The gamlss packages, Diagnostics and Algorithms
Flexible Regression and Smoothing

Bob Rigby\textsuperscript{1}  Mikis Stasinopoulos\textsuperscript{1}

\textsuperscript{1}STORM, London Metropolitan University

XXV SIMPOSIO INTERNACIONAL DE ESTAD\'ESICA, Armenia, Colombia, June 2015
The `gamlss` packages, Diagnostics and Algorithms

Outline

1. The R packages
2. Diagnostics
   - Normalised quantile residuals
   - Worm plots
3. Algorithms
4. The `gamlss()` function
5. The `gamlss` package
   - Fitting or Updating a Model
   - Extracting Information
   - Selecting a Model
   - Diagnostics
   - Centile estimation
   - Other useful functions
6. End
The R packages

- **gamlss** the original package
- **gamlss.dist** all `gamlss.family` distributions
- **gamlss.data** different sets of data
- **gamlss.add** for extra additive terms
- **gamlss.cens** for censored (left, right or interval) response variables
- **gamlss.demo** demos for distributions and smoothing
  - **gamlss.nl** non-linear term fitting
  - **gamlss.tr** generating truncated distributions
  - **gamlss.mx** finite mixtures distributions and random effects
- **gamlss.spatial** for spatial models
- **gamlss.util** for extra utilities
Normalised quantile residuals-continuous response

If \( y_i \) is an observation from a continuous response variable then

\[
\hat{u}_i = F(y_i | \hat{\theta}_i)
\]

where \( u_i = F(y_i | \theta_i) \) is the assumed cumulative distribution function for case \( i \).

\[
r_i = \Phi^{-1}(u_i)
\]

So \( r_i \) will have an approximately standard normal distribution

\[
r_i = "z-scores"
\]
Normalised quantile residuals-continuous response
Normalised quantile residuals-discrete response

If $y_i$ is an observation from a discrete integer response then

$$
\hat{u}_i
$$

is a random value from the uniform distribution on the interval

$$
[u_1, u_2] = \left[ F(y_i - 1|\hat{\theta}^i), F(y_i|\hat{\theta}^i) \right]
$$

$$
\hat{r}_i = \Phi^{-1}(\hat{u}_i)
$$
Normalised quantile residuals-continuous response
Worm plots: using `plot()`

- Against Fitted Values
- Against index
- Density Estimate
- Normal Q-Q Plot
Worm plots: using `wp()` van Buuren and Fredriks [2001]
Worm plots: different types

(a) resid mean too small

(b) resid mean too large

(c) resid variance too high

(d) resid variance too low

(e) resid negative skewness

(f) resid positive skewness

(g) resid lepto–kyrtotic

(h) resid platy–kyrtotic
Multiple worm plots against different range of x-variable

Given : xvar

Unit normal quantile

Deviation

Gamls packages, Diagnostics and Algorithms
Diagnostics
Worm plots
Multiple worm plots against different range of two x-variables
Worm plots are useful in:

- checking the assumed distribution of the response variable,
- checking whether the distribution is fitted correctly in all part of explanatory variable(s)
Algorithms: GAMLSS model

\[ y \overset{\text{ind}}{\sim} D(\mu, \sigma, \nu, \tau) \]

\[ g_1(\mu) = X_1\beta_1 + s_{11}(x_{11}) + \ldots + s_{1J_1}(x_{1J_1}) \]

\[ g_2(\sigma) = X_2\beta_2 + s_{21}(x_{21}) + \ldots + s_{2J_2}(x_{2J_2}) \]

\[ g_3(\nu) = X_3\beta_3 + s_{31}(x_{31}) + \ldots + s_{3J_3}(x_{3J_3}) \]

\[ g_4(\tau) = X_4\beta_4 + s_{41}(x_{41}) + \ldots + s_{4J_4}(x_{4J_4}) \]
The `gamlss` packages, Diagnostics and Algorithms

Algorithms: GAMLSS model random effects

\[ y \overset{\text{ind}}{\sim} D(\mu, \sigma, \nu, \tau) \]

\[ g_1(\mu) = X_1\beta_1 + Z_{11}\gamma_{11} + \ldots + Z_{1k_1}\gamma_{1J_1} \]

\[ g_2(\sigma) = X_2\beta_2 + Z_{21}\gamma_{21} + \ldots + Z_{2k_2}\gamma_{2J_2} \]

\[ g_3(\nu) = X_3\beta_3 + Z_{31}\gamma_{31} + \ldots + Z_{3k_3}\gamma_{3J_3} \]

\[ g_4(\tau) = X_4\beta_4 + Z_{41}\gamma_{41} + \ldots + Z_{4k_4}\gamma_{4J_4} \]
The likelihood:

\[ \ell = \sum_{i=1}^{n} \log f(y_i|\mu_i, \sigma_i, \nu_i, \tau_i) \]

The penalised log-likelihood:

\[ \ell_p = \ell - \frac{1}{2} \sum_{k=1}^{4} \sum_{j=1}^{J_k} \lambda_{kj} \gamma_{kj}^\top G_{kj} \gamma_{kj} \]

we will need estimates for the ‘betas’, the ‘gammas’ and the ‘lambdas’

\[ \beta = (\beta_1, \beta_2, \beta_3, \beta_4) \]

\[ \gamma = (\gamma_{11}, \ldots, \gamma_{1J_1}, \gamma_{21}, \ldots, \gamma_{4J_4}) \]

\[ \lambda = (\lambda_{11}, \ldots, \lambda_{1J_1}, \lambda_{21}, \ldots, \lambda_{4J_4}) \]
Algorithms: Estimating $\beta$ and $\gamma$ for fixed $\lambda$

- **RS** a generalization of the MADAM algorithm, Rigby and Stasinopoulos (1996a)
- **CG** a generalization of Cole and Green (1992) algorithm
- **mixed** a mixture of RS+CG (i.e. $j$ iterations of RS, followed by $k$ iterations of CG)
The `gamlss` packages, Diagnostics and Algorithms

Algorithms

(a) RS

(b) CG
Algorithms: The RS algorithm

- the outer iteration
- the inner iteration (or local scoring or GLIM algorithm)
- the modified backfitting algorithm
The `gamlss` packages, Diagnostics and Algorithms

Algorithms: outer iteration
Algorithms: inner iteration

Given the current $\hat{\mu}, \hat{\sigma}, \hat{\nu}, \hat{\tau}$

1. Calculate $z_k$ and $w_k$
2. Fit the linear explanatory variables and smoothers to $z_k$ with weights $w_k$ using modified backfitting and recalculate $\hat{\eta}_k$ and $\hat{\theta}_k$

No -> global deviance converged

Yes -> finish
Algorithms: modified backfitting

given current $z_k$, $w_k$, $\hat{\gamma}_{k1}$ and $\hat{\gamma}_{k2}$ for parameter $\theta_k$

1. Calculate $\tau = z_k - Z_{k1}\hat{\gamma}_{k1} - Z_{k2}\hat{\gamma}_{k2}$
2. Fit WLS to $\tau$ against $X_k$ using weights $w_k$, to get $\hat{\beta}_k$
3. Calculate $\tau = z_k - X_k\hat{\beta}_k - Z_{k2}\hat{\gamma}_{k2}$
4. Fit PWLS to $\tau$ against $Z_{k1}$ using weights $w_k$ to get new $\hat{\gamma}_{k1}$
5. Calculate $\tau = z_k - X_k\hat{\beta}_k - Z_{k1}\hat{\gamma}_{k1}$
6. Fit PWLS to $\tau$ against $Z_{k2}$ using weights $w_k$ to get new $\hat{\gamma}_{k2}$

Do the parameters $\hat{\beta}_k$, $\hat{\gamma}_{k1}$, $\hat{\gamma}_{k2}$ change?

Yes

No

Finish
Advantages of algorithms

1. flexible modular fitting procedure
2. easy implementation of new distributions
3. easy implementation of new additive terms
4. simple starting values for \((\mu, \sigma, \nu, \tau)\) easily found
5. stable and reliable algorithms
6. fast fitting (for fixed hyperparameters)
Algorithms: Estimating $\lambda$

**locally:** when the method of estimation of each $\lambda_{kj}$ is applied each time within the backfitting algorithm.

**globally:** when the method is applied outside the RS or CG GAMLSS algorithm.

Different methodologies for estimating the smoothing hyper-parameters:

- Generalised cross validation (GCV),
- Generalised Akaike information criterion (GAIC), and
- Maximum likelihood based methods (ML/REML).
The `gamlss()` function

```r
gamlss(formula = ~1, sigma.formula} = ~1,
       nu.formula = ~1, tau.formula = ~1,
       family = NO(),
       data = sys.parent(), weights = NULL,
       contrasts = NULL, method = RS(), start.from = NULL,
       mu.start = NULL, sigma.start = NULL,
       nu.start = NULL, tau.start = NULL,
       mu.fix = FALSE, sigma.fix = FALSE, nu.fix = FALSE,
       tau.fix = FALSE, control = gamlss.control(...),
       i.control = glim.control(...), ...)
Arguments of the `gamlss()` function

- **formula**: $y \sim x_1 + x_3$
- **sigma.fo**: $\sim x_1$
- **nu.fo**: $\sim x_2$
- **tau.fo**: $\sim 1$
- **data**: `abdom`
- **family**: `LO`
- **weights**: `freq`
- **method**: `mixed(10,50)`
- **control**: `gamlss.control(trace=FALSE)`
The `gamlss` packages, Diagnostics and Algorithms

The `gamlss()` function

Starting values

Generally are not needed:

```r
start.mu = 2 or start.mu = fitted(m1, "mu")
```

The same applies for other parameters

```r
start.sigma, start.nu, start.tau
```

**Starting from a previous model**

```r
start.from = model1
```
The `gamlss` package

Available functions

- Fitting or Updating a Model
- Extracting Information from the Fitted Model
- Selecting a Model
- Plotting and Diagnostics
- Centile Estimation
The `gamlss` packages, Diagnostics and Algorithms

The `gamlss` package

Fitting or Updating a Model

Fitting or Updating a Model

gamlss() for fitting and creating a `gamlss` object
refit() to refit a `gamlss` object (i.e. continue iterations)
update() to update a given `gamlss` model object
gamlssML() fitting a parametric distribution to a single (response) variable
histDist() to fit and plot a parametric distribution
fitDist() select a parametric distribution from an appropriate list
Extracting Information from the Fitted Model

- **GAIC()** generalised Akaike information criterion (or AIC)
- **coef()** the linear coefficients
- **deviance()** the global deviance $-2 \log L$
- **fitted()** the fitted values for a distribution parameter
- **predict()** to predict from new data individual distribution parameter values
- **predictAll()** to predict from new data all the distribution parameter values
- **print()** to print a `gamlss` object
- **residuals()** to extract the normalised (randomised) quantile residuals
- **summary()** to summarise the fit in a `gamlss` object
- **vcov()** to extract the variance-covariance matrix of the beta estimates.
Selecting a Model

- `add1()` `drop1()` to add or drop a single term
- `find.hyper()` to find the hyper-parameters
- `stepGAIC()` to select explanatory terms in one parameter
- `stepGAICAll.A()` to select explanatory terms in all the parameters (strategy A)
- `stepGAICAll.B()` to select explanatory terms in all the parameters (strategy B)
- `stepTGD()` selecting variables using a test set the global deviance for new (test) data set given a fitted gamlss model.
The gamlss packages, Diagnostics and Algorithms

The gamlss package
Diagnostics

Diagnostics

plot() a plot of four graphs for the normalized (randomized) quantile residuals

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

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

rqres.plot() for plotting QQ-plots of different realisations of randomised residuals (for discrete distributions)

wp() worm plot of the residuals from a fitted gamlss object

dtop() detrended Own’s plot of the residuals
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.
- `centiles.fan()` fan plot od centile curves
- `fitted.plot()` to plot fitted values for all the parameters against an x-variable
- `lms()` a function trying to automate the process of fitting growth curves
Other useful functions

prof.dev()  the profile global deviance of one of the distribution parameters

prof.term() for plotting the profile global deviance of one of the model (beta) parameters

show.link() for showing available link functions

term.plot() for plotting additive (smoothing) terms

gen.likelihood() generates the likelihood from a GAMLSS fitted model [used in vcov()]
END

for more information see

www.gamlss.org