
GAMLSS R Packages Reference Card

Contents

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

1 GAMLSS R Packages

The GAMLSS framework comprises seven different packages written in R, i.e. the original `gamlss` package and six add-on packages

1. the original `gamlss` package for fitting a GAMLSS model
2. the `gamlss.boot` package for bootstrapping centiles.
3. the `gamlss.cens` package for fitting censored (left, right or interval) response variables.
4. the `gamlss.dist` package for additional distributions

5. the `gamlss.mx` package for fitting finite mixture distributions.
6. the `gamlss.nl` package for fitting non-linear models
7. the `gamlss.tr` package for fitting truncated distributions.

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/>.

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)
- `GAIC()` identical to `AIC`
- `coef()` to extract the linear coefficients
- `deviance()` to extract the global deviance
- `extractAIC()` to extract the generalized Akaike information criterion

`fitted()` to extract the fitted values

`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

`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

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

`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) terms using the method of Chambers and Hastie (1992).

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

`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 value as in `VGD()`.

`VGD2()` identical to `VGD1()` but it takes as argument the new data, (`newdata`), rather than a factor which splits 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($\#$) information profile of a hyperparameter for a given penalty $\#$ 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 van Buuren 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>
varying coefficient	<code>vc()</code>
penalized splines	<code>ps()</code>
<code>loess</code>	<code>lo()</code>
fractional polynomials	<code>fp()</code>
power polynomials	<code>pp()</code>
non-linear fit	<code>nl()</code>
random effects	<code>random()</code>
random effects	<code>ra()</code>
random coefficient	<code>rc()</code>

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

8 `gamlss.family` Distributions

You must download the package `gamlss.dist` to obtain all distributions.

9 Package `gamlss.mx` Functions

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

`gamlss.NP()` for fitting a finite mixtures model 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 truncated distribution from a `gamlss.family`

Distributions	R Name
beta	<code>BE()</code>
beta inflated (at 0)	<code>BE0I()</code>
beta inflated (at 1)	<code>BE1I()</code>
beta inflated (at 0 and 1)	<code>BEINF()</code>
Box-Cox Cole and Green	<code>BCCG()</code>
Box-Cox power exponential	<code>BCPE()</code>
Box-Cox t	<code>BCT()</code>
exponential	<code>EXP()</code>
exponential Gaussian	<code>exGAUS()</code>
exponential gen. beta type 2	<code>EGB2()</code>
gamma	<code>GA()</code>
generalized beta type 1	<code>GB1()</code>
generalized beta type 2	<code>GB2()</code>
generalized gamma	<code>GG()</code>
generalized inverse Gaussian	<code>GIG()</code>
generalized t	<code>GT()</code>
Gumbel	<code>GU()</code>
inverse Gaussian	<code>IG()</code>
Johnson's SU (μ the mean)	<code>JSU()</code>
Johnson's original SU	<code>JSUo()</code>
logistic	<code>LO()</code>
log normal	<code>LOGNO()</code>
log normal (Box-Cox)	<code>LNO()</code>
NET	<code>NET()</code>
normal	<code>NO()</code>
normal family	<code>NOF()</code>
power exponential	<code>PE()</code>
reverse Gumbel	<code>RG()</code>
skew power exponential type 1	<code>SEP1()</code>
skew power exponential type 2	<code>SEP2()</code>
skew power exponential type 3	<code>SEP3()</code>
skew power exponential type 4	<code>SEP4()</code>
sinh-arcsinh	<code>SHASH()</code>
skew t type 1	<code>ST1()</code>
skew t type 2	<code>ST2()</code>
skew t type 3	<code>ST3()</code>
skew t type 4	<code>ST4()</code>
skew t type 5	<code>ST5()</code>
t Family	<code>TF()</code>
Weibull	<code>WEI()</code>
Weibull (PH)	<code>WEI2()</code>
Weibull (μ the mean)	<code>WEI3()</code>
zero adjusted IG	<code>ZAIG()</code>

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

Distributions	R Name
beta binomial	BB()
binomial	BI()
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 inflated poisson	ZIP()
zero inflated poisson (μ the mean)	ZIP2()

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

trun() Fits a truncated distribution from a **gamlss.family**

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

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

trun.q() Truncated inverse cumulative distribution 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 censored 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 distribution function of a **gamlss.family** distribution

cens.q() Generates a censored inverse cumulative distribution 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 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**)

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