BAMLSS

Bayesian Additive Models for Location Scale and Shape (and Beyond)

Nikolaus Umlauf

http://eeecon.uibk.ac.at/~umlauf/

Overview

- Introduction
- Distributional regression
- Lego toolbox
- R package **bamlss**
- Example

Introduction

A **not** complete list of software packages dealing with Bayesian regression models:

- **bayesm**, univariate and multivariate, SUR, multinomial logit, ...
- **bayesSurv**, survival regression, ...
- **MCMCpack**, linear regression, logit, ordinal probit, probit, Poisson regression, ...
- **MCMCglmm**, generalized linear mixed models (GLMM).
- **spikeSlabGAM**, Bayesian variable selection, model choice, in generalized additive mixed models (GAMM), ...
- **gammSlice**, generalized additive mixed models (GAMM).
- **BayesX**, structured additive distributional regression (STAR), ...
- **INLA**, generalized additive mixed models (GAMM), ...
- **WinBUGS**, **JAGS**, **STAN**, general purpose sampling engines.

⋮

Introduction

Most Bayesian software packages provide support for the estimation of so called mixed models (random effects), i.e., incorporating linear predictors of the form

$$\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} + \mathbf{U}\boldsymbol{\gamma},$$

where $\mathbf{X}\boldsymbol{\beta}$ are fixed effects, e.g., $p(\boldsymbol{\beta}) \propto$ const, and $\mathbf{U}\boldsymbol{\gamma}$ are the random effects, $\boldsymbol{\gamma} \sim N(\mathbf{0}, \mathbf{Q}(\boldsymbol{\tau}^2))$.

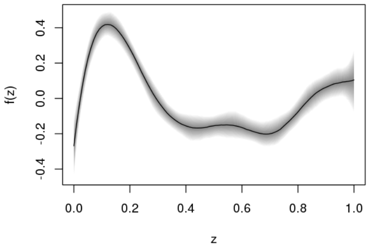
Few Bayesian software packages provide support for the estimation of semiparametric regression models with structured additive predictor

$$\eta = f_1(\mathbf{z}) + \ldots + f_p(\mathbf{z}) + \mathbf{x}^\top \boldsymbol{\beta},$$

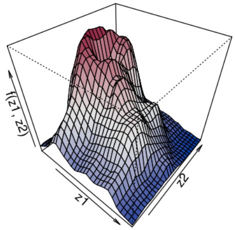
where f_j are possibly smooth functions and $\mathbf{z}$ represents a generic vector of all nonlinear modeled covariates.

Introduction

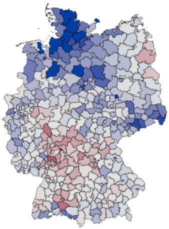
Nonlinear effects of continuous covariates



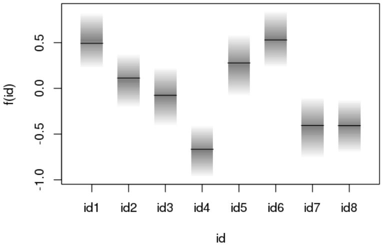
Two-dimensional surfaces



Spatially correlated effects f(z) = f(s)



Random intercepts

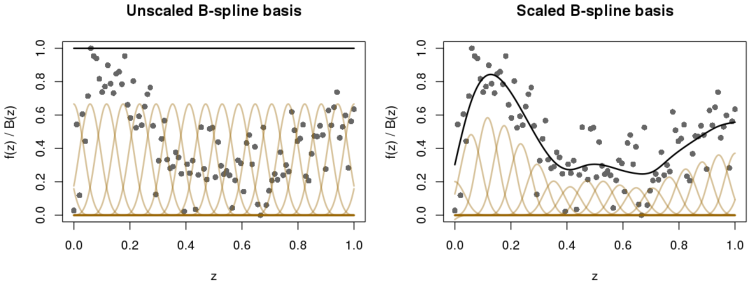


STAR Models

Within the basis function approach, the vector of function evaluations $\mathbf{f}_j = (f_j(\mathbf{z}_1), \dots, f_j(\mathbf{z}_n))$ of the $i = 1, \dots, n$ observations can be written in matrix notation

$$\mathbf{f}_j = \mathbf{Z}_j \gamma_j,$$

with $\mathbf{Z}_j$ as the design matrix, where γ_j are unknown regression coefficients. Form of $\mathbf{Z}_j$ only depends on the functional type chosen.



Introduction

Penalized least squares:

$$\mathrm{PLS}(\gamma, \boldsymbol{\lambda}) = ||\mathbf{y} - \boldsymbol{\eta}||^2 + \lambda_1 \gamma_1' \mathbf{K}_1 \gamma_1 + \ldots + \lambda_p \gamma_p' \mathbf{K}_p \gamma_p.$$

A general Prior for γ in the corresponding Bayesian approach

$$p(\gamma_j | \tau_j^2) \propto \exp\left(-\frac{1}{2\tau_j^2} \gamma_j' \mathbf{K}_j \gamma_j\right),$$

τ_j^2 variance parameter, governs the smoothness of f_j.

Structure of $\mathbf{K}_j$ also depends on the type of covariates and on assumptions about smoothness of $\mathbf{f}_j$.

The variance parameter τ_j^2 is equivalent to the inverse smoothing parameter in a frequentist approach.

Introduction

However, any basis function representation can be transformed into a mixed model representation

$$\mathbf{f}_j = \mathbf{Z}_j \boldsymbol{\gamma}_j = \mathbf{Z}_j(\tilde{\mathbf{X}}\boldsymbol{\beta} + \tilde{\mathbf{U}}\tilde{\boldsymbol{\gamma}}) = \mathbf{X}\boldsymbol{\beta} + \mathbf{U}\tilde{\boldsymbol{\gamma}},$$

with fixed effects $\boldsymbol{\beta}$ and random effects $\tilde{\boldsymbol{\gamma}} \sim N(\mathbf{0}, \tau^2\mathbf{I})$.

So the number of software packages that can estimate semiparametric models is actually quite large.

The number of different models that can be fit with these engines is even larger.

Introduction

The basic ideas are:

- Design a framework that makes it (a) easy to use different estimation engines and (b) fit models with a **structured additive predictor**.
- Therefore, we need to employ symbolic descriptions that do **not** restrict to any specific type of model and term structure.
- I.e., the aim is to use specialized/optimized engines to apply Bayesian **structured additive distributional regression** a.k.a. Bayesian additive models for location scale and shape (**BAMLSS**) and beyond.
- The approach should have **maximum flexibility/extendability**, also concerning functional types.

Distributional regression

Within this framework any parameter of a population distribution may be modeled by explanatory variables

$$\mathbf{y} \sim \mathcal{D}\left(g_1(\boldsymbol{\theta}_1) = \boldsymbol{\eta}_1,\ g_2(\boldsymbol{\theta}_2) = \boldsymbol{\eta}_2, \ldots,\ g_K(\boldsymbol{\theta}_K) = \boldsymbol{\eta}_K\right),$$

where $\mathcal{D}$ denotes any parametric distribution available for the response variable.

Each parameter is linked to a structured additive predictor

$$g_k(\boldsymbol{\theta}_k) = \boldsymbol{\eta}_k = \mathbf{Z}_{1k}\boldsymbol{\gamma}_{1k} + \ldots + \mathbf{Z}_{pk}\boldsymbol{\gamma}_{pk} + \mathbf{X}_k\boldsymbol{\beta}_k,\ \ k = 1, \ldots, K,$$

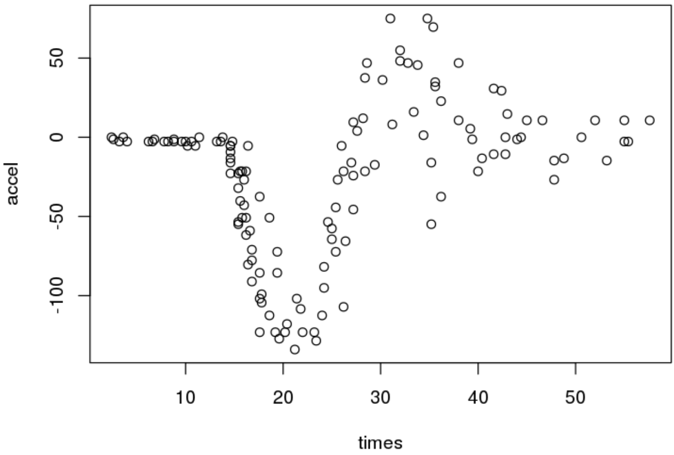
where $g_k(\cdot)$ are known monotonic link functions.

The observations y_i are assumed to be independent and conditional on a pre-specified parametric density $f(y_i|\boldsymbol{\theta}_{i1}, \ldots, \boldsymbol{\theta}_{iK})$.

Distributional regression

Example: Head acceleration in a simulated motorcycle accident

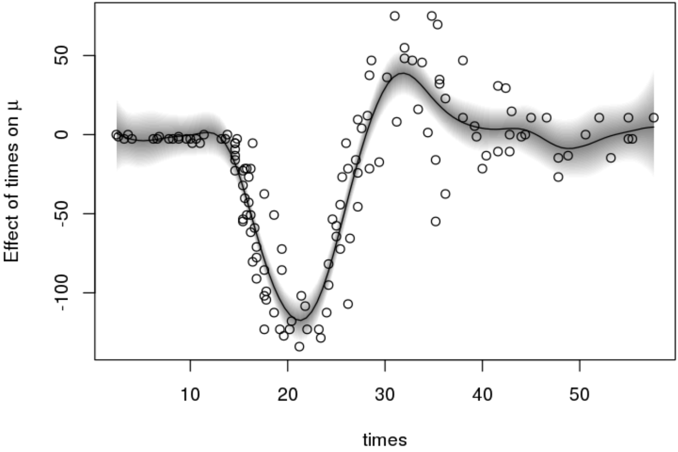
$$\texttt{accel} \sim N(\boldsymbol{\mu}, \boldsymbol{\sigma}^2).$$



Distributional regression

Example: Head acceleration in a simulated motorcycle accident

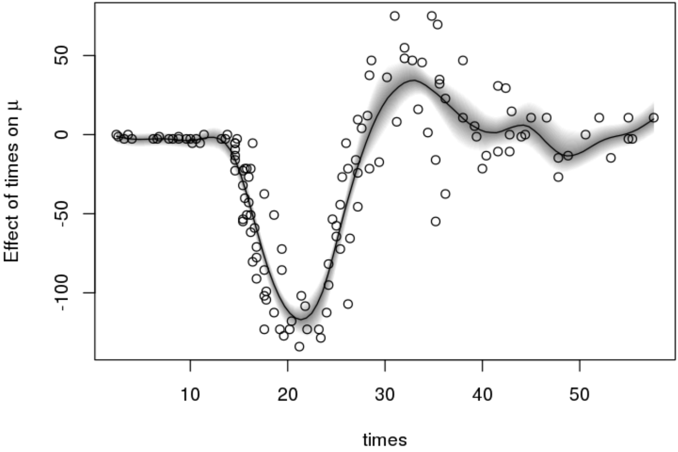
$$\texttt{accel} \sim N(\boldsymbol{\mu} = f(\texttt{times}), log(\boldsymbol{\sigma}^2) = \beta_0).$$



Distributional regression

Example: Head acceleration in a simulated motorcycle accident

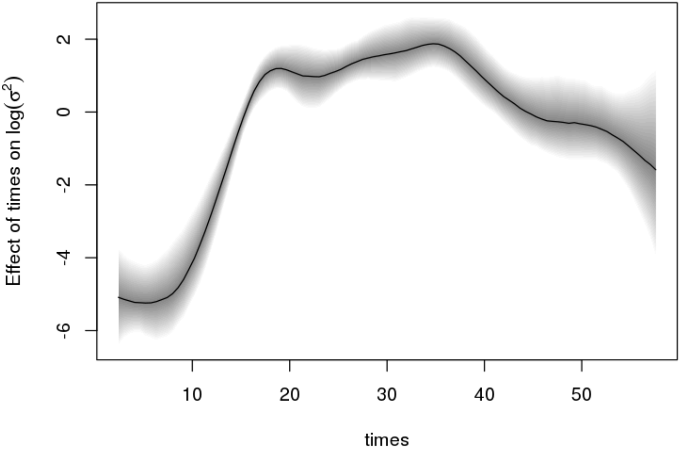
$$\texttt{accel} \sim N(\boldsymbol{\mu} = f(\texttt{times}), log(\boldsymbol{\sigma}^2) = f(\texttt{times})).$$



Distributional regression

Example: Head acceleration in a simulated motorcycle accident

$$\texttt{accel} \sim N(\boldsymbol{\mu} = f(\texttt{times}), log(\boldsymbol{\sigma}^2) = f(\texttt{times})).$$



A conceptional Lego toolbox
Families

Families specify the details of models.

Required details may differ from engine to engine, however, to fully “understand” a distribution we need the following:

- The density function.
- The distribution function.
- The quantile function.
- Link function(s).
- A random number generator.
- First and second derivatives of the log-likelihood (expectations).

So implementing a “new” distribution means creating a new family (object), including the minimum specifications required by the estimating engine(s).

A conceptional Lego toolbox
Priors

For the linear part $\mathbf{X}\boldsymbol{\beta}$, a common choice is $p(\boldsymbol{\beta}) \propto const$.

For the smooth terms, a general setup is obtained by

$$p(\boldsymbol{\gamma}_j | \tau_j^2) \propto \exp\left(-\frac{1}{2\tau_j^2}\boldsymbol{\gamma}_j^\top \mathbf{K}_j \boldsymbol{\gamma}_j\right),$$

where $\mathbf{K}_j$ is a quadratic penalty matrix that shrinks parameters towards zero or penalizes too abrupt jumps between neighboring parameters, e.g., for random effects $\mathbf{K}_j = \mathbf{I}$.

Weakly informative inverse Gamma hyperprior

$$p(\tau_j^2) = \frac{b_j^{a_j}}{\Gamma(a_j)}(\tau_j^2)^{-(a_j+1)} \exp(-b_j/\tau_j^2).$$

with $a_j = b_j = 0.001$.

A conceptional Lego toolbox
Model fitting

The main building block of regression model algorithms is the probability density function $f(\mathbf{y}|\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K)$.

Estimation typically requires to evaluate

$$\ell(\boldsymbol{\vartheta}|\mathbf{y}) = \sum_{i=1}^{n} \ln f(y_i|\theta_{i1} = h_1^{-1}(\eta_{i1}), \ldots, \theta_{iK} = h_K^{-1}(\eta_{iK})),$$

with $\boldsymbol{\vartheta} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \gamma_1, \ldots, \gamma_K)^\top$.

The log-posterior

$$\ln p(\boldsymbol{\vartheta}|\mathbf{y}) = \ell(\boldsymbol{\vartheta}|\mathbf{y}) + \sum_{k=1}^{K} \sum_{j=1}^{p_k} \left\{ \ln p(\boldsymbol{\beta}_{jk}|\tau_{jk}^2) + \ln p(\tau_{jk}^2) \right\},$$

where $\boldsymbol{\vartheta} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \gamma_1, \ldots, \gamma_K, \boldsymbol{\tau}_1^2, \ldots, \boldsymbol{\tau}_K^2)^\top$
(frequentist, penalized log-likelihood).

A conceptional Lego toolbox
Model fitting

Gradient based algorithms require the first derivative or score vector. Within the Bayesian formulation the resulting score vector is

$$s(\boldsymbol{\vartheta}) = \frac{\partial \ln p(\boldsymbol{\vartheta}|\mathbf{y})}{\partial \boldsymbol{\vartheta}} = \frac{\partial \ell(\boldsymbol{\vartheta}|\mathbf{y})}{\partial \boldsymbol{\vartheta}} + \sum_{k=1}^{K} \sum_{j=1}^{p_k} \left\{ \frac{\partial \ln p(\boldsymbol{\beta}_{jk}|\tau_{jk}^2)}{\partial \boldsymbol{\vartheta}} + \frac{\partial \ln p(\tau_{jk}^2)}{\partial \boldsymbol{\vartheta}} \right\},$$

The first order partial derivatives of the log-likelihood for $\boldsymbol{\vartheta}_k = (\boldsymbol{\beta}_k, \boldsymbol{\gamma}_k, \boldsymbol{\tau}_k^2)^\top$, can be further fragmented

$$\frac{\partial \ell(\boldsymbol{\vartheta}|\mathbf{y})}{\partial \boldsymbol{\vartheta}_k} = \frac{\partial \ell(\boldsymbol{\vartheta}|y_i)}{\partial \boldsymbol{\eta}_k} \frac{\partial \boldsymbol{\eta}_k}{\partial \boldsymbol{\vartheta}_k} = \frac{\partial \ell(\boldsymbol{\vartheta}|y_i)}{\partial \boldsymbol{\theta}_k} \frac{\partial \boldsymbol{\theta}_k}{\partial \boldsymbol{\eta}_k} \frac{\partial \boldsymbol{\eta}_k}{\partial \boldsymbol{\vartheta}_k},$$

since $\theta_{ik} = h_k^{-1}(\eta_{ik}(\boldsymbol{\vartheta}_k))$.

Applying, e.g., a Newton-Raphson algorithm additionally requires the second derivatives

$$\frac{\partial^2 \ell(\boldsymbol{\vartheta}|\mathbf{y})}{\partial \boldsymbol{\vartheta}_k \partial \boldsymbol{\vartheta}_s^\top} = \left(\frac{\partial \boldsymbol{\eta}_s}{\partial \boldsymbol{\vartheta}_s}\right)^\top \frac{\partial^2 \ell(\boldsymbol{\vartheta}|\mathbf{y})}{\partial \boldsymbol{\eta}_k \partial \boldsymbol{\eta}_s^\top} \frac{\partial \boldsymbol{\eta}_k}{\partial \boldsymbol{\vartheta}_k} \underbrace{+ \frac{\partial \ell(\boldsymbol{\vartheta}|\mathbf{y})}{\partial \boldsymbol{\eta}_k} \frac{\partial^2 \boldsymbol{\eta}_k}{\partial^2 \boldsymbol{\vartheta}_k}}_{\text{if } k=s} \quad s = 1, \dots, K.$$

PM-estimates with iteratively reweighted least squares (IWLS)

$$\mathbf{z}_k^{[t]} = \boldsymbol{\eta}_k^{[t]} + \left(\mathbf{W}_{kk}^{[t]}\right)^{-1} \mathbf{s}_k^{[t]},$$

with $\mathbf{s}_k = \partial \ell(\boldsymbol{\vartheta}|\mathbf{y}) / \partial \boldsymbol{\eta}_k$ and weights $\mathbf{W}_{kk} = -\partial^2 \ell(\boldsymbol{\vartheta}|\mathbf{y}) / \partial \boldsymbol{\eta}_k \partial \boldsymbol{\eta}_k^\top$.

Depending on the type of algorithm different weights are used, e.g., $\mathbf{W}_{kk} = E\left(-\partial^2 \ell(\boldsymbol{\vartheta}|\mathbf{y}) / \partial \boldsymbol{\eta}_k \partial \boldsymbol{\eta}_k^\top\right)$.

A conceptional Lego toolbox
Model fitting

MCMC simulation

- Metropolis-Hastings based on IWLS proposals:

$$\boldsymbol{\mu}_j = \mathbf{P}_j^{-1}\mathbf{Z}_j'\mathbf{W}(\mathbf{z} - \boldsymbol{\eta}_{-j}) \qquad \mathbf{P}_j = \mathbf{Z}_j'\mathbf{W}\mathbf{Z}_j + \frac{1}{\tau_j^2}\mathbf{K}_j,$$

  with working weights

$$\mathbf{W} = diag\left(E\left(-\frac{\partial^2\ell}{\partial\eta_i^2}\right)\right),$$

  and

$$\boldsymbol{\gamma}_j^{[t]} \sim N(\boldsymbol{\mu}_j, \mathbf{P}_j^{-1}).$$

- Other sampling schemes, e.g., slice sampling, NUTS, t-walk, ...?!

A conceptional Lego toolbox
Summary

The following quantities are repeatedly used within candidate algorithms:

- The density function $f(\mathbf{y}|\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K)$.
- The first order derivatives $\partial l(\boldsymbol{\vartheta}|\mathbf{y})/\partial\boldsymbol{\theta}_k$, $\partial\boldsymbol{\theta}_k/\partial\boldsymbol{\eta}_k$ and $\partial\boldsymbol{\eta}_k/\partial\boldsymbol{\vartheta}_k$.
- Second order derivatives $\partial^2 l(\boldsymbol{\vartheta}|\mathbf{y})/\partial\boldsymbol{\eta}_k\partial\boldsymbol{\eta}_k^\top$.
- Derivatives for priors, e.g., $ln\, p(\boldsymbol{\gamma}_{jk}|\tau_{jk}^2)$ and $ln\, p(\tau_{jk}^2)$.

A conceptional Lego toolbox

Algorithm

A simple generic algorithm for BAMLSS models:

while(eps > ε & i < maxit) {
    for(k in 1:K) {
        for(j in 1:p) {
            Compute $\boldsymbol{\eta}_{\texttt{-j}}^{[\texttt{k}]} = \boldsymbol{\eta}^{[\texttt{k}]} - \mathbf{f}_{\texttt{j}}^{[\texttt{k}]}$.
            Obtain new $(\boldsymbol{\gamma}_{\texttt{j}}^{[\texttt{k}]}, {\tau^2}_{\texttt{j}}^{[\texttt{k}]})^\top = \texttt{u}_{\texttt{j}}^{[\texttt{k}]}(\mathbf{y}, \boldsymbol{\eta}_{\texttt{-j}}^{[\texttt{k}]}, \mathbf{z}_{\texttt{j}}^{[\texttt{k}]}, \boldsymbol{\gamma}_{\texttt{j}}^{[\texttt{k}]}, {\tau^2}_{\texttt{j}}^{[\texttt{k}]}, \texttt{family}, \texttt{k})$.
            Update $\boldsymbol{\eta}^{[\texttt{k}]}$.
        }
    }
    Compute new eps
}

Functions $\texttt{u}_{\texttt{j}}^{[\texttt{k}]}(\cdot)$ could either return proposals from a MCMC sampler or updates from an optimizing algorithm.

R package bamlss

The package is available at

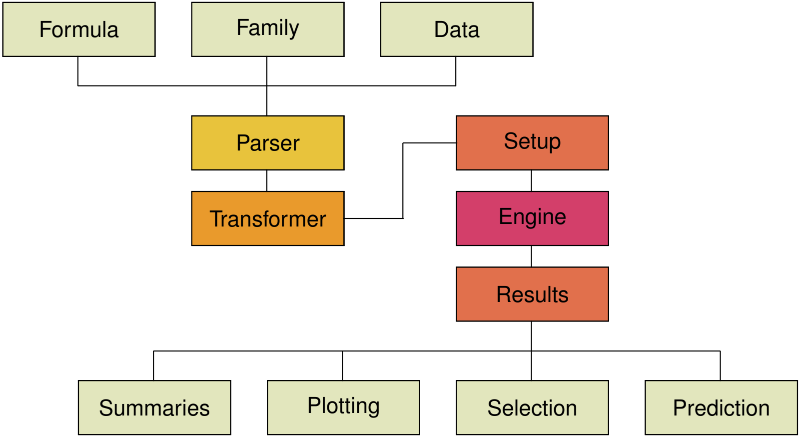
https://R-Forge.R-project.org/projects/BayesR/

In R, simply type

```
R> install.packages("bamlss",
+   repos = "http://R-Forge.R-project.org")
```

R package bamlss

Building blocks



In principle, the setup does not restrict to any specific type of engine (Bayesian or frequentist).

R package bamlss
Symbolic descriptions

Based on Wilkinson and Rogers (1973) a typical model description in R has the form

$$\texttt{response} \sim \texttt{x1 + x2}.$$

Using structured additive predictors we need generic descriptors for smooth/random terms, creating the type of term/basis we want to incorporate (model frame). The recommended R package **mgcv** (Wood 2006) has a pretty set up, e.g.

$$\texttt{response} \sim \texttt{x1 + x2 + s(z1) + s(z2, z3)}$$

$$\texttt{response} \sim \texttt{x1 + x2 + s(z1, bs = "ps")}.$$

R package bamlss

Symbolic descriptions

In the context of distributional regression we need formula extensions for multiple parameters. One convenient way to specify, e.g., the parameters of a normal model is:

```
list(
  response ~ x1 + x2 + s(z1) + s(z2),
  sigma ~ x1 + x2 + s(z1)
)
```

A four parameter example:

```
list(
  response ~ x1 + x2 + s(z1) + s(z2),
  sigma2 ~ x1 + x2 + s(z1),
  nu ~ s(z1),
  tau ~ s(z2)
)
```

R package bamlss

Symbolic descriptions

Hierarchical structures:

```
list(
  response ~ x1 + x2 + s(z1) + s(id1),
  id1 ~ x3 + s(z3) + s(id2),
  id2 ~ s(z4),
  sigma2 ~ x1 + x2 + s(z1),
  nu ~ s(z1) + s(id1),
  tau ~ s(z2)
)
```

Categorical responses:

```
list(
  response ~ x1 + x2 + s(z1) + s(z2),
  ~ x1 + x2 + s(z1) + s(z3)
)
```

R package bamlss
Input parameters

Parsing input parameters is based on **mgcv** infrastructures. In addition, the parser allows to define special user defined terms.

```
parse.input.bamlss(formula, data = NULL,
  family = gaussian, weights = NULL,
  subset = NULL, offset = NULL, na.action = na.omit,
  contrasts = NULL, knots = NULL, specials = NULL,
  reference = NULL, ...)
```

Creates the model frame, all necessary matrices, to set up a model.

```
R> f <- list(accel ~ s(times), sigma ~ s(times))
R> pm <- parse.input.bamlss(f, data = mcycle, family = gaussian)
R> names(pm)
```

```
[1] "mu"    "sigma"
```

```
R> names(pm$mu)
```

```
 [1] "formula"      "intercept"    "fake.formula" "response"
 [5] "pterms"       "sterms"       "smooth"       "sx.smooth"
 [9] "X"            "response.vec" "hlevel"
```

R package bamlss
Workflow example

JAGS

```
R> pm <- transformBUGS(pm)
R> ms <- setupJAGS(pm)
R> so <- samplerJAGS(ms)
R> mo <- resultsJAGS(pm, so)
R> summary(mo)
R> plot(mo)
```

BayesX

```
R> f <- list(
+   accel ~ sx(times),
+   sigma ~ sx(times)
+ )
R> pm <- parse.input.bayesr(f, data = mcycle, family = gaussian)
R> pm <- transformBayesX(pm)
R> ms <- setupBayesX(pm)
R> so <- samplerBayesX(ms)
R> mo <- resultsBayesX(pm, so)
R> summary(mo)
R> plot(mo)
```

R package bamlss
Available building blocks

Type	Name
Parser	`parse.input.bamlss()`
Transformer	`randomize()`, `transformBUGS()`, `transformBayesX()`, `tranformBayesG()`
Setup	`setupJAGS()`, `jags2stan()`
Engine	`samplerBayesX()`, `samplerJAGS()`, `samplerSTAN()`, `samplerBayesG()`, `engine_stacker()`
Results	`resultsBayesX()`, `resultsBUGS()`, `resultsBayesG()`

If new engines are implemented, one only needs to exchange the building block functions.

R package bamlss
Available families

Work in progress . . . (+ note that not all families are available for all implemented engines yet)

BCCG	cens	cloglog	lognormal
beta	dagum	lognormal2	quant
betazi	dirichlet	multinomial	t
betazi	gamma	mvn	truncgaussian
betazoi	gaussian	mvt	truncgaussian2
binomial	gaussian2	negbin	weibull
bivlogit	gengamma	pareto	zinb
bivprobit	invgaussian	poisson	zip

Families with ending 2 represent alternative parametrizations.

R package bamlss

Wrapper function

To ease the workflow, a wrapper function for the available engines is provided:

```
bamlss(formula, family = gaussian, data = NULL,
  knots = NULL, weights = NULL, subset = NULL,
  offset = NULL, na.action = na.fail, contrasts = NULL,
  engine = c("BayesG", "BayesX", "JAGS", "STAN"),
  cores = NULL, combine = TRUE,
  n.iter = 12000, thin = 10, burnin = 2000,
  seed = NULL, ...)
```

The function calls `xreg()` and returns an object of “`bamlss`” for which standard extractor and plotting functions are provided:

`summary()`, `plot()`, `fitted()`, `residuals()`, `predict()`, `coef()`, `DIC()`, `samples()`, `...`

Example

Dynamical Statistical Forecast of Alpine Snow Amounts

Reto Stauffer, Jakob W. Messner, Achim Zeileis and Georg J. Mayr

Affected:

- Public transport.
- Winter tourism.
- Outdoor sportsmen.
- Residents & infrastructure.

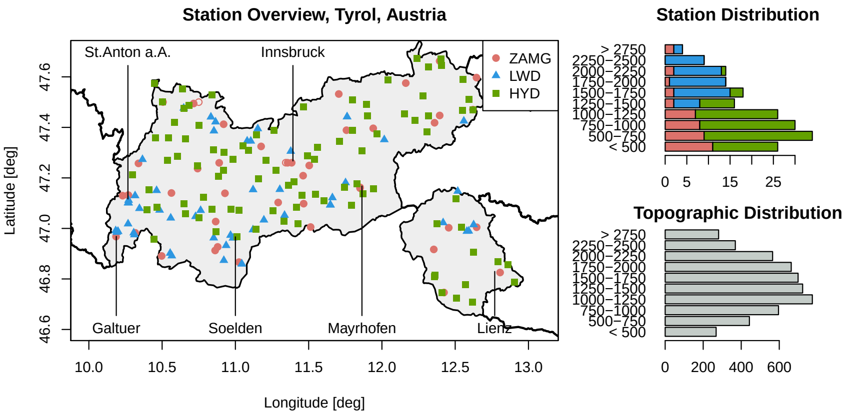
Forecasts needed for:

- Risk assessments.
- Public warning.
- Road/railroad maintenance.
- $+12h$ to few days in advance.

Challenges of rain/snow forecasting in complex terrain:

- Depends on various scales (global circulation $\rightarrow$ micro physics).
- Strongly modulated by local orography.
- Even high resolution NWP models do not resolve all important processes.
- Minor station density at high altitudes.

Example



Overview of all precipitation observation stations in Tyrol.

Left panel: Spatial distribution.

Right panel: Station and topographic distribution.

Example

Basic concept:
Use anomalies to eliminate station dependence

$$\underbrace{\text{obs} - \text{obs}_{clim}}_{\text{Observed anomalies}} = \beta_0 + \beta_1 \cdot (\underbrace{\text{ens} - \text{ens}_{clim}}_{\text{Forecast anomalies}}) + \varepsilon.$$

Corrected forecast:

$$\hat{y} = \text{obs}_{clim} + \beta_0 + \beta_1 \cdot (\text{ens} - \text{ens}_{clim}).$$

obs: Observations.

obs_{clim}: Climatology of observations.

ens: Ensemble forecasts from an NWP model.

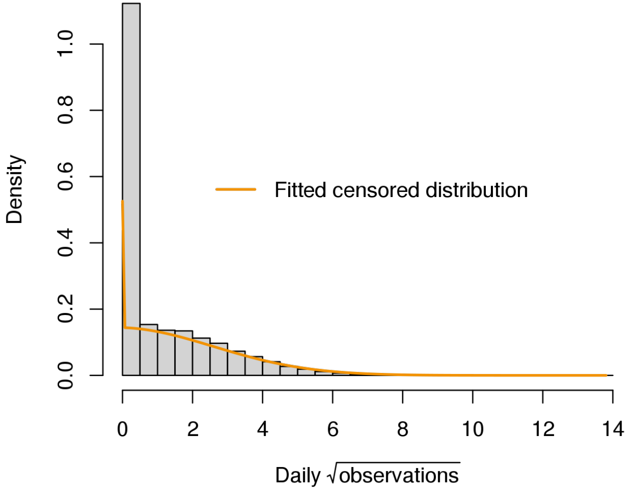
ens_{clim}: Climatology of past ensemble forecasts.

$\hat{y}$: Estimated, spatially corrected forecasts.

ε: Statistical (unexplained) error.

Example

Daily precipitation observations 1970 – 2011:



Example

Censored regression model: Latent Gaussian variable $\mathbf{y}^\star$ and observed response $\mathbf{y}$ (square root of daily precipitation observations)

$$\mathbf{y}^\star \sim N(\boldsymbol{\mu}, \boldsymbol{\sigma}^2),$$

$$\boldsymbol{\mu} = \boldsymbol{\eta}_\mu, \quad \log(\boldsymbol{\sigma}) = \boldsymbol{\eta}_\sigma,$$

$$\mathbf{y} = max(\mathbf{0}, \mathbf{y}^\star).$$

Predictors:

$$\boldsymbol{\eta} = \beta_0 + f(\texttt{yday}) + f(\texttt{alt}) + f(\texttt{lon}, \texttt{lat}).$$

Likelihood:

$$L(\boldsymbol{\vartheta}|\mathbf{y}) = \prod_{i=1}^{n} f(y_i|\boldsymbol{\vartheta}, \boldsymbol{\sigma}, \mathbf{z}_i)^{I(y_i>0)} \cdot P(y_i = 0|\mathbf{z}_i)^{I(y_i=0)}.$$

Example

```
R> library("bamlss")

R> load("data/raindata.rda")

R> f <- list(
+   sqrt(obs) ~ s(yday,bs="cc") + s(alt) + s(lon,lat,k=50),
+   sigma ~ s(yday,bs="cc") + s(alt) + s(lon,lat,k=50)
+ )

R> rainmodel <- bamlss(f, data = dat,
+   family = gF(cens, left = 0),
+   method = c("backfitting", "MCMC"),
+   update = "iwls", propose = "iwls",
+   n.iter = 12000, burnin = 2000, thin = 10)

R> summary(rainmodel)
```

Example

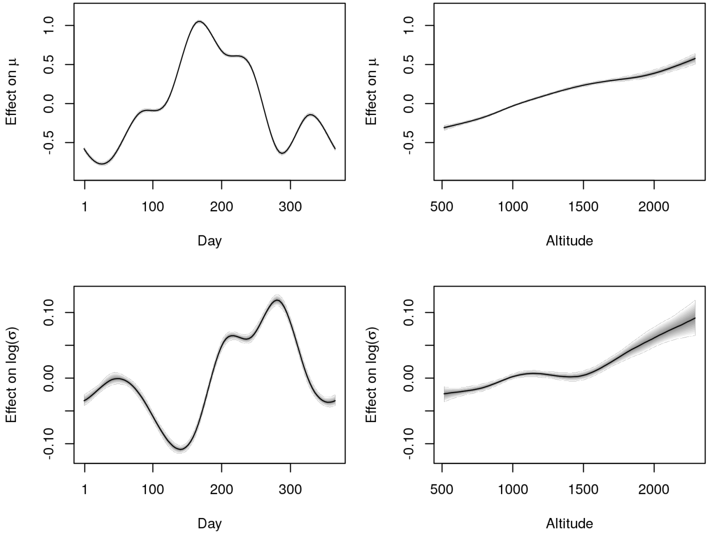
```
Call:
bamlss(formula = f, family = gF(cens, left = 0), data = dat, ...)

Family: cens
Link function: mu = identity, sigma = log
---

Results for mu:
---
Formula:
sqrt(obs) ~ s(yday, bs = "cc") + s(alt) + s(lon, lat, k = 50)

Parametric coefficients:
                Mean        Sd      2.5%       50%     97.5% alpha
(Intercept) -0.166456  0.003707 -0.173706 -0.166600 -0.159903     1

Smooth effects variances:
             Mean      Sd    2.5%     50%   97.5% alpha
s(yday)    4492.16 1794.10 2208.69 4108.18 8846.81 0.999
s(alt)      476.29  183.31  235.11  440.02  965.10 0.999
s(lon,lat)  273.88   41.31  204.93  270.01  367.54 0.997
```

Example

```
Results for sigma:
---
Formula:
~s(yday, bs = "cc") + s(alt) + s(lon, lat, k = 50)

Parametric coefficients:
                Mean        Sd     2.5%       50%     97.5% alpha
(Intercept) 0.973318 0.001221 0.970857 0.973322 0.975730 0.998

Smooth effects variances:
              Mean      Sd    2.5%      50%   97.5% alpha
s(yday)     506.40  231.64  234.84  460.71 1105.84 0.979
s(alt)       37.26   17.13   16.03   33.18   79.16 0.988
s(lon,lat)  110.74   17.77   82.03  109.02  154.17 0.939

---
DIC = 2.457e+06  N = 845321
```

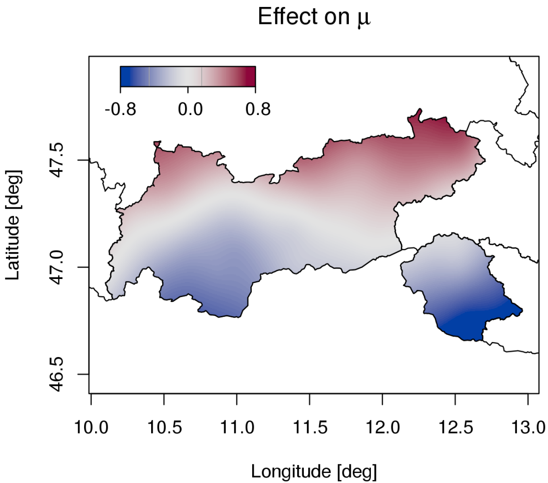
Example

```
R> plot(rainmodel, term = c("s(yday)", "s(alt)"))
```



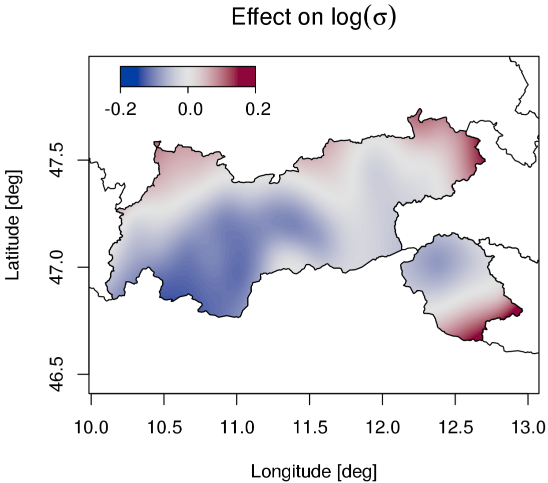
Example

```
R> plot(rainmodel, model = "mu", term = "s(lon,lat)")
```



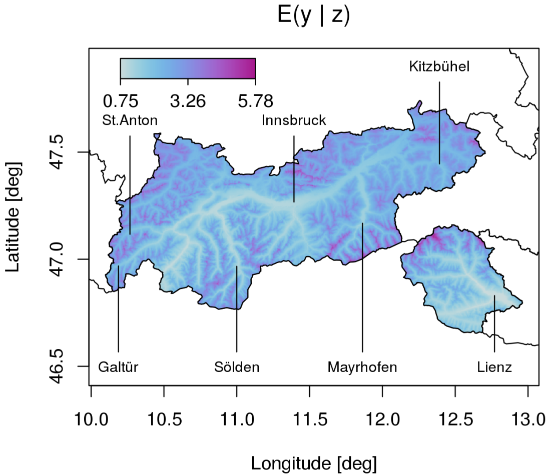
Example

```
R> plot(rainmodel, model = "sigma", term = "s(lon,lat)")
```



Example

```
R> p <- predict(rainmodel, model = "mu", newdata = nd, FUN = foo)
```



Example

Predictions for January 24:

Location	$P(y > 0 \mid z)$
Innsbruck	30.46%
St.Anton	37.40%
Galtür	37.61%
Lienz	24.86%
Sölden	30.38%
Mayrhofen	32.28%
Kitzbühel	38.29%

Thank you!!!

Klein N, Kneib T, Lang S (2013b). *Bayesian Structured Additive Distributional Regression*. Working Paper 2013-23, Working Papers in Economics and Statistics, Research Platform Empirical and Experimental Economics, Universität Innsbruck, August 2013. URL `http://econpapers.repec.org/paper/innwpaper/2013-23.htm`.

Lang S, Umlauf N, Wechselberger P, Harttgen K, Kneib T (2013): Multilevel structured additive regression. *Statistics and Computing*, 24(2), 223–238.

Umlauf N, Adler D, Kneib T, Lang S, Zeileis A (2015). *Structured additive regression models: An R interface to* ***BayesX***. Journal of Statistical Software, (to appear). URL `http://CRAN.R-project.org/package=R2BayesX`

Umlauf N, Mayr G, Messner J, Zeileis A (2012). Why does it always rain on me? A spatio-temporal analysis of precipitation in Austria. Austrian Journal of Statistics, 41(1):81-92.

Rigby RA, Stasinopoulos DM (2005). *Generalized Additive Models for Location, Scale and Shape (with Discussion)*. Applied Statistics 54, 507–554.

Wood SN (2006). *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC, Boca Raton.

Wood SN (2011). ***mgcv****: GAMs with GCV/AIC/REML Smoothness Estimation and GAMMs by PQL*. R package version 1.7-6. URL `http://CRAN.R-project.org/package=mgcv`