G.groupObese     1.0465    0.1714 225   6.104 0.00000
 psi.link = identity
```

```
Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-2.73565679 -0.55307557  0.01409987  0.63157506  3.20513447

Number of Observations: 264
Number of Groups: 33
```

As in the summary of usual lme fits, the last columns return t-values relevant to the null hypothesis of zero coefficient, and p-values relying on a Gaussian distribution of the estimator *under the null hypothesis*. It is well known that such approximation is not reliable for $\hat{\delta}_0$ (Muggeo, 2016, 2025), and therefore the term `U` misses the p-value; the corresponding t-value is displayed, as it can be used to quantify the strength of the slope change estimate regardless of the reference null distribution. For the changepoint, testing for a zero value is meaningless, and therefore both the t- and the p-value have been dropped. However, for the remaining regression coefficients including those in equations ((3)), the Normal approximation for the sampling distributions is generally understood to be valid, making the p-values acceptable from a statistical viewpoint, at least in large samples.

The `confint` method returns the interval estimates for the model parameters, using again the Wald statistic relying on a limiting Gaussian distribution and asymptotic model-based standard errors (that are returned internally by the `vcov` method)

```
> t(confint(os1))
                       2.5%      97.5%
(Intercept)      3.68444601  4.3853227
time            -1.55692116 -1.0023407
groupObese       0.06626251  0.9563422
U                1.36470061  1.9315360
G0               0.87932679  1.3378761
G.groupObese     0.71045946  1.3825110
U.groupObese    -0.85069494 -0.2193703
time:groupObese  0.24784367  0.8440827
```

By looking at the results, the curve for the Obese patients can be considered different from the Control group. More specifically, the Obese group has a higher intercept ($4.0349 + 0.5113$), a less steep left slope ($-1.2796 + 0.5460$), and a larger changepoint ($1.1086 + 1.0465$). To have estimates of the slopes and breakpoints in both groups, we use the functions `slope()` and `breakpoint()`,

```
> slope(os1) #in the baseline group
                 Est.     St.Err t value   0.95.low    0.95.up
leftSlope  -1.2796309 0.14147721  -9.045 -1.5569212 -1.0023407
rightSlope  0.3684874 0.03629992  10.151  0.2973409  0.4396339
>
> slope(os1, by=list(group="Obese"))
                 Est.     St.Err t value   0.95.low    0.95.up
leftSlope  -0.7336678 0.05585699 -13.135 -0.8431455 -0.6241901
rightSlope  0.3794180 0.04677694   8.111  0.2877368  0.4710991
```

The former call returns the slopes corresponding to the baseline values of the interaction term, i.e. `group='Control'`, while in the latter, we specify the covariate and its corresponding label of interest via argument `by` to obtain the slopes in the Obese group. Likewise, for the estimated breakpoint we type

```
> breakpoint(os1) #in the baseline group
        Est.    St.Err  0.95.low  0.95.up
psi 1.108601 0.1296831 0.8793268 1.337876
>
> breakpoint(os1, by=list(group="Obese"))
```

```
        Est.    St.Err  0.95.low  0.95.up
psi 2.155087 0.2701112  1.909431 2.400742
```

As in `confint`, the confidence intervals returned by `slope()` and `breakpoint()` are computed using the approximate Gaussian distribution of the Wald statistic. We remark such approximation is generally reliable if the sample is large, the breakpoint is understood to exist, and the segmented relationship is rather clear-cut. We present results from a small simulation study in the next section to bear this out.

Some graphical outputs could complete the analysis. For instance, to visualize the fitted segmented profiles, the method `plot` described in the previous section can be used. Thus typing

```
> plot(os1, n.plot=c(5,7), res=TRUE,
+      l.col=rep(c(4,2), c(13,20)), l.lty=rep(c(1,2), c(13,20)),
+      ylab="Phosphate level (mg/dl)", xlab="Hours after challenge")
```

produces the output of Figure 1. Dots are the observed data (via `res=TRUE`), and the lines portray the individual profiles (given by the sum of fixed effects and random predictions), while symbols on the bottom locate the individual changepoints. In this example there $n = 33$ subjects, so we decide to put the individual panels into 5 rows and 7 columns (`n.plot=c(5,7)`), and to use the blue and continuous line to display the first 13 subjects (of the Control group), and the red and dashed line for the 20 individuals of the Obese group. We specify such color and line type vectors in the argument `l.col` and `l.lty`.

Finally, for illustrative purposes, we also report additional plots in Figure 2 by running the following code

```
> par(mfrow=c(1,2))
> #left panel (population estimates)
> plot(os1, level=0, vline=TRUE, ylim=c(1,6), p.lwd=4, p.col=4,
+      pos.leg="topleft", text.leg = "Control" )
> plot(os1, level=0, by=list(group="Obese"), add=TRUE, p.col=2, p.lty=2,
+      pos.leg="top", text.leg = "Obese" ,vline=TRUE, p.lwd=5)
> #right panel
> plot(os1, n.plot=1, l.col=2:6, ylim=c(1,6), dens.col=grey(.5), res=FALSE)
```

In the left panel we first plot the baseline population segmented relationship in the baseline (Control) group, then we add the Obese group estimate via arguments `add` and `by`; `p.col` and `p.lwd` are used to set width and color of the segmented lines. On the right panel, we display the subject-specific profiles along with a kernel density of the individual breakpoints on the bottom. Unlike the previous plot in Figure 1, we do not specify the axes labels, but we let the function to take the variable names, as happen in the other plot functions in R.

By looking at the variance estimates of model `os1`, we note that the heterogeneity in the slope difference is quite low, $\hat{\sigma}_d = 0.0035$. While the estimation process did not return any error or warning message, we could assess whether the fit could be simplified by removing it in the interest of the parsimony principle. However, in the same perspective as model selection, we should also assess if data support a more complex model, for instance by also estimating $\sigma_{b_0 b_1}$ and σ_{dk}, for instance. To this end, we exploit the usual `update()` function in R, to change just a single argument value with respect to `os1`,

```
> os2<-update(os1, random=subj~time+G0)
> os3<-update(os1, random=list(subj=
+             pdBlocked(list(pdSymm(~1+time), pdSymm(~U+G0-1)))))
> AIC(os1,os2,os3)
    df      AIC
os1 13 327.5191
os2 12 325.5192
os3 15 327.1868
```

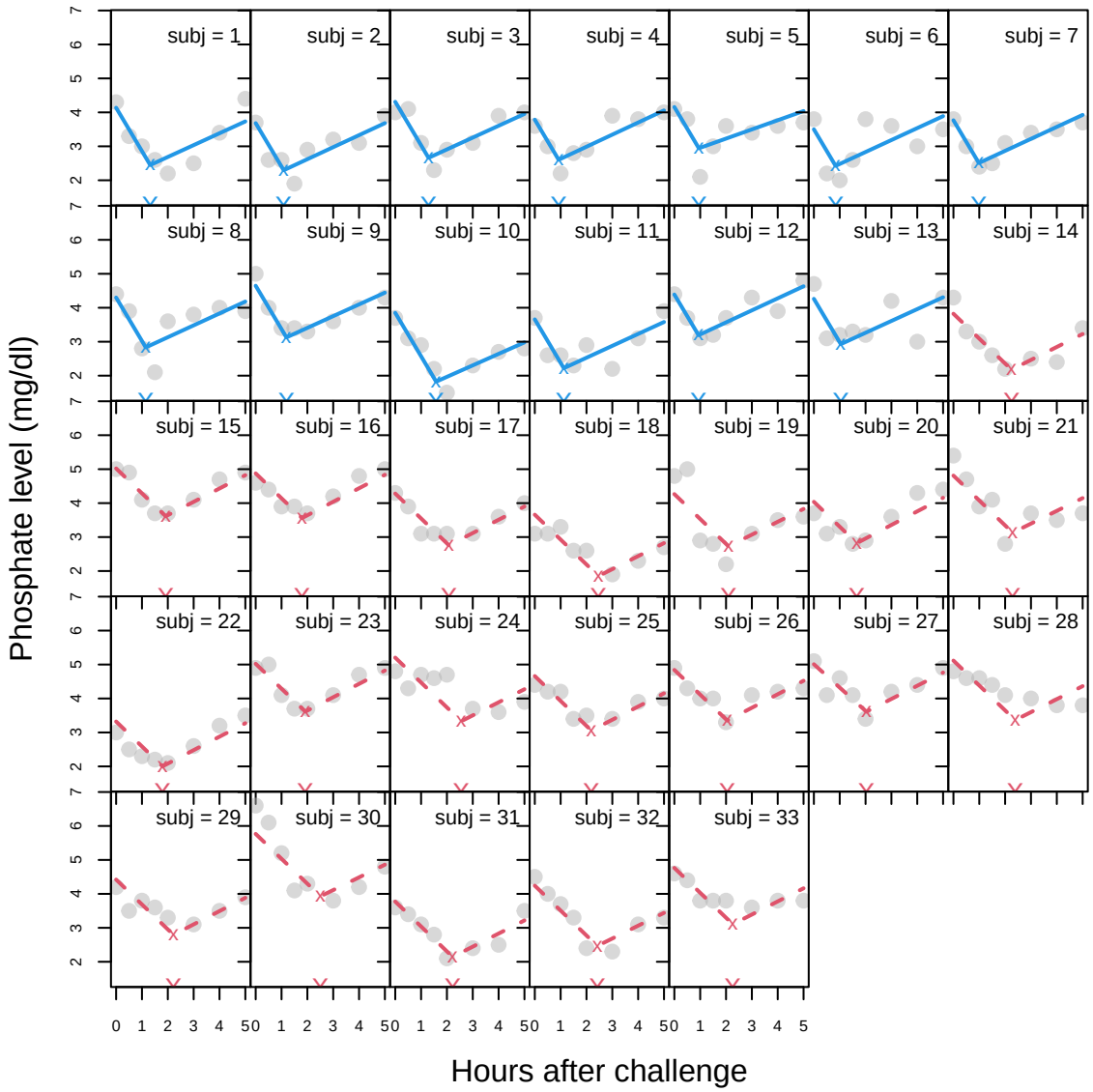

Figure 1 Output of `plot.segmented.lme(..)` to display the individual profiles. The continuous blue and the dashed red lines refer to the Control and Obese groups respectively.

While the AIC values (and the BIC as well) suggest that `os2` could be preferred, it should be remarked that model selection is a hot and open research topic in mixed models. We do not pursue the topic further, as it is beyond the scope of this paper.

3.2 *Example 2: simulated data*

As a further example we use simulated data. The aim is twofold: to illustrate the code for somewhat more complicated modelling, and to 'assess' the results reliability by comparing them with the true values. We discuss results for a single dataset and then across several datasets in a small simulation study at the end of section.

We simulate data by means of the function `sim.seg()` that, with its *main* arguments, reads as

```
> args(sim.seg)
  function (N, ni, fixEff = c(2, -0.2, 0.3, 0.4),
  S = diag(c(0.05, 0.05, 0.05, 0.07)^2), seg.Z = NULL, z.psi, x.diff,
  sd.eps = 0.01, seed)
```

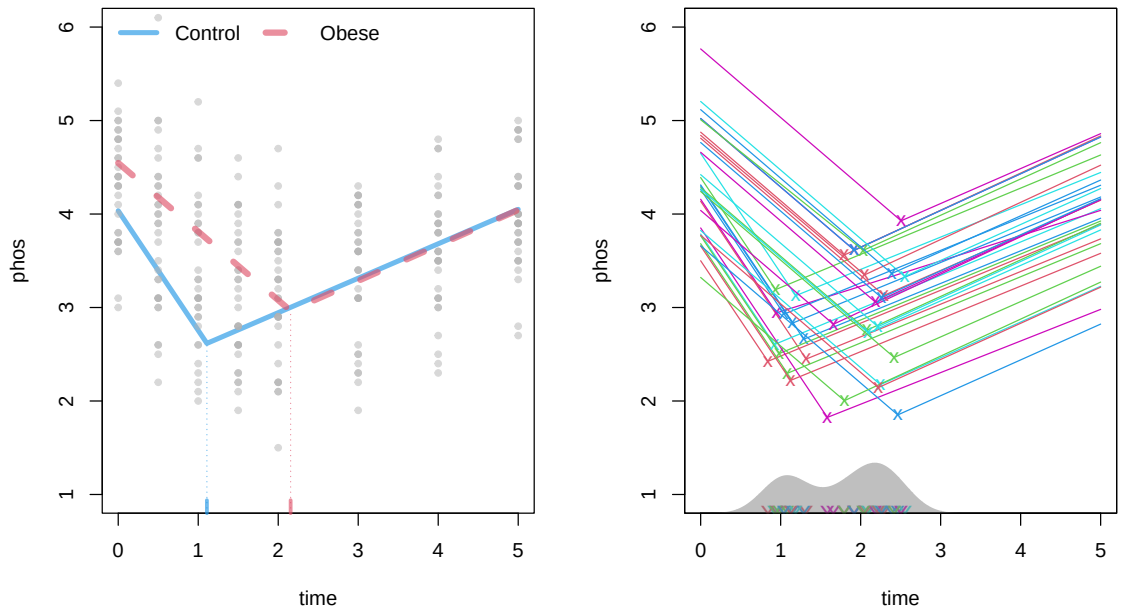

Figure 2 Additional output of `plot.segmented.lme(..)`. Marginal profiles in the left panel, and subject-specific predictions in the right. On the bottom, the nonparametric density estimate of the individual breakpoints is optionally reported (via the `dens.col`): the bimodality comes from having different fixed-effect breakpoint estimates in two groups. `time` in the x-axis refers hours after challenge and `phos` on the y-axis to phosphate level (mg/dl).

`N` and `ni` are the number of subjects and the number of measurements per subject, `fixEff` indicates the 4 fixed parameters $(\beta_0, \beta_1, \delta_0, \kappa_0)$, `S` the covariance matrix in ((4)), `seg.Z` the segmented covariate, and `x.diff` and `z.psi` the covariates and corresponding coefficients for the sub-regression models for δ_i and ψ_i using a linear link. Both arguments are specified via a list with coefficient(s) and covariate values, see the example below. It is assumed that such variables are subject-specific, so N values have to be provided for each variable: a N-dimensional vector for a single covariate (which needs a single coefficient), and a $N \times p$ matrix for p covariates (and p coefficients have to be specified). We set the seeds just to reproduce the results

```
> N=30
> set.seed(123)
> Xd=runif(N, 0,2) #1 covariate for delta (N-vector)
> Z=cbind(1:N/N, rnorm(N)) #2 covariates for psi (N x 2 matrix)
> dd=sim.seg(N, ni=12, S=diag(c(.12,.05,.3,.1)^2),
+          x.diff=list(.5,Xd), z.psi=list(c(.3,-.2),Z), sd.eps=.003, seed=15)
>
> d=dd$data
>
> head(d,3)
  id        y          x     Xdif      Zpsi1    Zpsi2
1  1 1.931448 0.06693461 0.575155 0.03333333 1.786913
2  1 1.932327 0.09619888 0.575155 0.03333333 1.786913
3  1 1.934428 0.10839553 0.575155 0.03333333 1.786913
```

In the simulated dataset, `Xdif` is the covariate for the slope change parameter, and `Zpsi1` and `Zpsi2` affect the changepoint.

To fit the segmented model, we can avoid to fit a lme object beforehand and specify the fixed effect formula directly in `segmented.lme()` which has to be called explicitly; we also need to indicate the dataframe where the variables are stored

```
> oseg1 <- segmented.lme(y~x, ~x, x.diff=~Xdif, z.psi=~Zpsi1+Zpsi2,
+                        random=id~1+x+U+G0, data=d)
```

Out of curiosity, we can compare the true fixed effect versus the corresponding estimates extracted by `fixef()`,

```
> cbind(true=c(2,-0.2,0.3,0.4,0.3,-0.2,0.5), est=fixef(oseg1))
            true        est
(Intercept)  2.0  2.0457565
x           -0.2 -0.1990579
U            0.3  0.2526996
G0           0.4  0.3547105
G.Zpsi1      0.3  0.3545174
G.Zpsi2     -0.2 -0.1827813
U.Xdif       0.5  0.5980579
```

while for the random effects, we take the main diagonal of the covariance matrix extracted by `getVarCov()`

```
> cbind(true=c(0.12,0.05,0.30,0.10), est=sqrt(diag(getVarCov(oseg1))))
            true        est
(Intercept) 0.12 0.11410761
x           0.05 0.05667073
U           0.30 0.25146646
G0          0.10 0.10089096
```

As in the previous example, to compute correctly the slopes and breakpoints accounting for the interaction variables, we need to specify the covariate pattern in the argument `by`. For instance, for the breakpoint depending on `Zpsi1` and `Zpsi2`, and assuming 0.5 and -1.2 are two values of interest, we type

```
> breakpoint(oseg1, by=list(Zpsi1=.5,Zpsi2=-1.2))
         Est.     St.Err  0.95.low   0.95.up
psi 0.7513068 0.02449226 0.6874128 0.8152007
```

`oseg1$psi.i` includes the predicted N changepoints with two attributes: `"ni"` the number of measurements per subject, and `"is.break"` a logical vector indicating whether the individual changepoint is within the covariate range or not. Possible `FALSE` values identify subjects wherein the changepoint has been predicted on the boundary, and whose corresponding profiles are estimated to be linear. Thus

```
> which(!attr(oseg1$psi.i, "is.break"))
 4 28
 4 28
```

Hence, the model identifies two subjects with a fitted linear profile, and it is worth checking if such 'predictions' are correct. In order to do that, note the object returned by `sim.seg()` also includes the actual breakpoints (`dd$psi.i`), and its percentile (`dd$perc.psi`) with respect to corresponding covariate ranges. Clearly, if the percentile is 0 or 1, the breakpoint is on boundary, leading to data simulated according to underlying linear profiles. We have

```
> which(dd$perc.psi%in%c(0,1))
[1]  4 28
```

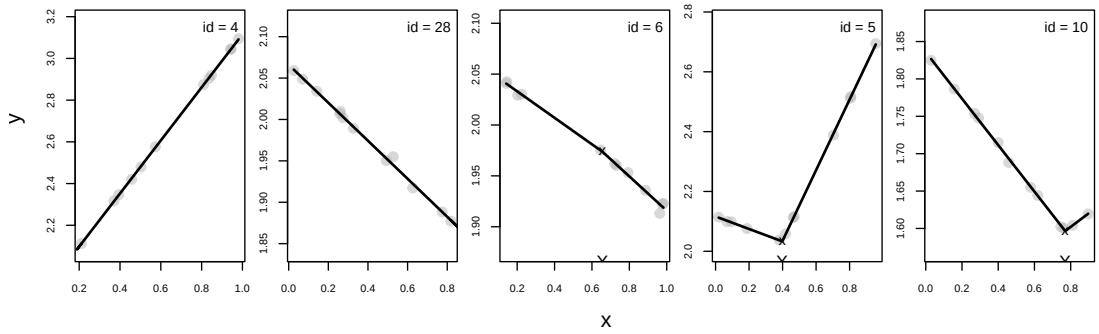

Figure 3 Output for 5 subjects on their individual x and y scales (via `xscale=-1` and `yscale=-1`).

Thus, we can claim predictions are correct to detect subjects with no change. We display their predicted linear profiles along with some additional segmented profiles (the 6th, the 5th and the 10th, say) using their specific x- and y-scales,

```
> plot(oseg1, id=c(4,28,6,5,10),n.plot=c(1,5), yscale=-1,xscale=-1)
```

In addition to the aforementioned linear profiles for subjects 4 and 28, it is worth stressing the fitted line for subj# 6: interestingly, even if data follow an almost linear pattern, the model succeeds in predicting correctly the corresponding breakpoint.

We now try to fit the same model, but with a non-null covariance between the breakpoint and the slope difference random effects, i.e. $\sigma_{dk} \neq 0$ in ((4)).

```
> oseg2<-update(oseg1,
+               random=list(id=pdBlocked(list(pdDiag(~1+x), pdSymm(~U+G0-1)))))
 There were 14 warnings (use warnings() to see them)
```

While the procedure succeeds in fitting the model, several warning messages are returned.

```
> warnings()
Warning messages:
1: In logLik.reStruct(object, conLin) :
  Singular precision matrix in level -1, block 28
..............
14: In logLik.reStruct(object, conLin) :
  Singular precision matrix in level -1, block 28
```

The reason is that we are trying to fit a over-parametrized model having an additional covariance parameter which is actually zero. In general, when we try to estimate more parameters than the data support, it is not rare to run into such warnings or even error messages with no final fit. Starting the analysis with a simple structure, e.g. a diagonal matrix, for the random effects covariance matrix is, generally, recommended.

We conclude this section by running a small simulation experiment using the following code. Results are displayed in Table 1 based on `B=300` runs rather than `B=3`.

```
> #a small simulation study
> N=50
> set.seed(123)
> Xd=runif(N, 0, 2) #1 covariate for delta (N-vector)
> Z=cbind(1:N/N, rnorm(N))
>
> B=3
> V=matrix(,B,4)
> SE<-E<-matrix(,B,7)
```

```
>
> for(i in 1:B){
+   dd=sim.seg(N,ni=10,S=diag(c(.12,.05,.3,.1)^2), seed=i,
+            x.diff=list(0.5,Xd), z.psi=list(c(.3,-.1),Z), sd.eps=.01)
+   d=dd$data
+   suppressWarnings(oseg1 <-segmented.lme(y~x, ~x,
+             x.diff=~Xdif, z.psi=~Zpsi1+Zpsi2,
+             random=id~1+x+U+G0, data=d))
+
+   V[i,]<-sqrt(diag(getVarCov(oseg1)))#sd
+   E[i,]<-fixef(oseg1)
+   SE[i,]<-sqrt(diag(vcov(oseg1)))
+ }
```

Table 1 Results of the simulation experiment: mean, median and standard deviations of the sampling distributions, and mean of the standard errors. The last column refers to the coverage levels of the 95% confidence intervals based on the Normal distribution of the Wald statistic. Results based on 300 replicates.

	sampling distribution summary					
4-7parameter	true	mean	med	sd	$\bar{se}$	CL
β_0	2.0	2.001	2.002	0.016	0.017	0.970
β_1	−0.2	−0.199	−0.199	0.011	0.009	0.923
δ_0	0.3	0.279	0.271	0.092	0.095	0.937
δ_1	0.5	0.504	0.501	0.078	0.079	0.950
κ_0	0.4	0.402	0.403	0.030	0.028	0.910
κ_1	0.3	0.280	0.280	0.055	0.049	0.877
κ_2	−0.1	−0.094	−0.094	0.017	0.015	0.897
*[.2cm] σ_{b0}	0.12	0.119	0.119	0.012	-	-
σ_{b1}	0.05	0.052	0.051	0.014	-	-
σ_d	0.30	0.308	0.307	0.037	-	-
σ_k	0.10	0.090	0.090	0.011	-	-

The means and medians are rather overlapped, suggesting that the underlying distributions are fairly symmetric; the estimators appear substantially unbiased, with $\hat{\kappa}_0$, $\hat{\kappa}_1$, and $\hat{\kappa}_2$ sharing slight differences between the means and the corresponding true parameter values. The model-based standard errors appears to quantify appropriately the corresponding standard deviations for all estimators but $\hat{\kappa}_0$, $\hat{\kappa}_1$, and $\hat{\kappa}_2$. Their corresponding coverage levels in the last column exhibit values lower than the nominal 95%, unlike the other interval estimators having values quite close to 0.95. Overall, the breakpoint parameters appear, unsurprisingly, to be those having an unsatisfactory performance of the Wald statistic. Better performances, however, are expected with larger samples or in scenarios with higher signal-to-noise ratio. Broader simulation settings, and further investigation are beyond the goal of this paper; some additional results are discussed in Muggeo et al. (2014). Regardless the sample size or the signal-to-noise ratio, we remark that the Wald statistic $\hat{\delta}_0/\mathrm{SE}(\hat{\delta}_0)$ is useless, and the existence of the breakpoint, with or without heterogeneity, cannot currently be tested via the functions in the package.

At the end we report technical information on the machine used to run the code

```
> sessionInfo()
R version 4.6.2 (2026-10-29 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 26200)

Matrix products: default
  LAPACK version 3.12.1
```

```
locale:
[1] LC_COLLATE=Italian_Italy.utf8  LC_CTYPE=Italian_Italy.utf8
LC_MONETARY=Italian_Italy.utf8
[4] LC_NUMERIC=C                   LC_TIME=Italian_Italy.utf8

time zone: Europe/Rome
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] segmented_2.3-0 nlme_3.1-171    MASS_7.3-66

loaded via a namespace (and not attached):
[1] compiler_4.6.2 tools_4.6.2    splines_4.6.2  grid_4.6.2     stats4_4.6.2
lattice_0.23-1
```

4 Conclusions

We have discussed how to fit segmented mixed models within the R environmentvia the segmented package. At time of writing, to the best of author's knowledge, there is no package on CRAN to fit such class of models in a likelihood (non-Bayesian) framework. While the Bayesian approach can lead to in-depth results via the parameter posterior distributions, a simple effective framework could foster a somewhat more widespread adoption and usage of the model in practical applications (Krefman et al., 2021).

The method (and functions) are not limited to analysis of data equal number of measurements per subject, by working reasonably well and relatively fast. In our experience, some troubles can arise when the structure of the random effects covariance matrix is uselessly complicated, namely too many variances and covariances. Especially with noisy data, we suggest to start from a simple model, such as a diagonal covariance matrix, and then to try progressively more complex structures as illustrated in sec 3.2. However the AIC based on the log marginal likelihood, as that returned by `AIC()` function, is known not to be appropriate when the focus is on the clusters, and a *conditional* AIC should be used (Vaida & Blanchard, 2005). Implementing that, along with an hypothesis testing procedures for the existence of breakpoint and corresponding heterogeneity (Muggeo, 2025; Segalas et al., 2019; Spuck et al., 2026), and model fitting with more than 1 breakpoint, represent notable extensions for the segmented package.

Data and Code Availability

The analyses and relevant results reported in this paper, have been obtained via the R package segmented, version 0.2-3. The glucoseData dataset of section 3.1 (example 1) is shipped with the package, and the simulated dataset of section 3.2 is obtained via the code reported in the text. The file `Rcode.R` contains the R code to replicate the analyses, including the simulation experiment and corresponding results of Table 1. The file `Rcode.R` along with segmented 2.3-0 are available as Supplentary Material of this paper at `https://jrepstat.fairpressjournals.com/articles/`; The R package segmented is also downloadable from CRAN at
`https://cran.r-project.org/web/packages/segmented/index.html`.
The author is the creator, developer and the maintainer of segmented package.

Acknowledgements

The author thanks the Editor and the three referees for the insightful comments that led to a substantial improvement of the paper. The author also thanks dr. V. Genova for reading the draft version of the manuscript.

References

Cudeck, R., & Klebe, K. J. (2002). Multiphase mixed-effects models for repeated measures data. *Psychological Methods*, *7*(1), 41–63. https://doi.org/10.1037/1082-989x.7.1.41

Fitzmaurice, G. M., Laird, N. M., & Ware, J. H. (2004). *Applied longitudinal analysis*. Wiley.

Hardy, R. (2001). Commentary: Are piecewise mixed effects models useful in epidemiology? *International Journal of Epidemiology*, *30*(6), 1341–1342. https://doi.org/10.1093/ije/30.6.1341

Jacqmin-Gadda, H., Commenges, D., & Dartigues, J.-F. (2006). Random changepoint model for joint modeling of cognitive decline and dementia. *Biometrics*, *62*, 254–260.

Kiuchi, A. S., Hartigan, J. A., Holford, T. R., Rubinstein, P., & Stevens, C. E. (1995). Change Points in the Series of T4 Counts Prior to AIDS. *Biometrics*, *51*(1), 236. https://doi.org/10.2307/2533329

Krefman, A. E., Labarthe, D., Greenland, P., Pool, L., Aguayo, L., Juonala, M., Kähönen, M., Lehtimäki, T., Day, R. S., Bazzano, L., Muggeo, V. M. R., Van Horn, L., Liu, L., Webber, L. S., Pahkala, K., Laitinen, T. T., Raitakari, O., Lloyd-Jones, D. M., & Allen, N. B. (2021). Influential Periods in Longitudinal Clinical Cardiovascular Health Scores. *American Journal of Epidemiology*, *190*(11), 2384–2394. https://doi.org/10.1093/aje/kwab149

Lange, N., Carlin, B. P., & Gelfand, A. E. (1992). Hierarchical Bayes Models for the Progression of HIV Infection Using Longitudinal CD4 T-Cell Numbers. *Journal of the American Statistical Association*, *87*(419), 615–626. https://doi.org/10.1080/01621459.1992.10475258

Morrell, C. H., Pearson, J. D., Carter, H. B., & Brant, L. J. (1995). Estimating Unknown Transition Times Using a Piecewise Nonlinear Mixed-Effects Model in Men with Prostate Cancer. *Journal of the American Statistical Association*, *90*(429), 45–53. https://doi.org/10.1080/01621459.1995.10476487

Muggeo, V. M., Atkins, D. C., Gallop, R. J., & Dimidjian, S. (2014). Segmented mixed models with random changepoints: a maximum likelihood approach with application to treatment for depression study. *Statistical Modelling*, *14*(4), 293–313. https://doi.org/10.1177/1471082x13504721

Muggeo, V. M. R. (2003). Estimating regression models with unknown break-points. *Statistics in Medicine*, *22*(19), 3055–3071. https://doi.org/10.1002/sim.1545

Muggeo, V. M. R. (2008). Segmented: An R package to fit regression models with broken-line relationships. *R News*, *8*(1), 20–25.

Muggeo, V. M. R. (2016). Testing with a nuisance parameter present only under the alternative: a score-based approach with application to segmented modelling. *Journal of Statistical Computation and Simulation*, *86*(15), 3059–3067. https://doi.org/10.1080/00949655.2016.1149855

Muggeo, V. M. R. (2025). Letter to the Editors: Comments on “Statistical inference on change points in generalized semiparametric segmented models” by Yang et al. (2025). *Biometrics*, *81*(4). https://doi.org/10.1093/biomtc/ujaf147

Pinheiro, J. C., & Bates, D. M. (2000). *Mixed-effects models in s and s-PLUS,*. Springer.

R Core Team. (2026). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. https://doi.org/10.32614/r.manuals

Segalas, C., Amieva, H., & Jacqmin-Gadda, H. (2019). A hypothesis testing procedure for random changepoint mixed models. *Statistics in Medicine*, *38*(20), 3791–3803. https://doi.org/10.1002/sim.8195

Spuck, N., Schmid, M., & Berger, M. (2026). Detection of nonlinearity, discontinuity and interactions in generalized regression models. *Statistical Modelling*, *26*, 248–276.

Vaida, F., & Blanchard, S. (2005). Conditional Akaike information for mixed-effects models. *Biometrika*, *92*(2), 351–370. https://doi.org/10.1093/biomet/92.2.351

Wood, S. N. (2001). Minimizing Model Fitting Objectives That Contain Spurious Local Minima by Bootstrap Restarting. *Biometrics*, *57*(1), 240–244. https://doi.org/10.1111/j.0006-341x.2001.00240.x

Zerbe, G. O. (1979). Randomization Analysis of the Completely Randomized Design Extended to Growth and Response Curves. *Journal of the American Statistical Association*, *74*(365), 215–221. https://doi.org/10.1080/01621459.1979.10481640

COLOPHON Typeset with Typst 0.15.0. Template jrepstat.typ at commit 7352725. Set in STIX Two Text, STIX Two Math, Public Sans and JetBrains Mono. Built 24 November 2026.