

Package ‘finitefourierfits’

July 27, 2020

Title Fit a Finite Fourier Basis to Data

Version 0.0.0.9000

Description Curve-fitting that uses fft to approximate a 2D relationship, and then fine-tunes that fit with nls.

Depends R (>= 3.4.4)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Suggests knitr, rmarkdown, testthat,

Collate 'finitefourierfits.R' 'utilities.R' 'fourier_summary.R'
'terms.R' 'fffit.R'

VignetteBuilder knitr

NeedsCompilation no

Author Benjamin Taft [aut, cre]

Maintainer Benjamin Taft <ben.taft@landmarkacoustics.com>

R topics documented:

.argmin	2
anova.fffit	2
coef.fffit	3
fffit	3
fffterm	4
finitefourierfits	5
formula.fffit	5
fourier.summary	6
half.fft	6
omegas	7
predict.fffit	7
summary.fffit	8

Index**9**

`.argmin`*The index of the minimum element in a container*

Description

`argmin` acts just like the mathematical concept it is named after.

Usage

```
.argmin(x)
```

Arguments

`x` Some collection, probably a [vector\(\)](#) of some kind.

Value

the index of the smallest item in the collection.

`anova.fffit`*Comparing the Quality of the Possible Fits*

Description

`anova.fffit` prints an ANOVA table of all of the models that the fit made.

Usage

```
## S3 method for class 'fffit'  
anova(object, ...)
```

Arguments

`object` An fffit object
`...` optional, additional arguments to `anova`. DON'T USE

Value

an anova object

See Also

[anova\(\)](#), [nls\(\)](#)

coef.fff ^{it}	<i>The Coefficients of the Best Fit</i>
------------------------	-----------------------------------------

Description

coef.fff^{it} returns the coefficients of the best-fit model

Usage

```
## S3 method for class 'fffit'
coef(object, index = NULL, ...)
```

Arguments

object	An fff ^{it} object
index	optional, a specific model to extract coefficients from.
...	optional, additional arguments to coef()

Value

a vector of coefficients

See Also

[coef\(\)](#), [nls\(\)](#)

fff ^{it}	<i>Fit a small, Fourier basis-like model to data</i>
-------------------	------------------------------------------------------

Description

When a functional-looking relation is poorly approximated by the tools [lm\(\)](#), [glm\(\)](#), or [nls\(\)](#), one option is to use a linear combination of cosine terms to approximate the relation within a strictly bounded domain. This function does just that.

Usage

```
fffit(x, y, model.selector = BIC, max.terms = 10, pad.multiplier = 4)
```

Arguments

x	Numeric, the independent variable
y	Numeric, the dependent variable
model.selector	Function, optional, the function for comparing models
max.terms	Integer, optional, the greatest number of terms allowed
pad.multiplier	Integer, optional, how much zero-padding to add

Value

an FFFit model

Examples

```
x <- rnorm(10)
y <- 10*x + rnorm(10)
my.fit <- fffit(x, y)
```

fffterm

Summarize a DFT Term for Use in an FFFit

Description

Several pieces of information from each term go in to building an `fffit()`, and it makes sense to calculate and store them once.

Usage

```
fffterm(ranking, obj)
```

Arguments

<code>ranking</code>	An integer representing where the term falls in some ordering of the terms of the DFT.
<code>obj</code>	A <code>fourier.summary()</code> object.

Value

an object of type `fffterm`, which has the following named members:

- a** The computed amplitude of the term
- p** The phase of the term
- f** The frequency of the term
- term** A string, " $a_n \cos(2\pi f_n x + p_n)$ ", for the ranking n .

See Also

`fourier.summary()`

finitefourierfits	<i>Fit a Finite Fourier Basis to Data</i>
-------------------	-------------------------------------------

Description

Sometimes there seems to be a clear functional relationship between two variables in a data set, but it is poorly approximated by polynomials. A Fourier basis may be able to help. It is, basically, a linear combination of harmonically-related, phase-shifted, weighted cosine curves. This package uses a very simple-minded approach to finding a basis with few curves.

formula.fffitt	<i>The Formula of the Best Fit</i>
----------------	------------------------------------

Description

formula.fffitt returns the formula of the best-fit model

Usage

```
## S3 method for class 'fffitt'  
formula(x, index = NULL, ...)
```

Arguments

x	An fffit object
index	optional, a specific model to get the formula for
...	optional, additional arguments to formula()

Value

a formula

See Also

[formula\(\)](#), [nls\(\)](#)

fourier.summary	<i>Compute a DFT, and Some Helpful Summaries and Descriptors</i>
-----------------	------------------------------------------------------------------

Description

fourier.summary zero-pads x, calculates the normal real to half-complex DFT of the padded vector, and then stores the amplitudes and phases, and amplitude rank of the non-negative frequency terms.

Usage

```
fourier.summary(x, sample.rate, multiplier = 4L)
```

Arguments

x	A numeric vector to be Fourier transformed
sample.rate	The sample rate of the time series underlying x.
multiplier	optional, controls how much padding x gets.

Value

an S3 object of class `fourier.summary()` with the following names:

dft The complex-valued `fft()` of x after it has been right-padded with zeros.

fft.size The length of dft

sample.rate This isn't actually used in the ctor, but its super useful to have here. } \item{a}{The real-valued magnitudes of the non-negative frequency term.} \item{f}{Frequencies in [0, Nyquist)} \item{p}{The real-valued phases of each non-negative frequency term.} \item{magnitude.order}{Indices into a ' that sort it from largest to smallest.

See Also

`fft()`, `half.fft()`

half.fft	<i>Half of the input, rounded down to the nearest integer.</i>
----------	----------------------------------------------------------------

Description

half.fft returns the pivot index between the positive and negative frequencies in the output of `fft()`.

Usage

```
half.fft(fft.size)
```

Arguments

`fft.size` The length of the Discrete Fourier Transform window.

Value

An integer equal to half of `fft.size`, rounded down.

<code>omegas</code>	<i>Angular frequencies for a Discrete Fourier Transform</i>
---------------------	-------------------------------------------------------------

Description

If there are `fft.size` samples in the DFT then the frequencies run from 0 (inclusive) to 2π (exclusive).

Usage

```
omegas(fft.size)
```

Arguments

`fft.size` The size of the DFT

Value

a vector of floats of length `fft.size` in $[0, 2\pi)$.

Examples

```
omegas(4)
```

<code>predict.fffit</code>	<i>Predicting from Finite Fourier Fits</i>
----------------------------	--------------------------------------------

Description

`predict.fffit` produces predicted values, obtained by evaluating the fit in the context of the `newdata`. This will give nonsensical results if the input domain is not within the bounds of the original analysis's domain.

Usage

```
## S3 method for class 'fffit'
predict(object, newdata = NULL, index = NULL, ...)
```

Arguments

object	An <code>fffit()</code> object
newdata	A data frame with new independent and dependent variables.
index	optional, a specific model to summarize
...	Additional arguments to <code>predict.nls</code>

Value

A numeric vector of predictions

See Also

[predict.nls\(\)](#)

summary.fffit	<i>Demonstrating the Quality of the Best Fit</i>
---------------	--------------------------------------------------

Description

`summary.fffit` prints an summary table of the model that fit best.

Usage

```
## S3 method for class 'fffit'  
summary(object, index = NULL, ...)
```

Arguments

object	An <code>fffit</code> object
index	optional, a specific model to summarize
...	optional, additional arguments to summary()

Value

a summary object

See Also

[summary\(\)](#), [nls\(\)](#)

Index

.argmin, [2](#)

anova(), [2](#)
anova.fffrit, [2](#)

coef(), [3](#)
coef.fffrit, [3](#)

fffrit, [3](#)
fffrit(), [4](#), [8](#)
fffritterm, [4](#)
fft(), [6](#)
finitefourierfits, [5](#)
formula(), [5](#)
formula.fffrit, [5](#)
fourier.summary, [6](#)
fourier.summary(), [4](#), [6](#)

glm(), [3](#)

half.fft, [6](#)
half.fft(), [6](#)

lm(), [3](#)

nls(), [2](#), [3](#), [5](#), [8](#)

omegas, [7](#)

predict.fffrit, [7](#)
predict.nls(), [8](#)

summary(), [8](#)
summary.fffrit, [8](#)

vector(), [2](#)