Package ‘FMM’

October 12, 2022

Type Package

Title Rhythmic Patterns Modeling by FMM Models

Version 0.3.1

Author Itziar Fernandez, Alejandro Rodriguez-Collado, Yolanda Larriba, Adrian Lamela, Christian Canedo, Cristina Rueda

Maintainer Itziar Fernandez <itziar.fernandez@uva.es>

Description Provides a collection of functions to fit and explore single, multi-component and restricted Frequency Modulated Moebius (FMM) models. 'FMM' is a nonlinear parametric regression model capable of fitting non-sinusoidal shapes in rhythmic patterns. Details about the mathematical formulation of 'FMM' models can be found in Rueda et al. (2019) <doi:10.1038/s41598-019-54569-1>.

URL https://github.com/alexARC26/FMM

License GPL (>= 2)

Encoding UTF-8

LazyData true

Depends R (>= 3.5)

Imports methods, rlang, foreach, iterators, parallel, doParallel

Suggests ggplot2, gridExtra, knitr, RColorBrewer, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation no

RoxygenNote 7.1.2

Config/testthat/edition 3

Repository CRAN

Date/Publication 2021-12-17 12:52:03 UTC
Description

Provides a collection of functions to fit and explore single, multi-component and restricted Frequency Modulated Moebius (FMM) models. "FMM" is a nonlinear parametric regression model capable of fitting non-sinusoidal shapes in rhythmic patterns.

Details

Package: FMM
Type: Package
Version: 0.1.1
Date: 2021
License: GPL Version 2 or later

For a complete list of functions with individual help pages, use library(help = "FMM").

Author(s)

Adrian Lamela, Itziar Fernandez, Yolanda Larriba, Alejandro Rodriguez, Cristina Rueda
Maintainer: Itziar Fernandez <itziar.fernandez@uva.es>

References

Description

Voltage electric activity data (mV) of the fifth annotated heartbeat for patient sel100 in 'QT database'. 200 samples were collected along 0.8 seconds with a sampling frequency of 250 Hz. Data records correspond to samples from 151248 to 151049 and can be downloaded as text files from 'Physionet' website. Annotated beats were manually revised by experts including R peaks and fiducial marks.

Usage

data(ecgData)

Format

A numeric vector.

Source

'QT database' from 'Physionet' <https://archive.physionet.org/cgi-bin/atm/ATM>

References


Examples

data(ecgData)
str(ecgData)

extractWaves

*Individual contribution to the fitted values of each FMM wave*

Description

extractWaves extracts individual contribution to the fitted values of each FMM wave.

Usage

extractWaves(objFMM)
fitFMM

Fitting FMM models

Description

fitFMM() is used to fit FMM models. The only required argument to fit FMM models is the input data. By default it is assumed that time points, corresponding to a single time period, are equally spaced from 0 to 2π.

Usage

fitFMM(
  vData,
  nPeriods = 1,
  timePoints = NULL,
  nback = 1,
  betaOmegaRestrictions = 1:nback,
  maxiter = nback,
  stopFunction = alwaysFalse,
  lengthAlphaGrid = 48,
  lengthOmegaGrid = 24,
  numReps = 3,
  showProgress = TRUE,
  showTime = FALSE,
`fitFMM`

```r
parallelize = FALSE,
restrExactSolution = FALSE
)
```

**Arguments**

- `vData` A numeric vector containing the data to be fitted a FMM model.
- `nPeriods` A numeric value specifying the number of periods at which `vData` is observed.
- `timePoints` A numeric vector containing the time points at which each data of one single period is observed. The default value is `NULL`, in which case they are equally spaced in range $[0, 2\pi]$. It must be between 0 and $2\pi$.
- `nback` Number of FMM components to be fitted. Its default value is 1.
- `betaOmegaRestrictions` An integer vector of length `nback` indicating which FMM waves are constrained to have equal beta and omega parameters. For example, `c(1,1,1,2,2)` indicates that $\beta_1=\beta_2=\beta_3$ and $\beta_4=\beta_5$ as well as $\omega_1=\omega_2=\omega_3$ and $\omega_4=\omega_5$. In brief, some waves are restricted to have the same shape. Its default value is the sequence `1:nback` to fit the FMM model without restrictions on shape parameters (beta and omega).
- `maxiter` Maximum number of iterations for the backfitting algorithm. By default, it is set at `nback`.
- `stopFunction` Function to check the convergence criterion for the backfitting algorithm (see Details).
- `lengthAlphaGrid` Precision of the grid of alpha in the search of the best model. If it is increased, more possible values of alpha will be considered, resulting in an increasing in the computation time too. By default, it is set to 48 possible values of alpha, equally spaced between 0 and $2\pi$.
- `lengthOmegaGrid` Precision of the grid of omega in the search of the best model. If it is increased, more possible values of omega will be considered, resulting in an increasing in the computation time too. By default it is set to 24 possible values of omega, equally spaced between 0 and 1 in a logarithmic way.
- `numReps` Number of times (alpha, omega) parameters are refined.
- `showProgress` `TRUE` to display a progress indicator on the console.
- `showTime` `TRUE` to display execution time on the console.
- `parallelize` `TRUE` to use parallelized procedure to fit FMM model. Its default value is `FALSE`. When it is `TRUE`, the number of cores to be used is equal to 12, or if the machine has less, the number of cores - 1.
- `restrExactSolution` `FALSE` to use an aproximated algorithm to fit the FMM model with restrictions (default). If `TRUE` is specified, the exact solution is computed.
Details

Data will be collected over \( n_{\text{Periods}} \) periods. When \( n_{\text{Periods}} > 1 \) the fitting is carried out by averaging the data collected at each time point across all considered periods. The model is fitting to summarized data. \texttt{timePoints} is a \( n \)-length numeric vector where \( n \) is the number of different time points per period.

The \texttt{stopFunction} argument can either be the functions \texttt{alwaysFalse} or \texttt{R2} included in the package or user-defined functions that have the same arguments. The included functions serve for the following:

- \texttt{alwaysFalse()}, its default value, which returns \texttt{FALSE} to force \texttt{maxiter} iterations; and
- \texttt{R2(vData,pred,prevPred,difMax = 0.001)}, a function that computes the difference between the explained variability in two consecutive iterations returning \texttt{TRUE} when the convergence criterion is reached. To calculate the explained variability difference, the data and the fitted values from the current and previous iteration are passed as arguments \texttt{vData}, \texttt{pred} and \texttt{prevPred}, respectively. The convergence criterion is fulfilled when the explained variability difference is less than the argument \texttt{difMax} (by default 0.001).

Value

An S4 object of class \texttt{'FMM'} with information about the fitted model. The object contains the following slots:

- \texttt{@timePoints} The time points as specified by the input argument. It is a numeric vector containing the time points at which each data of one single period is observed.
- \texttt{@data} The data as specified by the input argument. It is a numeric vector containing the data to be fitted a FMM model. Data could be collected over multiple periods.
- \texttt{@summarizedData} When the data has more than one period, a numeric vector containing data averaging the data at each time point across all considered periods.
- \texttt{@nPeriods} A numeric value containing the number of periods in data as specified by the input argument.
- \texttt{@fittedValues} A numeric vector of the fitted values by the FMM model.
- \texttt{@M} A numeric value of the estimated intercept parameter \( M \).
- \texttt{@A} A numeric value or vector of the estimated FMM wave amplitude parameter(s) \( A \).
- \texttt{@alpha} A numeric value or vector of the estimated FMM wave phase translation parameter(s) \( \alpha \).
- \texttt{@beta} A numeric value or vector of the estimated FMM wave skewness parameter(s) \( \beta \).
- \texttt{@omega} A numeric value or vector of the estimated FMM wave kurtosis parameter(s) \( \omega \).
- \texttt{@SSE} A numeric value of the sum of the residual squares values.
- \texttt{@R2} A numeric vector specifying the explained variance by each of the fitted FMM components.
- \texttt{@nIter} A numeric value specifying the number of iterations of the fitting algorithm.

References

Examples

# A monocomponent FMM model is fitted.
FMM_data <- generateFMM(2, 3, 1.5, 2.3, 0.1,
  from = 0, to = 2*pi, length.out = 100,
  outvalues = TRUE, sigmaNoise = 0.3, plot = FALSE)
fit <- fitFMM(FMM_data$y, lengthAlphaGrid = 10, lengthOmegaGrid = 10)
summary(fit)

# To see the differences between number of repetitions.
FMM_data <- generateFMM(2, 1, 1.5, 1.1, 0.14, outvalues = TRUE, sigmaNoise = 0.15, plot=TRUE)
fit1 <- fitFMM(FMM_data$y, lengthAlphaGrid = 6, lengthOmegaGrid = 3, numReps = 1)
fit2 <- fitFMM(FMM_data$y, lengthAlphaGrid = 6, lengthOmegaGrid = 3, numReps = 6,
  showProgress = FALSE)  # suppress progress messages
getSSE(fit1)
getSSE(fit2)

# Finer resolution grid.
fit3 <- fitFMM(FMM_data$y, lengthAlphaGrid = 10, lengthOmegaGrid = 5, numReps = 1)
getSSE(fit3)

# Two component FMM model with beta and omega restricted
restFMM2w_data <- generateFMM(M = 3, A = c(7, 4), alpha = c(0.5, 5), beta = c(rep(3, 2)),
  omega = rep(0.05, 2), from = 0, to = 2*pi, length.out = 100,
  sigmaNoise = 0.3, plot = FALSE)
fit2w.rest <- fitFMM(restFMM2w_data$y, nback = 2, maxiter = 1, numReps = 1,
  lengthAlphaGrid = 15, lengthOmegaGrid = 10,
  betaOmegaRestrictions = c(1, 1))
plotFMM(fit2w.rest, components = TRUE)

FMM-class

FMM Class Representation

Description

Class representation for an S4 object of class 'FMM'.

Value

The S4 object of class 'FMM' contains the following slots:

@timePoints The time points as specified by the input argument. It is a numeric vector containing the time points at which each data of one single period is observed.
@data The data as specified by the input argument. It is a numeric vector containing the data to be fitted a FMM model. Data could be collected over multiple periods.
@summarizedData When the data has more than one period, a numeric vector containing data averaging the data at each time point across all considered periods.
@nPeriods A numeric value containing the number of periods in data as specified by the input argument.
@fittedValues A numeric vector of the fitted values by the FMM model.
@M A numeric value of the estimated intercept parameter $M$.
@A A numeric value or vector of the estimated FMM wave amplitude parameter(s) $A$.
@alpha A numeric value or vector of the estimated FMM wave phase translation parameter(s) $\alpha$.
@beta A numeric value or vector of the estimated FMM wave skewness parameter(s) $\beta$.
@omega A numeric value or vector of the estimated FMM wave kurtosis parameter(s) $\omega$.
@SSE A numeric value of the sum of the residual squares values.
@R2 A numeric vector specifying the explained variance by each of the fitted FMM components.
@nIter A numeric value specifying the number of iterations of the fitting algorithm.

Examples

```r
## FMM class
getClass("FMM")
getSlots("FMM")
```

FMM-methods

Methods for objects of class 'FMM'

Description

The methods for objects of class 'FMM' are:

```r
coeff coef method for S4 class 'FMM', summary summary method for S4 class 'FMM', fitted fitted method for S4 class 'FMM', resid resid method for S4 class 'FMM'.
```

Usage

```r
## S4 coef method for signature 'FMM'
coef(object,...)

## S4 summary method for signature 'FMM'
summary(object,...)

## S4 fitted method for signature 'FMM'
fitted(object,...)
```
FMM-methods

## S4 resid method for signature 'FMM'

resid(object,...)

### Arguments

object  
object of class 'FMM'.

...  
additional arguments passed to the method.

### Value

- The function coef() returns a list with two components:
  - M  
    A numeric value. The estimated intercept parameter $M$.
  - wave  
    A data.frame with the estimates of each FMM wave parameters. It is organised as one component per row.

- The function summary() displays relevant results of the fitting. When it is assigned, this function returns a list with the following components:
  - coef  
    The list returns by the function coef().
  - peak.time  
    A data.frame with the estimates of the peak and trough times of each wave. It is organised as one component per row.
  - resid  
    The vector of residuals.
  - R.squared  
    A numerical value with the R squared of each wave. The total R square is computed as the sum of the contributions of each wave.

- The function fitted() returns a data.frame with two columns: timePoints and fittedValues.
- The function resid() returns a numeric vector with residuals of the model.

### Examples

```r
## Generate example data:
fmm2.data <- generateFMM(0, rep(2, 2), c(1.5, 3.4), c(0.2, 2.3), c(0.1, 0.2),
plot = FALSE, outvalues = TRUE,
sigmaNoise = 0.5) # add a gaussian noise with sigma = 0.5

## Fit the FMM model with nback = 2 component
fit <- fitFMM(vData = fmm2.data$y,timePoints = fmm2.data$t,nback = 2,
lengthAlphaGrid = 24,lengthOmegaGrid = 10)

## Extract coefficients of the model:
coef(fit)

## Summarize results:
summary(fit)

## Results on a list:
res <- summary(fit)
res$peak.time # fiducial points
```
generateFMM

Simulating data from FMM models

Description

generateFMM() simulates data from a FMM model defined by parameters $M$, $A$, $\alpha$, $\beta$ and $\omega$.

Usage

generateFMM(
  M,
  A,
  alpha,
  beta,
  omega,
  from = 0,
  to = 2 * pi,
  length.out = 200,
  plot = TRUE,
  outvalues = TRUE,
  sigmaNoise = 0
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>A numeric vector which contains the value of the intercept parameter $M$.</td>
</tr>
<tr>
<td>A</td>
<td>A positive numeric vector which contains the value of the FMM wave amplitude parameter $A$.</td>
</tr>
<tr>
<td>alpha</td>
<td>A numeric vector which contains the value of the FMM wave phase translation parameter $\alpha$.</td>
</tr>
<tr>
<td>beta</td>
<td>A numeric vector which contains the value of the FMM wave skewness parameter $\beta$.</td>
</tr>
<tr>
<td>omega</td>
<td>A numeric vector which contains the value of the FMM wave kurtosis parameter $\omega$. omega parameter must be between 0 and 1.</td>
</tr>
<tr>
<td>from</td>
<td>A numeric value which contains the initial time point of the simulated data. By default, it is 0.</td>
</tr>
<tr>
<td>to</td>
<td>A numeric value which contains the final time point of the simulated data. By default, it is $2\pi$.</td>
</tr>
</tbody>
</table>
**generateFMM**

`length.out`  A non-negative number which contains the desired length of the simulation. By default, it is 100.

`plot`  A logical value indicating whether the simulated data should be drawn on a plot. By default, it is TRUE.

`outvalues`  A logical value indicating whether the numerical simulation should be return. By default, it is TRUE.

`sigmaNoise`  A non-negative number which contains the standard deviation of the gaussian noise to be added. Its default value is zero equivalent to a simulation set-up without noise.

**Details**

To simulate a multicomponent FMM model, arguments `A`, `alpha`, `beta` and `omega` are vectors of length `m`, where `m` represents the number of FMM waves. With different lengths, the smaller vectors will be replicate until they are the same length as the longest vector.

With `sigmaNoise = s, s>0`, the `generateFMM` function uses `rnorm(length.out, 0, sigmaNoise)` to create the normally distributed noise and adds it to the simulated values.

**Value**

When `outvalues = TRUE` a list of with the following components is returned:

- `input`  a list with the input parameters `M`, `A`, `alpha`, `beta` and `omega`.
- `t`  a numeric vector with the time points at each data is simulated.
- `y`  a numeric vector with the data simulated.

When `plot = TRUE` a scatter plot of `y` vs `t` is drawn.

**References**


**Examples**

```r
# Simulate data from a monocomponent FMM model. A plot with the simulated model is shown
generateFMM(M = 2, A = 3, alpha = 1.5, beta = 2.3, omega = 0.1, outvalues = FALSE)

# Add a gaussian noise with standard deviation 0.3. The numeric results are returned
generateFMM(M = 2, A = 3, alpha = 1.5, beta = 2.3, omega = 0.1,
            sigmaNoise = 0.3, plot = FALSE, outvalues = TRUE)

# Simulate data from a multicomponent FMM model with two FMM waves
# both with amplitude parameter = 2
generateFMM(M = 0, A = rep(2, 2), alpha = c(1.5, 3.4), beta = c(0.2, 2.3), omega = c(0.1, 0.2))
```
getFMMPeaks

Peak and trough times and signal values

Description

getFMMPeaks() is used to estimate peak and trough times and signal values at those times for each component of the model. These parameters result to be useful in multiple applications.

Usage

getFMMPeaks(objFMM, timePointsIn2pi = TRUE)

Arguments

objFMM Object of class 'FMM'
timePointsIn2pi TRUE to return peak and trough times in the \([0, 2\pi]\) interval. When timePointsIn2pi = FALSE the positions of peak and trough times are returned. Its default value is TRUE.

Value

A list with the following components is returned:
tpeakU a numeric vector with the time points at which the peak of each wave is estimated.
tpeakL a numeric vector with the time points at which the trough of each wave is estimated.
ZU a numeric vector with the estimated signal peak values of each wave.
ZL a numeric vector with the estimated signal trough values of each wave.

References


Examples

## Generate example data:
fmm2.data <- generateFMM(0, rep(2, 2), c(1.5, 3.4), c(0.2, 2.3), c(0.1, 0.2), plot = FALSE, outvalues = TRUE, sigmaNoise = 0.5) # add a gaussian noise with sigma = 0.5

## Fit the FMM model with nback = 2 components
## fit is an object of S4 class 'FMM'
fit <- fitFMM(fmm2.data$y, timePoints = fmm2.data$t, nback = 2,
Description

A collection of functions to extract slots from S4 objects of class 'FMM'.

The extractor functions are:

- `getTimePoints` Extracts the timePoints slot from a S4 object of class 'FMM'.
- `getData` Extracts the data slot from a S4 object of class 'FMM'.
- `getSummarizedData` Extracts the summarizedData slot from a S4 object of class 'FMM'.
- `getNPeriods` Extracts the nPeriods slot from a S4 object of class 'FMM'.
- `getFittedValues` Extracts the fittedValues slot from a S4 object of class 'FMM'.
- `getM` Extracts the M slot from a S4 object of class 'FMM'.
- `getA` Extracts the A slot from a S4 object of class 'FMM'.
- `getAlpha` Extracts the alpha slot from a S4 object of class 'FMM'.
- `getBeta` Extracts the beta slot from a S4 object of class 'FMM'.
- `getOmega` Extracts the omega slot from a S4 object of class 'FMM'.
- `getSSE` Extracts the SSE slot from a S4 object of class 'FMM'.
- `getR2` Extracts the R2 slot from a S4 object of class 'FMM'.
- `getNIter` Extracts the nIter slot from a S4 object of class 'FMM'.

Usage

```
getM(objFMM)
getOmega(objFMM)
getData(objFMM)
```

Arguments

- `objFMM` an object of class of class 'FMM'.

Value

Return the content of the corresponding slot.
mouseGeneExp  
*Iqgap2* gene expression data from mouse liver

### Description

Data from High-temporal resolution profiling of mouse liver. Samples were collected every hour for 48 hours from 3-5 mice per time point from liver. Samples were pooled and analyzed using Affymetrix arrays.

### Usage

```r
data(mouseGeneExp)
```

### Format

A numeric vector.

### Source


### References


### Examples

```r
data(mouseGeneExp)
str(mouseGeneExp)
```

neuronalAPTrain  
*Neuronal AP Train Data simulated with Hodgkin-Huxley model*

### Description

Voltage data in mV simulated with Hodgkin Huxley model (parameters: C=1, gNa=232, gK=45, gL=0.215, vK=-12, vNa=115, vL=10.6, bar(alphaN)=0.95, bar(betaN)=1.3, bar(alphaM)=1, bar(betaM)=1.15, bar(alphaH)=1, bar(betaH)=1) and applied current of 4.5 microA 1 millisecond. The simulation has been done with a modified NeuroDynex Python module.

### Usage

```r
data(neuronalAPTrain)
```
neuronalSpike

Format
A numeric vector.

Source

References

Examples
data(neuronalAPTrain)
str(neuronalAPTrain)

neuronalSpike Neuronal Spike Data simulated with Hodgkin-Huxley model

Description
Voltage data in mV simulated with Hodgkin Huxley model (parameters: C=1, gNa=260, gK=30, gL=0.31, vK=-12, vNa=115, vL=10.6, bar(alphaN)=1.15, bar(betaN)=0.85, bar(alphaM)=0.9, bar(betaM)=1.3, bar(alphaH)=1, bar(betaH)=1) and applied current of 12 microA 1 millisecond. The simulation has been done with a modified NeuroDynex Python module.

Usage
data(neuronalSpike)

Format
A numeric vector.

Source

References

Examples
data(neuronalSpike)
str(neuronalSpike)
plotFMM is used to plot fitted FMM models. The function can either plot the fitted model against the data or each of the components of the model separately. Optionally 'ggplot2' can be used as graphic library.

Usage

plotFMM(
  objFMM,
  components = FALSE,
  plotAlongPeriods = FALSE,
  use_ggplot2 = FALSE,
  legendInComponentsPlot = TRUE,
  textExtra = ""
)

Arguments

objFMM Object of class FMM
components A logical value indicating if the centered wave components of the model should be separately plotted (case where it is TRUE). If FALSE, the default, the fitted FMM model along with the observed data is plotted.
plotAlongPeriods A logical value indicating if more than one period should be plotted in the plots by default. Its default value is FALSE.
use_ggplot2 A logical value. If FALSE, the default, R base graphics are used. If TRUE, 'ggplot2' library is used as graphics engine.
legendInComponentsPlot A logical value indicating whether the legend should be plotted in the components plot. By defaults it is TRUE.
textExtra A character vector for extra text to be added to the titles of the plots.

Details

plotFMM() can generate two types of plots: the basic plot compares the fitted model against the original data while the components plot represents separately the centered waves of the model (if the argument components is TRUE).

The function is also capable of plotting multiple periods if the data has more than one, as is the case in many applications such as chronobiology. In this case, the argument plotAlongPeriods should be TRUE. In the case of components plots the value taken by the latter argument is ignored as they are plotted along just one period.
While, by default, plots are created using base R graphics, 'ggplot2' can also be used for more aesthetic and customizable plots. Optional arguments legendInComponentsPlot and textExtra serve to control, respectively, whether a legend to the components plot should be added and adding extra text to the plot’s title.

Value

None if base R graphics are used, a named ggplot2 list if 'ggplot2' is used.

Examples

# Simulates an scenario in which an FMM model is suitable,
res <- generateFMM(2,3,1.5,2.3,0.1,outvalues = TRUE,sigmaNoise = 0.3, plot=FALSE)
# then a FMM model is fitted to the data.
fit <- fitFMM(res$y, lengthAlphaGrid=20,lengthOmegaGrid=12)
plotFMM(fit)

# Components plot of FMM Model fitted to neuronal data with various optional aesthetics
data("neuronalSpike")
fittedFMM2<fitFMM(neuronalSpike, nback=2,
   lengthAlphaGrid = 24,lengthOmegaGrid = 10, numReps = 1)
plotFMM(fittedFMM2, components = TRUE)
plotFMM(fittedFMM2, components = TRUE,
   legendInComponentsPlot = FALSE,
   textExtra = "Neuronal Data")

# With ggplot2, customizable plots can be created,
library(ggplot2)
# standard plots
plotFMM(fittedFMM2, use_ggplot2 = TRUE)
# modify x-axis with original timePoints
timePoints <- getTimePoints(fittedFMM2)
nObs <- length(timePoints)
sTimePoints <- round(c(1, nObs*0.25, nObs*0.5, nObs*0.75, nObs))
plotFMM(fittedFMM2, use_ggplot2 = TRUE) +
   scale_x_continuous(breaks = sTimePoints,
   labels = function(x) round(timePoints[x],2))
# and components plots
plotFMM(fittedFMM2, components = TRUE, use_ggplot2 = TRUE)

# Plot of fitted model to more than one period.
data("mouseGeneExp")
fittedFMM2<-fitFMM(mouseGeneExp, nPeriods = 2,
   lengthAlphaGrid = 20,lengthOmegaGrid = 10)
plotFMM(fittedFMM2, plotAlongPeriods = TRUE)
Index

* datasets
  - ecgData, 3
  - mouseGeneExp, 14
  - neuronalAPTrain, 14
  - neuronalSpike, 15

* package
  - FMM-package, 2

coef (FMM-methods), 8
coef, FMM-method (FMM-methods), 8

ecgData, 3
extractWaves, 3

fitFMM, 4
fitted (FMM-methods), 8
fitted, FMM-method (FMM-methods), 8
FMM (FMM-package), 2
FMM-class, 7
FMM-methods, 8
FMM-package, 2

generateFMM, 10
geta (getS4), 13
getAlpha (getS4), 13
getBeta (getS4), 13
data (getS4), 13
getFittedValues (getS4), 13
getFMMPeaks, 12
data (getS4), 13
getNIter (getS4), 13
getNPeriods (getS4), 13
getOmega (getS4), 13
getR2 (getS4), 13
getS4, 13
getSSE (getS4), 13
getSummarizedData (getS4), 13
data (getS4), 13

mouseGeneExp, 14
neuronalAPTrain, 14
neuronalSpike, 15
plotFMM, 16
resid (FMM-methods), 8
resid, FMM-method (FMM-methods), 8
summary (FMM-methods), 8
summary, FMM-method (FMM-methods), 8