`fitTimeSeries`: Longitudinal differential abundance analysis for marker-gene surveys

Hisham Talukder, Joseph N. Paulson, Hector Corrada Bravo

Applied Mathematics & Statistics, and Scientific Computation
Center for Bioinformatics and Computational Biology
University of Maryland, College Park

jpaulson@umiacs.umd.edu

Modified: February 18, 2015. Compiled: April 15, 2025

Contents

1 Introduction

This is a vignette specifically for the fitTimeSeries function. For a full list of functions available in the package: help(package=metagenomeSeq). For more information about a particular function call: ?function.

Smoothing spline regression models [1] are commonly used to model longitudinal data and form the basis for methods used in a large number of applications [2,3]. Specifically, an extension of the methodology called Smoothing-Spline ANOVA [4] is capable of directly estimating a smooth function of interest while incorporating other covariates in the model.

A common approach to detect regions/times of interest in a genome or for differential abundance is to model differences between two groups with respect to the quantitative measurements as smooth functions and perform statistical inference on these models. In particular, widely used methods for region finding using DNA methylation data use local regression methods to estimate these smooth functions. An important aspect of these tools is their ability to incorporate sample characteristics as covariates in these models, e.g., sex and age in population studies, or technical factors like processing batches. Incorporating these sources of variability, both biological and technical is essential in high-throughput studies. Therefore, these methods require that the models used are capable of estimating both smooth functions and sample-specfic characteristics. We present fitTimeSeries - a method for estimating and detecting regions/times of interest due to differential abundance of a quantitative measurement (for example, normalized abundance).

1.1 Problem Formulation

We model data in the following form:

$$Y_{itk} = f_i(t, x_k) + e_{tk}$$

where i represents group factor (diet, health status, etc.), t represents series factor (for example, time or location), k represents replicate observations, x_k are covariates for sample k (including an indicator for group membership $I\{k \in i\}$) and e_{tk} are independent $N(0, \sigma^2)$ errors. We assume f_i to be a smooth function, defined in an interval $[a, b]$, that can be parametric, non-parametric or a mixture of both.

Our goal is to identify intervals where the absolute difference between two groups $\eta_d(t) = f_1(t, \cdot) - f_2(t, \cdot)$ is large, that is, regions, R_{t_1,t_2}, where: $R_{t_1,t_2} = \{t_1, t_2 \in x \; such \; that \; |\eta_d(x)| \geq C\}$ and C is a predefined constant threshold.

To identify these areas we use hypothesis testing using the area $A_{t_1,t_2} = \int_{R_{t_1,t_2}} \eta_d(t)dt$ under the estimated function of $\eta_d(t)$ as a statistic with null and alternative hypotheses

$$H_0 : A_{t_1,t_2} \leq K$$

$$H_1 : A_{t_1,t_2} > K$$

with K some fixed threshold.

We employ a permutation-based method to calculate a null distribution of the area statistics $A_(t1, t2)$'s. To do this, the group-membership indicator variables (0-1 binary variable) are randomly permuted B times, e.g., $B = 1000$ and the method above is used to estimate the difference function η_d^b (in this case simulating the null hypothesis) and an area statistics $A_(t1, t2)^b$ for each random permutation. Estimates $A_(t1, t2)^b$ are then used to construct an empirical estimate of $A_(t1, t2)$ under the null hypothesis. The observed area, $A_(t1, t2)^*$, is compared to the empirical null distribution to calculate a p-value. Figure 1 illustrates the relationship between $R_(t1, t2)$ and $A_(t1, t2)$. The key is to estimate regions $R_(t1, t2)$ where point-wise confidence intervals would be appropriate.

2 Data preparation

Data should be preprocessed and prepared in tab-delimited files. Measurements are stored in a matrix with samples along the columns and features along the rows. For example, given m features and n samples, the entries in a marker-gene or metagenomic count matrix $\mathbf{C}$ (m, n), c_{ij}, are the number of reads annotated for a particular feature i (whether it be OTU, species, genus, etc.) in sample j. Alternatively, the measurements could be some quantitative measurement such as methylation percentages or CD4 levels.

$$\begin{array}{c} \\ feature_1 \\ feature_2 \\ \vdots \\ feature_m \end{array} \begin{array}{c} \begin{array}{cccc} sample_1 & sample_2 & \dots & sample_n \end{array} \\ \begin{pmatrix} c_{11} & c_{12} & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \dots & c_{mn} \end{pmatrix} \end{array}$$

Data should be stored in a file (tab-delimited by default) with sample names along the first row, feature names in the first column and should be loaded into R and formatted into a MRexperiment object. To prepare the data please read the section on data preparation in the full metagenomeSeq vignette - `vignette("metagenomeSeq")`.

2.1 Example datasets

There is a time-series dataset included as an examples in the `metagenomeSeq` package. Data needs to be in a `MRexperiment` object format to normalize, run the statistical tests, and visualize. As an example, throughout the vignette we'll use the following datasets. To understand a `fitTimeSeries`'s usage or included data simply enter `?fitTimeSeries`.

```
library(metagenomeSeq)
library(gss)
```

2. Humanized gnotobiotic mouse gut [5]: Twelve germ-free adult male C57BL/6J mice were fed a low-fat, plant polysaccharide-rich diet. Each mouse was gavaged with healthy adult human fecal material. Following the fecal transplant, mice remained on the low-fat, plant polysacchaaride-rich diet for four weeks, following which a subset of 6 were switched to a high-fat and high-sugar diet for eight weeks. Fecal samples for each mouse went through PCR amplification of the bacterial 16S rRNA gene V2 region weekly. Details of experimental protocols and further details of the data can be found in Turnbaugh et. al. Sequences and further information can be found at: `http://gordonlab.wustl.edu/TurnbaughSE_10_09/STM_2009.html`

```
data(mouseData)
mouseData

## MRexperiment (storageMode: environment)
## assayData: 10172 features, 139 samples
##   element names: counts
## protocolData: none
## phenoData
##   sampleNames: PM1:20080107 PM1:20080108 ... PM9:20080303
##     (139 total)
```

```
##   varLabels: mouseID date ... status (5 total)
##   varMetadata: labelDescription
## featureData
##   featureNames: Prevotellaceae:1 Lachnospiraceae:1 ...
##     Parabacteroides:956 (10172 total)
##   fvarLabels: superkingdom phylum ... OTU (7 total)
##   fvarMetadata: labelDescription
## experimentData: use 'experimentData(object)'
## Annotation:
```

2.2 Creating a `MRexperiment` object with other measurements

For a fitTimeSeries analysis a minimal MRexperiment-object is required and can be created using the function `newMRexperiment` which takes a count matrix described above and phenoData (annotated data frame). `Biobase` provides functions to create annotated data frames.

```
# Creating mock sample replicates
sampleID = rep(paste("sample", 1:10, sep = ":"), times = 20)
# Creating mock class membership
class = rep(c(rep(0, 5), rep(1, 5)), times = 20)
# Creating mock time
time = rep(1:20, each = 10)

phenotypeData = AnnotatedDataFrame(data.frame(sampleID, class, time))
# Creating mock abundances
set.seed(1)
# No difference
measurement1 = rnorm(200, mean = 100, sd = 1)
# Some difference
measurement2 = rnorm(200, mean = 100, sd = 1)
measurement2[1:5] = measurement2[1:5] + 100
measurement2[11:15] = measurement2[11:15] + 100
measurement2[21:25] = measurement2[21:25] + 50
mat = rbind(measurement1, measurement2)
colnames(mat) = 1:200
mat[1:2, 1:10]
```

```
##                       1        2         3        4        5
## measurement1  99.37355 100.1836  99.16437 101.5953 100.3295
## measurement2 200.40940 201.6889 201.58659 199.6691 197.7148
##                       6        7        8        9       10
## measurement1  99.17953 100.4874 100.7383 100.5758  99.69461
## measurement2 102.49766 100.6671 100.5413  99.9866 100.51011
```

If phylogenetic information exists for the features and there is a desire to aggregate measurements based on similar annotations choosing the featureData column name in lvl will aggregate measurements using the default parameters in the `aggregateByTaxonomy` function.

```
# This is an example of potential lvl's to aggregate by.
data(mouseData)
colnames(fData(mouseData))
```

```
## [1] "superkingdom" "phylum"       "class"        "order"
## [5] "family"       "genus"        "OTU"
```

Here we create the actual MRexperiment to run through fitTimeSeries.

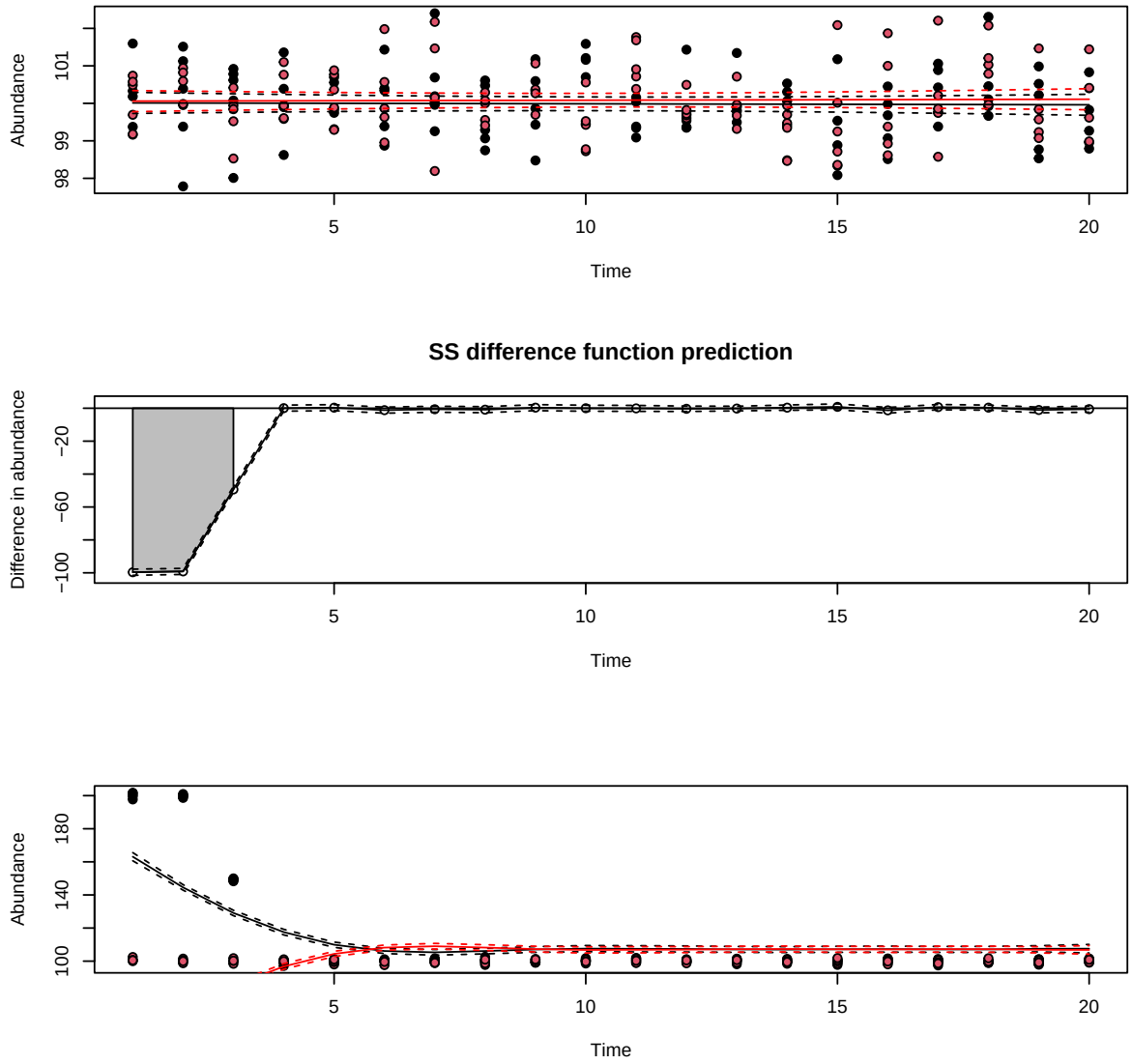
```
obj = newMRexperiment(counts=mat,phenoData=phenotypeData)
obj

## MRexperiment (storageMode: environment)
## assayData: 2 features, 200 samples
##   element names: counts
## protocolData: none
## phenoData
##   sampleNames: 1 2 ... 200 (200 total)
##   varLabels: sampleID class time
##   varMetadata: labelDescription
## featureData: none
## experimentData: use 'experimentData(object)'
## Annotation:

res1 = fitTimeSeries(obj,feature=1,
              class='class',time='time',id='sampleID',
              B=10,norm=FALSE,log=FALSE)
res2 = fitTimeSeries(obj,feature=2,
              class='class',time='time',id='sampleID',
              B=10,norm=FALSE,log=FALSE)


classInfo = factor(res1$data$class)
```

```
par(mfrow=c(3,1))
plotClassTimeSeries(res1,pch=21,bg=classInfo)
plotTimeSeries(res2)
plotClassTimeSeries(res2,pch=21,bg=classInfo)
```



3 Time series analysis

Implemented in the `fitTimeSeries` function is a method for calculating time intervals for which bacteria are differentially abundant. Fitting is performed using Smoothing Splines ANOVA (SS-ANOVA), as implemented in the `gss` package. Given observations at multiple time points for two groups the method calculates a function modeling the difference in abundance across all time. Using group membership permutations we estimate a null distribution of areas under the difference curve for the time intervals of interest and report significant intervals of time. Here we provide a real example from the microbiome of two groups of mice on different diets.

The gnotobiotic mice come from a longitudinal study ideal for this type of analysis. We choose to perform our analysis at the class level and look for differentially abundant time intervals for "Actinobacteria". For demonstrations sake we perform only 10 permutations.

If you find the method useful, please cite: "Longitudinal differential abundance analysis for marker-gene surveys" Talukder H*, Paulson JN*, Bravo HC. (Submitted)

```
res = fitTimeSeries(obj = mouseData, lvl = "class", feature = "Actinobacteria",
    class = "status", id = "mouseID", time = "relativeTime", B = 10)

# We observe a time period of differential abundance for
# 'Actinobacteria'
res$timeIntervals

##      Interval start Interval end     Area    p.value
## [1,]              9           50 92.32485 0.09090909

str(res)

## List of 5
##  $ timeIntervals: num [1, 1:4] 9 50 92.3249 0.0909
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:4] "Interval start" "Interval end" "Area" "p.value"
##  $ data         :'data.frame': 139 obs. of  9 variables:
##   ..$ abundance   : num [1:139] 0 3.82 3.13 7.4 0 ...
##   ..$ class       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ time        : num [1:139] 21 22 28 0 35 6 42 49 56 63 ...
##   ..$ id          : Factor w/ 12 levels "PM1","PM10","PM11",..: 1 1 1 1 1 1 1
##   ..$ mouseID     : Factor w/ 12 levels "PM1","PM10","PM11",..: 1 1 1 1 1 1 1
##   ..$ date        : Date[1:139], format: "2008-01-07" ...
##   ..$ diet        : chr [1:139] "BK" "BK" "BK" "BK" ...
##   ..$ relativeTime: num [1:139] 21 22 28 0 35 6 42 49 56 63 ...
##   ..$ status      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ fit          :'data.frame': 78 obs. of  3 variables:
##   ..$ fit       : num [1:78] 0.351 0.487 0.619 0.75 0.878 ...
##   ..$ se        : num [1:78] 1.091 1.024 0.966 0.916 0.875 ...
##   ..$ timePoints: num [1:78] 0 1 2 3 4 5 6 7 8 9 ...
##  $ perm         : num [1:10, 1] 2.9 12.5 12.1 -30.5 -36.5 ...
##  $ call         : language fitTimeSeries(obj = mouseData, feature = "Actinoba
```

For example, to test every class in the mouse dataset:

```
set.seed(123)
classes = unique(fData(mouseData)[,"class"])

timeSeriesFits = lapply(classes,function(i){
        fitTimeSeries(obj=mouseData,
            feature=i,
            class="status",
            id="mouseID",
            time="relativeTime",
            lvl='class',
            C=.3,# a cutoff for 'interesting'
            B=1) # B is the number of permutations and should clearly not be 1
    })
names(timeSeriesFits) = classes

# Removing classes of bacteria without a potentially
```

```
# interesting time interval difference.
timeSeriesFits = lapply(timeSeriesFits,function(i){i[[1]]})[-grep("No",timeSeries

# Naming the various interesting time intervals.
for(i in 1:length(timeSeriesFits)){
    rownames(timeSeriesFits[[i]]) =
      paste(
        paste(names(timeSeriesFits)[i]," interval",sep=""),
        1:nrow(timeSeriesFits[[i]]),sep=":"
      )
}

# Merging into a table.
timeSeriesFits = do.call(rbind,timeSeriesFits)

# Correcting for multiple testing.
pvalues = timeSeriesFits[,"p.value"]
adjPvalues = p.adjust(pvalues,"bonferroni")
timeSeriesFits = cbind(timeSeriesFits,adjPvalues)

head(timeSeriesFits)
```

```
##                                    Interval start Interval end
## Bacteroidetes interval:1                       17           20
## Bacteroidetes interval:2                       22           77
## Bacilli interval:1                             21           77
## Erysipelotrichi interval:1                     16           77
## Betaproteobacteria interval:1                  24           33
## Epsilonproteobacteria interval:1               17           22
##                                          Area p.value adjPvalues
## Bacteroidetes interval:1           10.422590     0.5          1
## Bacteroidetes interval:2         -117.021102     0.5          1
## Bacilli interval:1                471.995977     0.5          1
## Erysipelotrichi interval:1        115.957380     0.5          1
## Betaproteobacteria interval:1     -21.148062     0.5          1
## Epsilonproteobacteria interval:1   -3.261411     0.5          1
```

Please see the help page for `fitTimeSeries` for parameters. Note, only two groups can be compared to each other and the time parameter must be an actual value (currently no support for posix, etc.).

3.1 Paramaters

There are a number of parameters for the `fitTimeSeries` function. We list and provide a brief discussion below. For parameters influencing `ssanova`, `aggregateByTaxonomy`, `MRcounts` type ?function for more details.

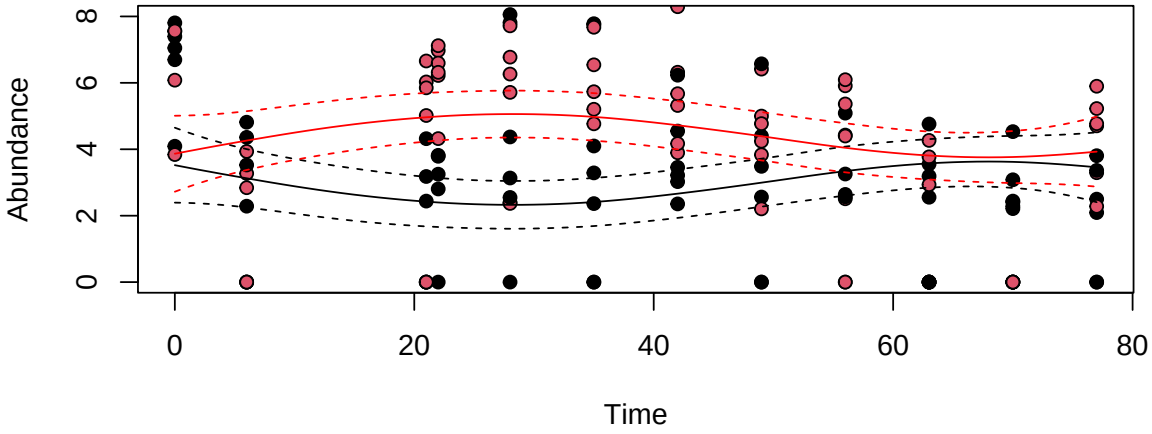
- obj - the metagenomeSeq MRexperiment-class object.
- feature - Name or row of feature of interest.
- class - Name of column in phenoData of MRexperiment-class object for class memberhip.
- time - Name of column in phenoData of MRexperiment-class object for relative time.

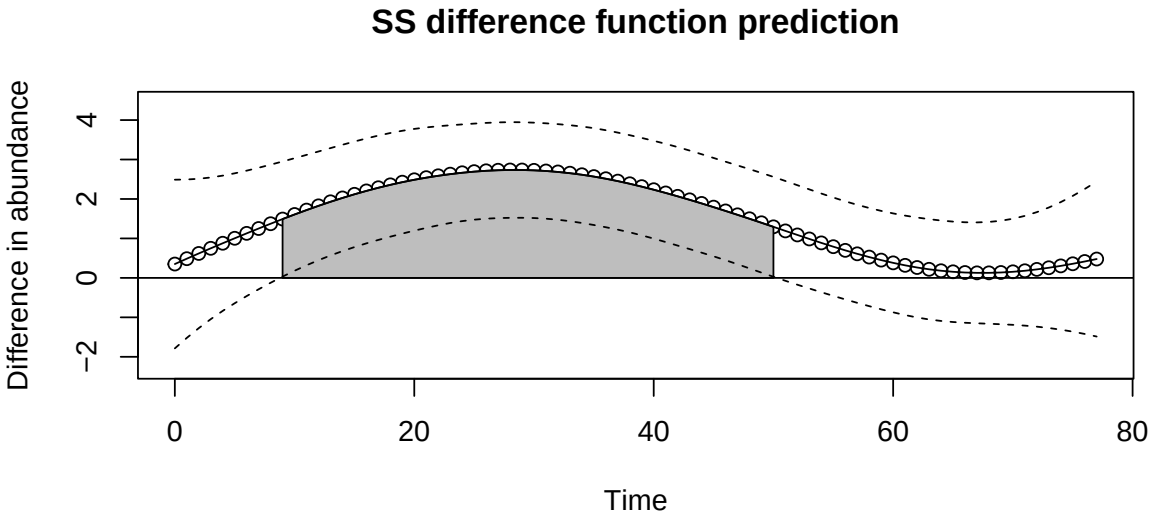
- id - Name of column in phenoData of MRexperiment-class object for sample id.
- method - Method to estimate time intervals of differentially abundant bacteria (only ssanova method implemented currently).
- lvl - Vector or name of column in featureData of MRexperiment-class object for aggregating counts (if not OTU level).
- C - Value for which difference function has to be larger or smaller than (default 0).
- B - Number of permutations to perform (default 1000)
- norm - When aggregating counts to normalize or not. (see MRcounts)
- log - Log2 transform. (see MRcounts)
- sl - Scaling value. (see MRcounts)
- ... - Options for ssanova

4 Visualization of features

To help with visualization and analysis of datasets `metagenomeSeq` has several plotting functions to gain insight of the model fits and the differentially abundant time intervals using `plotClassTimeSeries` and `plotTimeSeries` on the result. More plots will be updated.

```
par(mfrow = c(2, 1))
plotClassTimeSeries(res, pch = 21, bg = res$data$class, ylim = c(0,
    8))
plotTimeSeries(res)
```





5 Summary

metagenomeSeq's fitTimeSeries is a novel methodology for differential abundance testing of longitudinal data. If you make use of the statistical method please cite our paper. If you made use of the manual/software, please cite the manual/software!

5.1 Citing fitTimeSeries

```
citation("metagenomeSeq")

## To cite the original statistical method and normalization
## method implemented in metagenomeSeq use
##
##   Paulson JN, Stine OC, Bravo HC, Pop M (2013).
##   "Differential abundance analysis for microbial
```

```
##   marker-gene surveys." _Nat Meth_, *advance online
##   publication*. doi:10.1038/nmeth.2658
##   <https://doi.org/10.1038/nmeth.2658>,
##   <http://www.nature.com/nmeth/journal/vaop/ncurrent/abs/nmeth.2658.html>.
##
## To cite the metagenomeSeq software/vignette guide use
##
##   Paulson JN, Olson ND, Braccia DJ, Wagner J, Talukder H,
##   Pop M, Bravo HC (2013). _metagenomeSeq: Statistical
##   analysis for sparse high-throughput sequncing._.
##   Bioconductor package,
##   <http://www.cbcb.umd.edu/software/metagenomeSeq>.
##
## To cite time series analysis/function fitTimeSeries use
##
##   Paulson* JN, Talukder* H, Bravo HC (2017).
##   "Longitudinal differential abundance analysis of
##   marker-gene surveys using smoothing splines."
##   _biorxiv_. doi:10.1101/099457
##   <https://doi.org/10.1101/099457>,
##   <https://www.biorxiv.org/content/10.1101/099457v1>.
##
## To see these entries in BibTeX format, use
## 'print(<citation>, bibtex=TRUE)', 'toBibtex(.)', or set
## 'options(citation.bibtex.max=999)'.
```

5.2 Session Info

```
sessionInfo()

## R version 4.5.0 RC (2025-04-04 r88126)
## Platform: aarch64-apple-darwin20
## Running under: macOS Ventura 13.7.1
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libR
## LAPACK: /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libR
##
## locale:
## [1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: America/New_York
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods
## [7] base
##
## other attached packages:
##  [1] gss_2.2-9            metagenomeSeq_1.50.0
```

```
##  [3] RColorBrewer_1.1-3   glmnet_4.1-8
##  [5] Matrix_1.7-3         limma_3.64.0
##  [7] Biobase_2.68.0       BiocGenerics_0.54.0
##  [9] generics_0.1.3       knitr_1.50
##
## loaded via a namespace (and not attached):
##  [1] gplots_3.2.0       xfun_0.52          Wrench_1.26.0
##  [4] KernSmooth_2.23-26 highr_0.11         gtools_3.9.5
##  [7] statmod_1.5.0      formatR_1.14       locfit_1.5-9.12
## [10] grid_4.5.0         caTools_1.18.3     evaluate_1.0.3
## [13] bitops_1.0-9       foreach_1.5.2      compiler_4.5.0
## [16] codetools_0.2-20   Rcpp_1.0.14        lattice_0.22-7
## [19] parallel_4.5.0     splines_4.5.0      shape_1.4.6.1
## [22] tools_4.5.0        iterators_1.0.14   matrixStats_1.5.0
## [25] survival_3.8-3
```

References

[1] G. Wahba. *Spline Models in Statistics*. CBMS-NSF Regional Conference Series. SIAM, Philadelphia, PA, 1990.

[2] Héctor Corrada Bravo. *Graph-based data analysis: tree-structured covariance estimation, prediction by regularized kernel estimation and aggregate database query processing for probabilistic inference*. ProQuest, 2008.

[3] H. Jaroslaw, N. Elena, and M.L. Nan. Longcrisp: A test for bumphunting in longitudinal data. *Statistics in Medicine*, 26:1383–1397, 2006.

[4] C. Gu. *Smoothing Spline Anova Model*. Springer Series in Statistics. Springer, 2002.

[5] Peter J Turnbaugh, Vanessa K Ridaura, Jeremiah J Faith, Federico E Rey, Rob Knight, and Jeffrey I Gordon. The effect of diet on the human gut microbiome: a metagenomic analysis in humanized gnotobiotic mice. *Science translational medicine*, 1(6):6ra14, 2009.