
GAMLSS R Packages Reference Card

Contents

1	GAMLSS R Packages
2	Fitting or Updating a Model (in <code>gamlss</code> package)
3	Extracting Information from the Fitted Model
4	Selecting a Model
5	Plotting and Diagnostics
6	Centile Estimation
7	Additive Terms
8	<code>gamlss.family</code> Distributions
9	Package <code>gamlss.mx</code> Functions
10	Package <code>gamlss.tr</code> Functions
11	Package <code>gamlss.cens</code> Functions
12	Package <code>gamlss.nl</code> Functions
13	Package <code>gamlss.demo</code> Functions
14	Package <code>gamlss.util</code> Functions
15	The <code>gamlss</code> Function Arguments

1 GAMLSS R Packages

1	The GAMLSS framework comprises ten different packages written in R, i.e. the original <code>gamlss</code> package and six add-on packages
2	<ol style="list-style-type: none">1. the original <code>gamlss</code> package for fitting a GAMLSS model
2	<ol style="list-style-type: none">2. the <code>gamlss.add</code> package for extra additive terms.
2	<ol style="list-style-type: none">3. the <code>gamlss.cens</code> package for fitting censored (left, right or interval) response variables.
3	<ol style="list-style-type: none">4. the <code>gamlss.data</code> package for data used
3	<ol style="list-style-type: none">5. the <code>gamlss.demo</code> package useful for teaching purposes (i.e. plotting the distributions interactively)
3	<ol style="list-style-type: none">6. the <code>gamlss.dist</code> package for <code>gamlss.family</code> distributions
5	<ol style="list-style-type: none">7. the <code>gamlss.mx</code> package for fitting finite mixture distributions.
5	<ol style="list-style-type: none">8. the <code>gamlss.nl</code> package for fitting non-linear models
5	<ol style="list-style-type: none">9. the <code>gamlss.tr</code> package for fitting truncated distributions.
5	<ol style="list-style-type: none">10. the <code>gamlss.util</code> package for additional utilities
5	The GAMLSS framework packages can be downloaded and installed from CRAN, the R library at http://www.r-project.org/ . Test versions may be found at the GAMLSS web site at http://www.gamlss.com/ .
6	

2 Fitting or Updating a Model (in `gamlss` package)

`gamlss()` for fitting and creating a `gamlss` object

`refit()` to refit a `gamlss` object (i.e. continue iterations) if it has not converged

`update()` to update a given `gamlss` model object

`histDist()` to fit a parametric distribution to a single (response) variable and plot simultaneously a histogram and the fitted distribution of this variable

3 Extracting Information from the Fitted Model

`AIC()` to extract the generalized Akaike information criterion (GAIC) from a fitted `gamlss` model object

`GAIC()` identical to `AIC`

`coef()` to extract the linear coefficients from a fitted `gamlss` model object

`deviance()` to extract the global deviance of the `gamlss` model object

`edf()` to extract the effective degrees of freedom from a fitted `gamlss` model object

`edfAll()` to extract the effective degrees of freedom for all the parameters from a fitted `gamlss` model object

`extractAIC()` to extract the generalized Akaike information criterion from a fitted `gamlss` model object

`fitted()` to extract the fitted values from a fitted `gamlss` model object

`formula()` to extract a model formula

`fv()` to extract the fitted values for a distribution parameter (see also `fitted()` and `lpred()`)

`logLik()` to extract the log likelihood

`lp()` to extract the linear predictor for a distribution parameter (see also `lpred`)

`lpred()` to extract the fitted values, linear predictor or specified terms (with standard errors) for a distribution parameter.

`model.frame()` to extract the model frame of a specified distribution parameter

`model.matrix()` to extract the design matrix of a specified distribution parameter

`predict()` to predict from new data individual distribution parameter values (see also `lpred` below)

`predictAll()` to predict from new data all the distribution parameter values

`print()` : to print a `gamlss` object

`residuals()` to extract the normalized (randomized) quantile residuals from a fitted `gamlss` model object. See Dunn and Smyth (1996) for a definition of the normalized (randomized) quantile residuals.

`summary()` to summarize the fit in a `gamlss` object

`terms()` to extract terms from a `gamlss` object

`vcov()` to extract the variance-covariance matrix of the beta estimates (for all distribution parameter models).

4 Selecting a Model

`add1()` to add a single term, from those supplied, to a fitted `gamlss` model object

`addterm()` to add a single term, from those supplied, to a fitted `gamlss` model object (used by `stepGAIC()` below).

`drop1()` to drop one term at the time from the fitted `gamlss` model object.

`dropterm()` to fit all models that differ from the current fitted `gamlss` model object by dropping a single term (used by `stepGAIC()` below).

`find.hyper()` to find the hyperparameters (e.g. degrees of freedom for smoothing terms and/or non-linear parameters) by minimizing the profile Generalized Akaike Information Criterion (GAIC) based on the global deviance

`gamlss.scope()` to define the scope for `stepGAIC()`

`stepGAIC()` to select explanatory terms using GAIC

`stepGAIC.CH()` to select (additive) using the method of Chambers and Hastie (1992).

`stepGAIC.VR()` to select (parametric) terms using method of Venables and Ripley (2002).

`stepGAICAll()` (experimental) to select explanatory terms for all the parameters using GAIC

`stepTGD()` (experimental) to select explanatory terms using the global deviance of a test data set.

`VGD()` to calculate the global deviance of model using the validation set data set, (where the training part of the data is used for fitting and the validation for calculating the global deviance).

`VGD1()` identical to `VGD()` but the output is a list rather than a values as in `VGD()`.

`VGD2()` identical to `VGD1()` but it takes as argument the new data, (`newdata`), rather than a factor which split the combined data in two as in functions `VGD()` or `VGD1()`.

`TGD()` to calculate the global deviance for new (test) data set given a fitted `gamlss` model.

5 Plotting and Diagnostics

`plot()` a plot of four graphs for the normalized (randomized) quantile residuals of a `gamlss` object. The residual plots are: (i) against an x-variable (ii) against the fitted values, (iii) a density plot and (iv) a QQ-plot. Note that residuals are randomized only for discrete response variables, see Dunn and Smyth (1996).

`par.plot()` for plotting parallel profile plots for individual participants in repeated measurement analysis

`pdf.plot()` for plotting the pdf functions for a given fitted `gamlss` object or a given `gamlss.family` distribution

`prof.dev()` for plotting the profile global deviance of one of the distribution parameters μ , σ , ν or τ .

`prof.term()` for plotting the profile global deviance of one of the model (beta) parameters. It can be also used to study the GAIC(\sharp) information profile of a hyperparameter for a given penalty \sharp for the GAIC.

`Q.stats()` for printing the Q statistics of Royston and Wright (2000).

`rqres.plot()` for plotting QQ-plots of different realizations of normalized randomized quantile residuals for a model with a discrete `gamlss.family` distribution.

`show.link()` for showing available link functions for distribution parameters in any `gamlss.family` distribution

`term.plot()` for plotting additive (smoothing) terms in any distribution parameter model

`wp()` worm plot of the residuals from a fitted `gamlss` object. See vanBuuren and Fredriks (2001).

6 Centile Estimation

`centiles()` to plot centile curves against an x-variable.

`centiles.com()` to compare centiles curves for more than one object.

`centiles.split()` as for `centiles()`, but splits the plot at specified values of x.

`centiles.pred()` to predict and plot centile curves for new x-values.

`fitted.plot()` to plot fitted values for all the parameters against an x-variable

7 Additive Terms

Additive terms	R Name
cubic splines	<code>cs()</code>
cycle P-spline	<code>pvc()</code>
varying coefficient	<code>pvc()</code>
penalized splines	<code>pb()</code>
loess	<code>lo()</code>
fractional polynomials	<code>fp()</code>
non-linear fit	<code>nl()</code>
random effects	<code>random()</code>
random effects	<code>ra()</code>
ridge regression	<code>ri()</code>
Simon Wood's gam	<code>ga()</code>

Table 1: Additive terms implemented within the `gamlss` packages

8 `gamlss.family` Distributions

The package `gamlss.dist` contains all the continuous, discrete and mixed distributions for `gamlss.family`. The distributions are shown in tables 2, 3 and 4 respectively.

Distributions	R Name
beta	BE()
Box-Cox Cole and Green	BCCG()
Box-Cox power exponential	BCPE()
Box-Cox t	BCT()
exponential	EXP()
exponential Gaussian	exGAUS()
exponential gen. beta type 2	EGB2()
gamma	GA()
generalized beta type 1	GB1()
generalized beta type 2	GB2()
generalized gamma	GG()
generalized inverse Gaussian	GIG()
generalized t	GT()
Gumbel	GU()
inverse Gaussian	IG()
Johnson's SU (μ the mean)	JSU()
Johnson's original SU	JSUo()
logistic	LO()
log normal	LOGNO()
log normal (Box-Cox)	LNO()
NET	NET()
normal	NO()
normal family	NOF()
power exponential	PE()
reverse Gumbel	RG()
skew power exponential type 1	SEP1()
skew power exponential type 2	SEP2()
skew power exponential type 3	SEP3()
skew power exponential type 4	SEP4()
sinh-arcsinh	SHASH()
skew t type 1	ST1()
skew t type 2	ST2()
skew t type 3	ST3()
skew t type 4	ST4()
skew t type 5	ST5()
t Family	TF()
Weibull	WEI()
Weibull (PH)	WEI2()
Weibull (μ the mean)	WEI3()

Table 2: Continuous distributions implemented within the **gamlss** packages

Distributions	R Name
beta binomial	BB()
binomial	BI()
logarithmic	LG()
Delaporte	DEL()
negative binomial type I	NBI()
negative binomial type II	NBII()
Poisson	PO()
Poisson inverse Gaussian	PIG()
Sichel	SI()
Sichel (μ the mean)	SICHEL()
zero altered beta binomial	ZABB()
zero altered binomial	ZABI()
zero altered logarithmic	ZALG()
zero altered neg. binomial	ZANBI()
zero altered poisson	ZAP()
zero inflated beta binomial	ZIBB()
zero inflated binomial	ZIBI()
zero inflated neg. binomial	ZINBI()
zero inflated poisson	ZIP()
zero inflated poisson (μ the mean)	ZIP2()
zero inflated poisson inv. Gaussian	ZIPIG()

Table 3: Discrete distributions implemented within the **gamlss** packages

beta inflated (at 0)	BEOI()
beta inflated (at 0)	BEINFO()
beta inflated (at 1)	BEZI()
beta inflated (at 1)	BEINF1()
beta inflated (at 0 and 1)	BEINF()
zero adjusted GA	ZAGA()
zero adjusted IG	ZAIG()

Table 4: Mixed distributions implemented within the **gamlss** packages

9 Package `gamlss.mx` Functions

`gamlss.MX()` for fitting a finite mixtures with no parameters in common (in `gamlss.mx`) package

`gamlss.NP()` for fitting a finite mixtures with parameters in common (in `gamlss.mx`) package

`nlgamlss()` for fitting a non-linear model (in `gamlss.nl`) package

10 Package `gamlss.tr` Functions

`gen.trun()` Generates a truncate distribution from a `gamlss.family`

`trun()` fits a truncate Distribution from a `gamlss.family`

`trun.d()` Truncated probability density function of a `gamlss.family` distribution

`trun.p()` Truncated cumulative density function of a `gamlss.family` distribution

`trun.q()` Truncated inverse cumulative density function of a `gamlss.family` distribution

`trun.r()` Generates random values from a truncated density function of a `gamlss.family` distribution

11 Package `gamlss.cens` Functions

`gen.cens()` Generates appropriate functions to be used to fit a censored response variable

`trun()` Fits a truncate Distribution from a `gamlss.family` distribution

`cens.d()` Generates a censored probability density function of a `gamlss.family` distribution

`cens.p()` Generates a censored cumulative density function of a `gamlss.family` distribution

`cens.q()` Generates a censored inverse cumulative density function of a `gamlss.family` distribution

12 Package `gamlss.nl` Functions

`nlgamlss()` Fitting a non-linear GAMLSS, (note that here the first argument is `y`, not `formula`)

13 Package `gamlss.demo` Functions

`demo.BetaSplines()` A demo for Beta splines.

`demo.discreteSmo()` A demo for the Whittaker smoothing method.

`demo.histSmo()` A demo for histogram smoothing method based on fitting a Poisson distribution.

`demo.interpolateSmo()` A demo for showing the interpolation and extrapolation on the Whittaker smoothing method.

`demo.PenSplines()` A demo for P-splines.

`demo.Locmean()` A demo for local mean smoothing.

`demo.Locpoly()` A demo for local polynomial smoothing.

`demo.WLocpoly()` A demo for weighted local polynomial smoothing

`demo.WLocmean()` A demo for weighted local mean (kernel) smoothing

`demo.NO()` A demo for the normal distribution. Every `gamlss.family` distribution has its own demo which can be access using `demo.NAME()` where `name` is a `gamlss.family` distribution.

14 Package `gamlss.util` Functions

`Locmean()` A functions to fit local mean smoother

`fitFixBP()` Functions to Fit Univariate Break Point Models.

`scattersmooth()` Two dimensional Smooth scatter plots

`penLS()` A function to fit penalised least squares

`penReg()` A function to fit penalised regression.

15 The `gamlss` Function Arguments

`formula` a model formula (including the response variable y) for the `mu` parameter (compulsory), e.g. $y \sim x$.

`sigma.formula` a model formula for `sigma`, e.g. $\sim x$

`nu.formula` a model formula for `nu`, e.g. $\sim x$

`tau.formula` a model formula formula for `tau`, e.g. $\sim x$

`family` a `gamlss.family` distribution family

`data` a data frame containing the variables occurring in the formulae

`weights` a vector of weights. `weights` can be used i) to weight out observations (with weights equal to 1 or 0) ii) for a weighted likelihood analysis (typically appropriate if weights represent frequencies). Any other use of the `weights` could have side effects.

`contrasts` list of contrasts to be used for some or all of the factors appearing as variables in the parameter(s) model formula.

`method` the algorithms for GAMLSS, i.e. `RS()`, `CG()` or `mixed()`.

`start.from` a fitted GAMLSS model from which to take the starting values for the current model

`mu.start` vector or scalar for initial values for the location parameter `mu`.

`sigma.start` vector or scalar for initial values for the scale parameter `sigma`.

`nu.start` vector or scalar of initial values for the shape parameter `nu`.

`tau.start` vector or scalar of initial values for `tau`.

`mu.fix` fixing the `mu` parameter

`sigma.fix` fixing the `sigma` parameter

`nu.fix` fixing the `nu` parameter

`tau.fix` fixing the `tau` parameter

`control` control parameters of the outer iterations algorithm (see `gamlss.control`) function

`i.control` control parameters of the inner iterations of the RS algorithm (see `glim.control`)

Mikis Stasinopoulos and Bob Rigby
London Metropolitan University
d.stasinopoulos@londonmet.ac.uk
r.rigby@londonmet.ac.uk