

# Package: flexmsm (via r-universe)

October 18, 2024

**Type** Package

**Title** A General Framework for Flexible Multi-State Survival Modelling

**Version** 0.1.2

**Description** A general estimation framework for multi-state Markov processes with flexible specification of the transition intensities. The log-transition intensities can be specified through Generalised Additive Models which allow for virtually any type of covariate effect. Elementary specifications such as time-homogeneous processes and simple parametric forms are also supported. There are no limitations on the type of process one can assume, with both forward and backward transitions allowed and virtually any number of states.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.2.3

**Imports** GJRM, mgcv, stats, trust, matrixStats, parallel

**Suggests** testthat (>= 3.0.0)

**Config/testthat/edition** 3

**Depends** R (>= 2.10)

**NeedsCompilation** no

**Author** Alessia Eletti [aut, cre], Giampiero Marra [aut], Rosalba Radice [aut]

**Maintainer** Alessia Eletti <alessia.eletti.19@ucl.ac.uk>

**Repository** CRAN

**Date/Publication** 2024-07-19 10:00:02 UTC

## Contents

flexmsm-package . . . . .	2
---------------------------	---

conv.check . . . . .	3
fMmsm . . . . .	4
fmsm . . . . .	5
fmsmObject . . . . .	9
IDM_cav . . . . .	10
LikGradapproxHess.general . . . . .	11
LikGradHess.CM . . . . .	12
LikGradHess.general . . . . .	13
logLik.fmsm . . . . .	15
P.pred . . . . .	15
plot.fmsm . . . . .	17
print.fmsm . . . . .	17
print.summary.fMmsm . . . . .	18
print.summary.fmsm . . . . .	18
Q.matr.setup.general . . . . .	19
Q.pred . . . . .	20
simulateIDM . . . . .	21
state.pairs.CT . . . . .	22
summary.fMmsm . . . . .	22
summary.fmsm . . . . .	23
<b>Index</b>	<b>24</b>

---

flexmsm-package	<i>flexmsm: Flexible Multi-State Modelling</i>
-----------------	------------------------------------------------

---

## Description

Provides a function for fitting any type of multistate survival model, with flexibly defined transition intensities and any type of observation scheme. The package also provides a host of tools for straightforward interpretation and visualisation of the fitted model.

The main fitting function is `fmsm`, which fits the multistate survival model, providing parameter and as key inference quantities (p-values, estimated degrees of freedom, ...), as well as the elements needed to obtain predicted transition intensities and probabilities, along with their confidence intervals.

The main auxiliary functions are `Q.pred` and `P.pred`.

## Details

Provides functions for fitting and interpreting the output of general flexible multistate survival models. The process is defined by means of a list of model specifications for the transition intensities, each of which follow syntax similar to that used for GAMs in `mgcv`.

The estimation approach is based on a carefully structured, stable penalised likelihood approach, with the smoothers (representing several types of covariate effects) set up using penalised regression splines. The numerical routine carries out function minimization using a trust region algorithm

in combination with an adaptation of an automatic multiple smoothing parameter estimation procedure for Generalised Additive Models (see `mgcv` for more details on this last point). The smooths supported by this package are those available in `mgcv`.

Confidence intervals for smooth components and nonlinear functions of the model parameters are derived using a Bayesian approach. P-values for testing individual smooth terms for equality to the zero function are also provided and based on the approach implemented in `mgcv`. The usual plotting and summary functions are also available.

Plots of the estimated transition intensities and transition probabilities can be obtained along with their respective confidence intervals. This includes 3D plots when two-dimensional splines are included in the model specification of one or more transition intensities.

### Author(s)

Alessia Eletti (University College London, Department of Statistical Science), Giampiero Marra (University College London, Department of Statistical Science) and Rosalba Radice (Cass Business School, City, University of London).

Maintainer: Alessia Eletti <alessia.eletti.19@ucl.ac.uk>

### References

Eletti, A., Marra, G., Radice, R., (submitted), A General Estimation Framework for Multi-State Markov Processes with Flexible Specification of the Transition Intensities.

### See Also

[fmsm](#)

---

conv.check

*Convergence diagnostics on fitted model output.*

---

### Description

Convergence diagnostics on fitted model output.

### Usage

```
conv.check(object, also.unpen = FALSE)
```

### Arguments

<code>object</code>	Fitted model object.
<code>also.unpen</code>	If TRUE, displays eigenvalues range also for the unpenalised Hessian. Defaults to FALSE.

### Value

Convergence diagnostics.

---

fMmsm	<i>Flexible transition intensity based models for two dependant multi-state processes</i>
-------	-------------------------------------------------------------------------------------------

---

## Description

XXXX.

## Usage

```
fMmsm(formula1, data1, id1, state1,
      params1 = NULL, spP1 = NULL, constraint1 = NULL,
      formula2, data2, id2, state2,
      params2 = NULL, spP2 = NULL, constraint2 = NULL,
      phi = NULL,
      pmethod = 'eigendecomp',
      aggregate = TRUE, sp.method = 'perf', iterlimsp = 50,
      Q.diagnostics = TRUE, iterlim = 100, verbose,
      tols = 1e-7, tols.EFS = 0.1, parallel = FALSE, no_cores = 2)
```

## Arguments

formula1	Model specification for the transition intensities of the first process.
data1	Dataset of the first process.
id1	Name of the variable in the dataset representing the unique code associated with each patient in the first process.
state1	Name of the variable in the first process dataset representing the state occupied by the patient at the given time.
params1	XXX.
spP1	Smoothing parameter for the first process.
constraint1	XXX.
formula2	Model specification for the transition intensities of the second process.
data2	Dataset of the second process.
id2	Name of the variable in the dataset representing the unique code associated with each patient in the second process.
state2	Name of the variable in the second process dataset representing the state occupied by the patient at the given time.
params2	XXX.
spP2	Smoothing parameter for the second process.
constraint2	XXX.
phi	XXX.
pmethod	Which method should be used for the computation of the transition probability matrix. Available options are

- 'eigendecomp' (default): this method is based on the eigendecomposition of the transition intensity matrix (from Kalbfleisch & Lawless 1985);
- 'analytic': uses analytic expressions of the transition probabilities, obtained by solving the Kolmogorov forward differential equation, only implemented for IDMs for now;
- 'scaling&squaring': this is the scaling and squaring method implemented as proposed in Fung (2004). This is inefficient, so its use is not recommended. Can be used to investigate convergence errors.

aggregate	Whether or not data should be aggregated (this slightly improves efficiency as redundancies in the data are eliminated). The default is TRUE.
sp.method	Method to be used for smoothing parameter estimation. The default is magic, the automatic multiple smoothing parameter selection algorithm. Alternatively, efs can be used for the Fellner-Schall method. To suppress the smoothing parameter estimation set this to NULL.
iterlimsp	Maximum allowed iterations for smoothing parameter estimation.
Q.diagnostics	If TRUE, diagnostics information on the Q matrix are saved. The default TRUE.
iterlim	Maximum allowed iterations for trust region algorithm.
verbose	XXX.
tolsp	Convergence criterion used in magic based smoothing parameter estimation.
tolsp.EFS	Convergence criterion used in efs based smoothing parameter estimation.
parallel	If TRUE parallel computing is used during estimation. This can only be used by Windows users for now.
no_cores	Number of cores used if parallel computing chosen. The default is 2. If NULL, all available cores are used.

### Value

The function returns an object of class `fmsm` as described in `fmsmObject`.

---

fmsm	<i>Flexible transition intensity based models for univariate multistate processes</i>
------	---------------------------------------------------------------------------------------

---

### Description

Fits a flexible multistate survival model. Any type of process is supported, including both forward and backward transitions, and must be specified by providing a list of equations, one for each transition intensity allowed. Any type of observation scheme is allowed: the process can be observed in continuous time, intermittently at fixed times, there can be an absorbing state as well as censored states. Virtually any type of covariate effects are supported and can be specified by means of splines, with the same syntax used to specify Generalised Additive Models (GAMs) in R.

**Usage**

```
fmsm(formula, data, id, state, death, pmethod = 'eigendecomp',
      aggregate = TRUE, params.0 = NULL, sp.0 = NULL,
      constraint = NULL, sp.method = 'perf', iterlimsp = 50,
      Q.diagnostics = TRUE, fit = TRUE, iterlim = 100,
      tolspace = 1e-7, tolspace.EFS = 0.1, parallel = FALSE, no_cores = 2,
      cens.state = NULL, living.exact = NULL, verbose = FALSE,
      justComp = NULL, approxHess = FALSE)
```

**Arguments**

formula	Model specification for the transition intensities.
data	Dataset.
id	Name of the variable in the dataset representing the unique code associated with each patient.
state	Name of the variable in the dataset representing the state occupied by the patient at the given time.
death	TRUE if the last state is an absorbing state, FALSE otherwise.
pmethod	Which method should be used for the computation of the transition probability matrix. Available options are <ul style="list-style-type: none"> <li>'eigendecomp' (default): this method is based on the eigendecomposition of the transition intensity matrix (from Kalbfleisch &amp; Lawless 1985);</li> <li>'analytic': uses analytic expressions of the transition probabilities, obtained by solving the Kolmogorov forward differential equation, only implemented for IDMs for now;</li> <li>'scaling&amp;squaring': this is the scaling and squaring method implemented as proposed in Fung (2004). This is inefficient, so its use is not recommended. Can be used to investigate convergence errors.</li> </ul>
aggregate	Whether or not data should be aggregated (this slightly improves efficiency as redundancies in the data are eliminated). The default is TRUE.
params.0	Starting values for the model parameters. Defaults to NULL, i.e. they are computed internally.
sp.0	Starting values for the smoothing parameters. Defaults to NULL, i.e. they are computed internally.
constraint	A list containing the constraints to be applied to the model parameters. For example, assuming a process with three transitions, <code>constraint = list(x1 = c(1, 1, 1), x2 = c(1, 2, 2))</code> means that the effect of covariate x1 is constrained to be equal across the three transitions and that the effects of x2 on the second and third transitions are constrained to be equal, but the effect on the first transition is left unconstrained.
sp.method	Method to be used for smoothing parameter estimation. The default is <code>magic</code> , the automatic multiple smoothing parameter selection algorithm. Alternatively, <code>efs</code> can be used for the Fellner-Schall method. To suppress the smoothing parameter estimation set this to NULL.

iterlimsp	Maximum allowed iterations for smoothing parameter estimation.
Q.diagnostics	If TRUE, diagnostics information on the Q matrix are saved. The default TRUE.
fit	If FALSE, fitting is not carried. May be useful to extract model setup quantities.
iterlim	Maximum allowed iterations for trust region algorithm.
tolsp	Convergence criterion used in magic based smoothing parameter estimation.
tolsp.EFS	Convergence criterion used in efs based smoothing parameter estimation.
parallel	If TRUE parallel computing is used during estimation. This can only be used by Windows users for now.
no_cores	Number of cores used if parallel computing chosen. The default is 2. If NULL, all available cores are used.
cens.state	Code used in the dataset to indicate the censored states.
living.exact	Name of the variable in the dataset indicating whether an observation is exactly observed or not.
verbose	If TRUE, prints the convergence criterion obtained at each iteration of the full algorithm. The default is FALSE.
justComp	Can be c('lik', 'grad', 'hess') (or a subvector of this) to compute the log-likelihood, gradient and Hessian at the starting parameter value, without carrying out fitting. Defaults to NULL.
approxHess	If TRUE an approximation of the Hessian based on the outer product of the gradient vector is computed. The default is FALSE.

### Value

The function returns an object of class `fmsm` as described in `fmsmObject`.

### Examples

```
## Not run:

#####
# MULTISTATE SURVIVAL MODELLING with CAV DATA ####
#####

library(flexmsm)

# Import data
Data <- IDM_cav

# MODEL SPECIFICATION ####
formula <- list(years ~ s(years, bs = 'cr', k = 10) + dage + pdiag, # 1-2
                years ~ s(years, bs = 'cr', k = 10) + dage + pdiag, # 1-3
                0, # 2-1
                years ~ s(years, bs = 'cr', k = 10) + dage + pdiag, # 2-3
                0, # 3-1
                0 # 3-2
                )
```

```

# Counts of pairs of consecutive states observed (C = counts, T = times)
counts.CT <- state.pairs.CT(formula = formula, data = Data, time = 'years',
                             state = 'state', id = 'PTNUM')

counts.CT$counts

# MODEL FITTING ###

# NOTE ***
# Takes about 18 minutes on a machine with Windows 10,
# Intel 2.20 GHz core, 16 GB of RAM and 8 cores, using all cores.
# The default is to use 2 cores, this takes about 26 minutes.
# To use all available cores on your device input no_cores = NULL.
# ****

fmsm.out <- fmsm(formula = formula, data = Data,
                 id = 'PTNUM', state = 'state', death = TRUE,
                 fit = TRUE, parallel = TRUE, no_cores = 2,
                 pmethod = 'analytic')

print(fmsm.out)

AIC(fmsm.out)
BIC(fmsm.out)

# FITTING SUMMARY ####
summary(fmsm.out)
conv.check(fmsm.out)

#####
# VISUALISATION ####
#####

# PLOT THE SMOOTHS OF TIME FOR EACH TRANSITION ####
# par(mfrow = c(1,3))
plot(fmsm.out)

# Consider a patient with:
dage.pred <- 16      # - 16 year old donor
pdiag.pred <- 0      # - IDC as principal diagnosis
start.pred <- 0      # - start observation at time t = 0
stop.pred <- 15      # - t = 15 years for time horizon
n.pred <- 21         # - 21 time points
no.state.pred <- -13 # - (because we don't need this, so anything is fine)

newdata <- data.frame(PTNUM = rep(1, n.pred),
                      years = seq(start.pred, stop.pred, length.out = n.pred),
                      state = rep(no.state.pred, n.pred),
                      dage = rep(dage.pred, n.pred), pdiag = rep(pdiag.pred, n.pred))

```



```

# ESTIMATED TRANSITION INTENSITIES #####

# Plot of estimated transition intensities
# par(mfrow = c(1,3))
Q.hat <- Q.pred(fmsm.out, newdata = newdata, get.CI = TRUE, plot.Q = TRUE, rug = TRUE,
               ylim = c(0, 1.5))

# Estimated transition intensity matrix at, e.g., t = 0
round(Q.hat$Q.hist[, ,1], 3)

# ESTIMATED TRANSITION PROBABILITIES #####

# Plot of estimated transition probabilities
# par(mfrow = c(2,3))
P.hat <- P.pred(fmsm.out, newdata = newdata, get.CI = TRUE, plot.P = TRUE, rug = TRUE)

# Estimated 15 year transition probability matrix
round(P.hat$P.pred, 3)
# e.g., there is a 6.2% chance of observing CAV onset 15 years after transplant

## End(Not run)

```

---

fmsmObject

*Fitted fmsmObject object*


---

## Description

The `fmsm` function returns the fitted model object `fmsmObject`. This is of class "fmsm" and includes the components listed below. These are intended for confident users. To extract results from the fitted model objects, functions such as `summary.fmsm`, `plot.fmsm`, `Q.pred` and `P.pred` should be used instead.

## Value

<code>suStf</code>	A list with all of the quantities used for estimation and post-estimation computations. This includes the full design matrix <code>full.X</code> , the starting parameters used, <code>params.0</code> and <code>sp.0</code> , and more technical quantities such as the positions of the smooths' parameters and of the parametric coefficients.
<code>msm.fit.object</code>	This contains all of the details of the model fitting.
<code>msm.post.object</code>	This contains all of the post-estimation details.
<code>formula</code>	Formula used in the model specification.

<code>short.formula</code>	Short version of the model specification, i.e. only non-zero transition specifications are included.
<code>n</code>	Number of observations in the dataset.
<code>N</code>	Number of unique individuals.
<code>logLik</code>	The value of the log-likelihood at convergence.
<code>t.edf</code>	Total effective degrees of freedom.
<code>singleComp</code>	If present, log-likelihood, gradient and Hessian computed at the starting parameter (without fitting).

**See Also**

[fmsm](#), [summary.fmsm](#)

---

IDM\_cav

*Cardiac allograft vasculopathy (CAV) data*


---

**Description**

A series of approximately yearly angiographic examinations of heart transplant recipients. Onset of cardiac allograft vasculopathy, a deterioration of the arterial wall, and death are monitored. This is a subset of data from the `cav` dataset in R package `msm`.

**Usage**

```
IDM_cav
```

**Format**

A data frame with 2,803 observations of 614 patients and 5 variables. These are grouped by patient number and ordered by years after transplant.

**PTNUM** Unique number identifying each patient.

**years** Examination time (years after transplant).

**state** State of the examination. State 1 represents no CAV, state 2 represents CAV, state 3 represents death.

**dage** Age of the heart donor (years).

**pdiag** Primary diagnosis (reason for transplant). IHD = ischaemic heart disease, IDC = idiopathic dilated cardiomyopathy.

**Source**

Papworth Hospital, U.K.

---

 LikGradapproxHess.general

*Likelihood, gradient and Hessian for univariate transition intensity based models*

---

## Description

Likelihood, gradient and Hessian for univariate transition intensity based models

## Usage

```
LikGradapproxHess.general(
  params,
  data = NULL,
  full.X = NULL,
  MM,
  pen.matr.S.lambda,
  aggregated.provided = FALSE,
  do.gradient = TRUE,
  do.hessian = TRUE,
  pmethod = "analytic",
  death,
  Qmatr.diagnostics.list = NULL,
  verbose = FALSE,
  parallel = FALSE,
  no_cores = 2
)
```

## Arguments

params	Parameters vector.
data	Dataset in proper format.
full.X	Full design matrix.
MM	List of necessary setup quantities.
pen.matr.S.lambda	Penalty matrix multiplied by smoothing parameter lambda.
aggregated.provided	Whether aggregated form was provided (may become obsolete in the future if we see original dataset as special case of aggregated where nrep = 1).
do.gradient	Whether or not to compute the gradient.
do.hessian	Whether or not to compute the Hessian.
pmethod	Method to be used for computation of transition probability matrix. See help of <code>msm()</code> for further details.
death	Whether the last state is an absorbing state.

<code>Qmatr.diagnostics.list</code>	List of maximum absolute values of the Q matrices computed during model fitting.
<code>verbose</code>	Whether to print out the progress being made in computing the likelihood, gradient and Hessian.
<code>parallel</code>	Whether or not to use parallel computing (only for Windows users for now).
<code>no_cores</code>	Number of cores used if parallel computing chosen. The default is 2. If NULL, all available cores are used.

**Value**

Penalized likelihood, gradient and Hessian associated with model at given parameters, for use by trust region algorithm.

---

LikGradHess.CM	<i>Likelihood, gradient and Hessian for univariate transition intensity based models for the base dependence model.</i>
----------------	-------------------------------------------------------------------------------------------------------------------------

---

**Description**

Likelihood, gradient and Hessian for univariate transition intensity based models for the base dependence model.

**Usage**

```
LikGradHess.CM(
  params,
  data = NULL,
  full.X = NULL,
  MM,
  pen.matr.S.lambda,
  aggregated.provided = FALSE,
  do.gradient = TRUE,
  do.hessian = TRUE,
  pmethod = "analytic",
  death,
  Qmatr.diagnostics.list = NULL,
  verbose = FALSE,
  parallel = FALSE,
  no_cores = 2,
  CM.comp = TRUE,
  P.save.all = FALSE
)
```

**Arguments**

params	Parameters vector.
data	Dataset in proper format.
full.X	Full design matrix.
MM	List of necessary setup quantities.
pen.matr.S.lambda	Penalty matrix multiplied by smoothing parameter lambda.
aggregated.provided	Whether aggregated form was provided (may become obsolete in the future if we see original dataset as special case of aggregated where nrep = 1).
do.gradient	Whether or not to compute the gradient.
do.hessian	Whether or not to compute the Hessian.
pmethod	Method to be used for computation of transition probability matrix. See help of msm() for further details.
death	Whether the last state is an absorbing state.
Qmatr.diagnostics.list	List of maximum absolute values of the Q matrices computed during model fitting.
verbose	Whether to print out the progress being made in computing the likelihood, gradient and Hessian.
parallel	Whether or not to use parallel computing (only for Windows users for now).
no_cores	Number of cores used if parallel computing chosen. The default is 2. If NULL, all available cores are used.
CM.comp	If TRUE compute the Copula contribution from each individual for the base dependence model. The default is TRUE.
P.save.all	If TRUE save the P matrices computed at each observation. The default is FALSE.

**Value**

Penalized likelihood, gradient and Hessian associated with model at given parameters, for use by trust region algorithm.

---

LikGradHess.general *Likelihood, gradient and Hessian for univariate transition intensity based models*

---

**Description**

Likelihood, gradient and Hessian for univariate transition intensity based models

**Usage**

```
LikGradHess.general(
  params,
  data = NULL,
  full.X = NULL,
  MM,
  pen.matr.S.lambda,
  aggregated.provided = FALSE,
  do.gradient = TRUE,
  do.hessian = TRUE,
  pmethod = "analytic",
  death,
  Qmatr.diagnostics.list = NULL,
  verbose = FALSE,
  parallel = FALSE,
  no_cores = 2
)
```

**Arguments**

<code>params</code>	Parameters vector.
<code>data</code>	Dataset in proper format.
<code>full.X</code>	Full design matrix.
<code>MM</code>	List of necessary setup quantities.
<code>pen.matr.S.lambda</code>	Penalty matrix multiplied by smoothing parameter lambda.
<code>aggregated.provided</code>	Whether aggregated form was provided (may become obsolete in the future if we see original dataset as special case of aggregated where <code>nrep = 1</code> ).
<code>do.gradient</code>	Whether or not to compute the gradient.
<code>do.hessian</code>	Whether or not to compute the Hessian.
<code>pmethod</code>	Method to be used for computation of transition probability matrix. See help of <code>msm()</code> for further details.
<code>death</code>	Whether the last state is an absorbing state.
<code>Qmatr.diagnostics.list</code>	List of maximum absolute values of the Q matrices computed during model fitting.
<code>verbose</code>	Whether to print out the progress being made in computing the likelihood, gradient and Hessian.
<code>parallel</code>	Whether or not to use parallel computing (only for Windows users for now).
<code>no_cores</code>	Number of cores used if parallel computing chosen. The default is 2. If NULL, all available cores are used.

**Value**

Penalized likelihood, gradient and Hessian associated with model at given parameters, for use by trust region algorithm.

---

logLik.fmsm	<i>Extract the log likelihood for the fitted multistate model</i>
-------------	-------------------------------------------------------------------

---

**Description**

It extracts the log-likelihood for a fitted fmsm model.

**Usage**

```
## S3 method for class 'fmsm'
logLik(object, ...)
```

**Arguments**

object	Fitted model object of class fmsm produced by function <a href="#">fmsm</a> .
...	Unused in this case.

**Value**

Standard logLik object.

---

P.pred	<i>Predict and plot the transition probabilities</i>
--------	------------------------------------------------------

---

**Description**

Function to predict and plot the estimated transition probabilities (and confidence intervals).

**Usage**

```
P.pred(object, newdata, get.CI = TRUE,
        n.sim.CI = 1000, prob.lev = 0.05,
        plot.P = FALSE, which.plots = NULL,
        rug = FALSE, params.0 = NULL, ...)
```

**Arguments**

<code>object</code>	Fitted model object.
<code>newdata</code>	Dataframe containing the profile for which one wished to obtain the predicted transition probabilities.
<code>get.CI</code>	Whether to compute the confidence intervals.
<code>n.sim.CI</code>	Number of simulations to be used for confidence intervals computation.
<code>prob.lev</code>	Probability level of confidence intervals.
<code>plot.P</code>	Whether to output plots of transition probabilities.
<code>which.plots</code>	Number between 1 and the maximum number of non-null transition probabilities. This can be used if only some plots are to be plotted.
<code>rug</code>	Whether to include a rugplot of the observed transition times.
<code>params.0</code>	Parameter value at which the transition probability matrix needs to be computed. By default this is NULL and the MLE found in the fitted model object <code>object</code> is used. If both a fitted model and a manually inputted parameters vector are provided, the former will be ignored and the P matrix will be computed at the <code>params.0</code> value.
<code>...</code>	Other graphical arguments.

**Value**

	Estimated transition probabilities (and confidence intervals).
<code>P.pred</code>	Predicted transition probability matrix corresponding to the time horizon specified in <code>newdata</code> . This is a <code>nstates x nstates</code> matrix.
<code>P.CI.lower</code>	Matrix containing the lower bounds of the confidence intervals for the predicted transition probability matrix.
<code>P.CI.upper</code>	Matrix containing the upper bounds of the confidence intervals for the predicted transition probability matrix.
<code>P.hist</code>	List of predicted transition probability matrices computed at each time point specified in <code>newdata</code> . This is a <code>nstates x nstates x n.pred</code> array, where <code>n.pred</code> is the number of rows in <code>newdata</code> .
<code>P.CI.lower.hist</code>	List of matrices containing the lower bounds of the confidence intervals for each predicted transition probability matrix in <code>P.hist</code> .
<code>P.CI.upper.hist</code>	List of matrices containing the upper bounds of the confidence intervals for each predicted transition probability matrix in <code>P.hist</code> .
<code>full.X</code>	Full design matrix corresponding to the <code>newdata</code> provided.
<code>P.sim.hist</code>	List of transition probability matrices simulated to obtain the confidence intervals at each time point from <code>newdata</code> . May be useful to quickly obtain intervals for a different confidence level.

**See Also**

[fmsm](#)



---

plot.fmsm	<i>Function to plot the smooths included in the model specifications.</i>
-----------	---------------------------------------------------------------------------

---

**Description**

Function to plot the smooths included in the model specifications.

**Usage**

```
## S3 method for class 'fmsm'  
plot(x, ...)
```

**Arguments**

x	Fitted model object.
...	Other graphical arguments.

**Value**

Plots the smooths.

---

print.fmsm	<i>Print a fmsm object</i>
------------	----------------------------

---

**Description**

The print method for the fmsmObject produced by [fmsm](#).

**Usage**

```
## S3 method for class 'fmsm'  
print(x, ...)
```

**Arguments**

x	fmsm object produced by function <a href="#">fmsm</a> .
...	Unused in this case.

**Value**

print.fmsm prints out a matrix summarising the positions of the transition intensities, the transition intensities formulae, the total number of observations, etc for the fitted multistate survival model.

---

```
print.summary.fMmsm  Flexible transition intensity based models for univariate multistate processes
```

---

### Description

Flexible transition intensity based models for univariate multistate processes

### Usage

```
## S3 method for class 'summary.fMmsm'
print(
  x,
  digits = max(3, getOption("digits") - 3),
  signif.stars = getOption("show.signif.stars"),
  ...
)
```

### Arguments

x	Fitted model object.
digits	Number of digits printed in the output.
signif.stars	By default significance stars are printed alongside the output.
...	Other arguments.

### Value

Prints model term summaries.

---

```
print.summary.fmsm  Flexible transition intensity based models for two dependant multistate processes
```

---

### Description

Flexible transition intensity based models for two dependant multistate processes

### Usage

```
## S3 method for class 'summary.fmsm'
print(
  x,
  digits = max(3, getOption("digits") - 3),
  signif.stars = getOption("show.signif.stars"),
  ...
)
```

**Arguments**

x	Fitted model object.
digits	Number of digits printed in the output.
signif.stars	By default significance stars are printed alongside the output.
...	Other arguments.

**Value**

Prints model term summaries.

---

Q.matr.setup.general *Internal function*

---

**Description**

Internal function needed for setup of Q matrix and its first and second derivative.

**Usage**

```
Q.matr.setup.general(
  params,
  nstates,
  full.X,
  start.pos.par,
  l.short.formula,
  whereQ,
  firstD = TRUE,
  secondD = TRUE,
  bound.eta = FALSE,
  pos.optparams,
  pos.optparams2
)
```

**Arguments**

params	Parameters vector.
nstates	Number of states.
full.X	Full design matrix.
start.pos.par	Positions within full parameters vector of starting point for each sub-parameters vector corresponding to each transition intensity specification.
l.short.formula	Number of transitions.
whereQ	Positions within Q matrix of not-null transition intensities.
firstD	Whether the first derivative of the Q matrix should be computed.

secondD	Whether the second derivative of the Q matrix should be computed.
bound.eta	Whether to bound the additive predictor, defaults to FALSE. This is only used for debugging purposes, do not change.
pos.optparams	Vector with positions of parameters vector in the form used by the optimization algorithm (i.e. when one or more parameters are constrained to be equal these will only appear once).
pos.optparams2	Like pos.optparams but the count is not stopped at the constrained parameters.

**Value**

Q matrix and its first and second derivatives with respect to the parameters vector.

---

Q.pred	<i>Predict and plot the transition intensities</i>
--------	----------------------------------------------------

---

**Description**

Function to predict and plot the estimated transition intensities (and confidence intervals).

**Usage**

```
Q.pred(object, newdata, get.CI = TRUE,
       n.sim.CI = 1000, prob.lev = 0.05,
       plot.Q = FALSE, which.plots = NULL,
       cond.list.2d = NULL, plot.Q.2d = FALSE,
       rug = TRUE, ...)
```

**Arguments**

object	Fitted model object.
newdata	Dataframe containing the profile for which one wished to obtain the predicted transition intensities.
get.CI	Whether to compute the confidence intervals.
n.sim.CI	Number of simulations to be used for confidence intervals computation.
prob.lev	Probability level of confidence intervals.
plot.Q	Whether to output plots of transition intensities.
which.plots	Number between 1 and the maximum number of non-null transition intensities. This can be used if only some plots are to be plotted.
cond.list.2d	Value of covariate(s) to be kept fixed in the plotting of 3D-transition intensities.
plot.Q.2d	Whether to plot 3D transition intensities (only valid if 2D-smooths are present).
rug	Whether to include a rugplot of the observed transition times.
...	Other graphical parameters.

**Value**

Estimated transition intensities (and confidence intervals).

Q.hist	List of predicted transition intensity matrices computed at each time point specified in newdata. This is a $nstates \times nstates \times n.pred$ array, where $n.pred$ is the number of rows in newdata.
Q.CI.lower	Matrix containing the lower bounds of the confidence intervals for the predicted transition intensity matrix.
Q.CI.upper	Matrix containing the upper bounds of the confidence intervals for the predicted transition intensity matrix.
full.X	Full design matrix corresponding to the newdata provided.
Q.sim.hist	List of transition intensity matrices simulated to obtain the confidence intervals at each time point from newdata. May be useful to quickly obtain intervals for a different confidence level.

**See Also**

[fmsm](#)

---

simulateIDM	<i>Function to predict and plot the estimated transition intensities (and confidence intervals).</i>
-------------	------------------------------------------------------------------------------------------------------

---

**Description**

Function to predict and plot the estimated transition intensities (and confidence intervals).

**Usage**

```
simulateIDM(N = N, seed = seed, og.12 = TRUE)
```

**Arguments**

N	Total number of individuals.
seed	Seed used for the simulation.
og.12	If TRUE a common shape for the first transition is used. If FALSE a sinusoid is used.

**Value**

Simulated data generated from an Illness-Death model (IDM).

---

state.pairs.CT	<i>Function to extract state pair counts and observed (right) times.</i>
----------------	--------------------------------------------------------------------------

---

**Description**

Function to extract state pair counts and observed (right) times.

**Usage**

```
state.pairs.CT(
  formula = NULL,
  data = NULL,
  whereQ = NULL,
  nstates = NULL,
  time = NULL,
  state = NULL,
  id = NULL
)
```

**Arguments**

formula	Model specification.
data	Data.
whereQ	Placement of allowed transition intensities. Only for internal use. Defaults to NULL and is obtained automatically when formula is provided.
nstates	Total number of states. Only for internal use. Defaults to NULL and is obtained automatically when formula is provided.
time	Name of variable containing the time-to-event.
state	Name of variable containing the states.
id	Name of variable containing the unique code identifying the individuals.

**Value**

A table with the state-pair counts and a list with the observed (right) times for each transition.

---

summary.fMmsm	<i>Summary for fitted model output.</i>
---------------	-----------------------------------------

---

**Description**

Summary for fitted model output.

**Usage**

```
## S3 method for class 'fMmsm'  
summary(object, ...)
```

**Arguments**

object	Fitted model object.
...	Other arguments.

**Value**

Summary of fitted model object.

---

summary.fmsm

*Summary for fitted model output.*

---

**Description**

Summary for fitted model output.

**Usage**

```
## S3 method for class 'fmsm'  
summary(object, ...)
```

**Arguments**

object	Fitted model object.
...	Other arguments.

**Value**

Summary of fitted model object.

# Index

- \* **datasets**
    - IDM\_cav, 10
  - \* **flexible**
    - flexmsm-package, 2
  - \* **intensities**
    - flexmsm-package, 2
  - \* **intermittently**
    - flexmsm-package, 2
  - \* **multistate**
    - flexmsm-package, 2
  - \* **observed**
    - flexmsm-package, 2
  - \* **package**
    - flexmsm-package, 2
  - \* **penalised**
    - flexmsm-package, 2
  - \* **regression**
    - flexmsm-package, 2
  - \* **smooth**
    - flexmsm-package, 2
  - \* **spline**
    - flexmsm-package, 2
  - \* **transition**
    - flexmsm-package, 2
- conv.check, 3
- flexmsm (flexmsm-package), 2
- flexmsm-package, 2
- fMmsm, 4
- fmsm, 2, 3, 5, 9, 10, 15–17, 21
- fmsmObject, 9
- IDM\_cav, 10
- LikGradapproxHess.general, 11
- LikGradHess.CM, 12
- LikGradHess.general, 13
- logLik.fmsm, 15
- P.pred, 2, 9, 15
- plot.fmsm, 9, 17
- print.fmsm, 17
- print.summary.fMmsm, 18
- print.summary.fmsm, 18
- Q.matr.setup.general, 19
- Q.pred, 2, 9, 20
- simulateIDM, 21
- state.pairs.CT, 22
- summary.fMmsm, 22
- summary.fmsm, 9, 10, 23