GGAMM

A software for empirical Bayes inference in generalized geoadditive mixed models

Thomas Kneib

Department of Statistics
Ludwig Maximilians University Munich

1 Introduction

This manual describes functions for Splus/R that allow the estimation of generalized geoadditive mixed models (GGAMM). More specific, semiparametric regression models including nonlinear effects of continuous covariates (modelled as P-splines), structured effects of spatial covariates (modelled as Markov random fields) and subject specific random effects (random intercepts and random slopes) may be estimated. Inference is based on representing generalized geoadditive mixed models as generalized linear mixed models. The main advantage of this representation is, that it allows to estimate the smoothing parameters of the P-splines and the Markov random fields simultaneously with the other model components via (restricted) maximum likelihood. From a Bayesian perspective this yields empirical Bayes or posterior mode estimates.

Section 2 reviews the theoretical background, based on Fahrmeir, Kneib & Lang (2003). In addition, some numerical issues are discussed in greater detail, based on Kneib (2003). Readers who are more interested in using the software are referred to section 3 which describes the Splus/R-implementation of `ggamm`.

2 Theoretical background

2.1 Observation model

Consider regression situations, where observations $(y_i, x_i, u_i),\ i = 1, \dots, n$, on a response y, a vector $x = (x_1, \dots, x_p)$ of continuous covariates, time scales, spatial covariates or group indicators and a vector u of further (mostly categorial) covariates are given. Generalized additive and semiparametric models (Hastie and Tibshirani, 1990) assume that, given x_i and u_i, the distribution of y_i belongs to an exponential family, with mean $\mu_i = E(y_i|x_i, u_i)$ linked to an additive semiparametric predictor η_i by

$$\mu_i = h(\eta_i),\ \eta_i = f_1(x_{i1}) + \dots + f_p(x_{ip}) + u_i'\gamma. \tag{1}$$

Here h is a known response function, and $f_1, \dots, f_p$ are unknown smooth functions of the covariates.

At first sight, the model (1) looks like a usual (semiparametric) GAM where the influence of some continuous covariates or time scales is assumed to be possibly nonlinear. The full generality of our model will be clarified in section 2.2. For the moment note

- that the covariates x_{ij} must not necessarily be continuous. For instance, the vector x_i may also contain a *spatial covariate* which gives information about the location a particular observation pertains to. In our first application on rental guides for flats, the spatial covariate contains information about the subquarter (in Munich) where the flat or apartment is located. Another example are *unordered cluster variables* that indicate the cluster index for every observation in the dataset.
- that a component of x_i may not be scalar but also 2 dimensional to model interactions between covariates.

The simple GAM-like notation in (1) is primarily used to provide a unified framework for our model and to facilitate descriptions of estimation procedures in Section 2.4.

2.2 Prior assumptions

For Bayesian inference, the unknown functions $f_1, \dots, f_p$ in (1), more exactly corresponding vectors of function evaluations, and the fixed effects parameters γ are considered as random variables and must be supplemented by appropriate prior assumptions.

We will assume independent diffuse priors $p(\gamma_j) \propto const$, for the fixed effects parameters γ. Priors for the unknown functions $f_1, \dots, f_p$ depend on the *type of the covariate* and on *prior beliefs about smoothness of* f_j. In the following we will always be able to express the vector of function evaluations $f_j = (f_j(x_{1j}), \dots, f_j(x_{nj}))'$ of an unknown function f_j as the matrix product of a design matrix X_j and a vector of unknown parameters β_j, i.e.

$$f_j = X_j \beta_j. \tag{2}$$

Then, we obtain the predictor (1) in matrix notation as

$$\eta = X_1\beta_1 + \cdots + X_p\beta_p + U\gamma, \tag{3}$$

where U corresponds to the usual design matrix for fixed effects.

A prior for a function f_j is now defined by specifying a suitable design matrix X_j and a prior distribution for the vector β_j of unknown parameters. The general form of the prior for β_j is given by

$$p(\beta_j|\tau_j^2) \propto \exp(-\frac{1}{2\tau_j^2}\beta_j' K_j \beta_j), \tag{4}$$

where K_j is a *penalty matrix* that penalizes to abrupt jumps between neighbouring parameters. In most cases K_j will be rank deficient and therefore the prior for β_j is partially improper.

The variance parameter τ_j^2 is equivalent to the inverse smoothing parameter in a frequentist approach and controls the trade off between flexibility and smoothness.

In the following we will describe specific examples of priors for some nonlinear function f_j.

2.2.1 Continuous covariates and time scales

Several alternatives have been recently proposed for specifying smoothness priors for continuous covariates. These are *random walk priors* (see Fahrmeir & Lang (2001a) and Fahrmeir & Lang (2001b)), *Bayesian P-splines* (Lang & Brezger (2003)) and *Bayesian smoothing splines* (Hastie & Tibshirani (2000)). In the following we will focus on P-splines. The approach assumes that an unknown smooth function f_j of a covariate x_j can be approximated by a polynomial spline of degree l_j defined on a set of equally spaced knots $x_j^{min} = \zeta_0 < \zeta_1 < \cdots < \zeta_{r-1} < \zeta_r = x_j^{max}$ within the domain of x_j. Such a spline can be written in terms of a linear combination of $M_j = r_j + l_j$ B-spline basis functions B_m, i.e.

$$f_j(x) = \sum_{m=1}^{M_j} \beta_{jm} B_m(x).$$

Here $\beta_j = (\beta_{j1}, \dots, \beta_{jM_j})'$ corresponds to the vector of unknown regression coefficients. The $n \times M_j$ design matrix X_j consists of the basis functions evaluated at the observations x_{ij}, i.e. $X_j(i, m) = B_m(x_{ij})$. The crucial point is the choice of the number of knots. For a small number of knots, the resulting spline may be not flexible enough to capture the variability of the data. For a large number of knots, estimated curves tend to overfit the data and, as a result, to rough functions are obtained. As a remedy Eilers & Marx (1996) suggest a moderately large number of equally spaced knots (usually between 20 and 40) to ensure enough flexibility, and to define a roughness penalty based on first or second order differences of adjacent B-Spline coefficients to guarantee sufficient smoothness of the fitted curves. This leads to penalized likelihood estimation with penalty terms

$$P(\lambda_j) = \lambda_j \sum_{m=k_j+1}^{M_j} (\Delta^{k_j}\beta_{jm})^2, \quad k_j = 1, 2, \tag{5}$$

where Δ^{k_j} is the difference operator. First order differences penalize abrupt jumps $\beta_{jm} - \beta_{j,m-1}$ between successive parameters and second order differences penalize deviations from the linear trend $2\beta_{j,m-1} - \beta_{j,m-2}$. In a Bayesian approach we use the stochastic analogue of difference penalties, i.e. first or second order random walks, as a prior for the regression coefficients. First and second order random walks are defined by

$$\beta_{jm} = \beta_{j,m-1} + u_{jm} \quad \text{or} \quad \beta_{jm} = 2\beta_{j,m-1} - \beta_{j,m-2} + u_{jm} \tag{6}$$

with Gaussian errors $u_{jm} \sim N(0, \tau_j^2)$ and diffuse priors $p(\beta_{j1}) \propto \textit{const}$, or $p(\beta_{j1})$ and $p(\beta_{j2}) \propto \textit{const}$, for initial values, respectively. The joint distribution of the regression parameters β_j is easily computed as a product of conditional densities defined by (6) and can be brought into the general form (4). The penalty matrix is of the form $K_j = D_j' D_j$ where D_j is a first or second order difference matrix. More details about Bayesian P-splines can be found in Lang & Brezger (2003).

2.2.2 Spatial covariates

Suppose now that a covariate x_j represents the location or site in connected geographical regions. For simplicity we assume $x_j \in \{1, \ldots, M_j\}$, i.e. the regions are labelled consecutively by the numbers $1, \ldots, M_j$. A common way to deal with spatial covariates is to assume that neighbouring sites are more alike than two arbitrary sites. Thus for a valid prior definition a set of neighbours for each site m must be defined. For geographical data one usually assumes that two sites m and m' are neighbours if they share a common boundary.

The simplest (but most often used) spatial smoothness prior for the function evaluations $f_j(m) = \beta_{jm}$ is

$$\beta_{jm}|\beta_{jm'}, \, m \neq m', \tau_j^2 \sim N\left(\frac{1}{N_m}\sum_{m' \in \partial_m} \beta_{jm'}, \frac{\tau_j^2}{N_m}\right), \tag{7}$$

where N_m is the number of adjacent sites and $m' \in \partial_m$ denotes that site m' is a neighbour of site m. Thus the (conditional) mean of β_{jm} is an unweighted average of function evaluations of neighbouring sites. The prior is a direct generalization of a first order random walk to two dimensions and is called a Markov random field. More general priors based on weighted averages can be found e.g. in Besag, York & Mollié (1991). The $n \times M_j$ design matrix X_j is now a 0/1 incidence matrix. Its value in the i-th row and the m-th column is 1 if the i-th observation is located in site or region m, and zero otherwise. The $M_j \times M_j$ penalty matrix K_j has the form of an adjacency matrix.

2.2.3 Unordered group indicators and unstructured spatial effects

In many situations we observe the problem of heterogeneity among clusters of observations caused by unobserved covariates. Neglecting unobserved heterogeneity may lead to considerably biased estimates for the remaining effects. Suppose $x_j \in \{1, \dots, M_j\}$ is now a cluster variable indicating the cluster a particular observation i belongs to. A common approach to overcome the difficulties of unobserved heterogeneity is to introduce additional Gaussian i.i.d. effects

$$f_j(m) = \beta_{jm} \tag{8}$$

with

$$\beta_{jm} \sim N(0, \tau_j^2), \quad m = 1, \dots, M_j. \tag{9}$$

The design matrix X_j is again a $n \times M_j$ 0/1 incidence matrix and the penalty matrix is the identity matrix, i.e. $K_j = I$. From a classical perspective, (9) defines i.i.d. *random effects*. However, from a Bayesian point of view all unknown parameters are assumed to be random and hence the notation "random effects" in this context is misleading. We think of (9) more as an approach for modelling an unsmooth function.

I.i.d. Gaussian effects (9) may also be used for a more sophisticated modelling of spatial effects. If x_j is a spatial covariate it may be useful to split up the effect f_j into a spatially correlated (smooth) effect $f_{j,str}$ and a spatially uncorrelated (unsmooth) effect $f_{j,unstr}$, i.e.

$$f_j = f_{j,str} + f_{j,unstr}.$$

A rationale is that a spatial effect is usually a surrogate of many unobserved influential factors, some of them may obey a strong spatial structure and others may be present only locally. By estimating a structured and an unstructured effect we aim at distinguishing between the two kinds of influential factors. For the smooth spatial effect we may again assume a Markov random field prior and for the uncorrelated effect we may assume the prior (9).

Often random effects (9) are not only used to model so-called *random intercepts* but also for *random slopes*. Random slopes are also included in the presented setting, if we assume, that x_j is composed of two covariates $x_j^{(1)}$ and $x_j^{(2)}$, i.e. $x_j = (x_j^{(1)}, x_j^{(2)})'$, where $x_j^{(2)}$ is an unordered group indicator and $x_j^{(1)}$ is a continuous variable. Now we define $f_j(x_j)$ through

$$f_j(x_j) = f_j^*(x_j^{(2)})x_j^{(1)}, \tag{10}$$

where $f_j^*(x_j^{(2)})$ is defined as in (8). So we obtain for $x_j^{(2)} = m$:

$$f_j(x_j) = f_j^*(m)x_j^{(1)} = x_j^{(1)}\beta_{jm}.$$

The design matrix X_j is now given by

$$\text{diag}(x_{1j}^{(1)}, \dots, x_{nj}^{(1)})X_j^*$$

where X_j^* is the incidence matrix for the unordered group indicator $x_j^{(2)}$.

Note, that in fact (10) is a varying coefficient model (VCM) (compare Hastie & Tibshirani (1993)) with an unordered group indicator as effect modifier. More general VCMs might also be incorporated into our model but since the current implementation only supports random slopes we do not emphasize this here.

2.3 Mixed Model representation

In this section, we highlight the close relationship between penalized regression and generalized linear mixed models (GLMM), see also Lin & Zhang (1999) and Green (1987) in the context of smoothing splines. In fact, model (1) with the structured additive predictor (3) can always be expressed as a GLMM. This provides a nice and elegant way for simultaneous estimation of the functions f_j, $j = 1, \ldots, p$ and the variance (or inverse smoothing) parameters τ_j^2 in an empirical Bayes approach (see the next section). To rewrite the model as a GLMM we proceed as follows:

The vectors of regression coefficients β_j, $j = 1, \ldots, p$, are decomposed into an *unpenalized* and a *penalized part*. Suppose that the j-th coefficient vector has dimension $M_j \times 1$ and the corresponding penalty matrix K_j has rank rk_j. Then we define the decomposition

$$\beta_j = X_j^{unp}\beta_j^{unp} + X_j^{pen}\beta_j^{pen}, \tag{11}$$

where the columns of the $M_j \times (M_j - rk_j)$ matrix X_j^{unp} contain a basis of the nullspace of K_j. The $M_j \times rk_j$ matrix X_j^{pen} is given by $X_j^{pen} = L_j(L_j'L_j)^{-1}$ where the $M_j \times rk_j$ matrix L_j is determined by the decomposition of the penalty matrix K_j into $K_j = L_jL_j'$. A requirement for the decomposition is that $L_j'X_j^{unp} = X_j^{unp}L_j' = 0$ holds. Hence the parameter vector β_j^{unp} represents the part of β_j which is not penalized by K_j whereas the vector β_j^{pen} represents the deviations of the parameters β_j from the nullspace of K_j.

In general, the decomposition $K_j = L_jL_j'$ of K_j can be obtained from the spectral decomposition $K_j = \Gamma_j\Omega_j\Gamma_j'$. The $(rk_j \times rk_j)$ diagonal matrix Ω_j contains the positive eigenvalues ω_{jm}, $m = 1, \ldots, T_j$, of K_j in descending order, i.e. $\Omega_j = diag(\omega_{j1}, \ldots, \omega_{j,rk_j})$. Γ_j is a $(M_j \times rk_j)$ orthogonal matrix of the corresponding eigenvectors. From the spectral decomposition we can choose $L_j = \Gamma_j\Omega_j^{\frac{1}{2}}$.

In some cases a more favorable decomposition can be found. For instance, for P-splines defined in Section 2.2.1 a more favorable choice for L_j is given by $L_j = D_j'$ where D_j is the first or second order difference matrix. Of course, for (the "random effects") prior (9) of section 2.2.3 a decomposition of $K_j = I$ is not necessary. Also, the unpenalized part vanishes completely.

The matrix X_j^{unp} is the identity vector for P-splines with first order random walk penalty and Markov random fields. For P-splines with second order random walk penalty X_j^{unp} is a two column matrix whose first column is again the identity vector and the second column is composed of the (equidistant) knots of the spline.

From the decomposition (11) we get

$$\frac{1}{\tau_j^2}\beta_j'K_j\beta_j = \frac{1}{\tau_j^2}(\beta_j^{pen})'\beta_j^{pen}.$$

From the general prior (4) for β_j it follows that

$$p(\beta_{jm}^{unp}) \propto const, \qquad m = 1, \ldots, M_j - rk_j$$

and

$$\beta_j^{pen}|\tau_j^2 \sim N(0, \tau_j^2 I). \tag{12}$$

Finally, by defining the matrices $\tilde{U}_j = X_jX_j^{unp}$ and $\tilde{X}_j = X_jX_j^{pen}$, we can rewrite the predictor (3) as

$$\begin{aligned}
\eta &= \sum_{j=1}^{p} X_j\beta_j + U\gamma \\
&= \sum_{j=1}^{p}(X_jX_j^{unp}\beta_j^{unp} + X_jX_j^{pen}\beta_j^{pen}) + U\gamma \\
&= \sum_{j=1}^{p}(\tilde{U}_j\beta_j^{unp} + \tilde{X}_j\beta_j^{pen}) + U\gamma \\
&= \tilde{U}\beta^{unp} + \tilde{X}\beta^{pen}.
\end{aligned}$$

The design matrix $\tilde{X}$ and the vector β^{pen} are composed of the matrices $\tilde{X}_j$ and the vectors β_j^{pen}, respectively. More specifically, we obtain

$$\tilde{X} = (\tilde{X}_1 \; \tilde{X}_2 \; \cdots \; \tilde{X}_p)$$

and the stacked vector

$$\beta^{pen} = ((\beta_1^{pen})', \ldots, (\beta_p^{pen})')'.$$

Similarly the matrix $\tilde{U}$ and the vector β^{unp} are given by

$$\tilde{U} = (\tilde{U}_1 \; \tilde{U}_2 \; \cdots \; \tilde{U}_p \, U)$$

and

$$\beta^{unp} = ((\beta_1^{unp})', \ldots, (\beta_p^{unp})', \gamma')'.$$

After all, we obtain a GLMM with fixed effects β^{unp} and random effects β^{pen} with

$$\beta^{pen} \sim N(0, \Lambda)$$

where $\Lambda = diag(\tau_1^2, \ldots, \tau_1^2, \ldots, \tau_p^2, \ldots, \tau_p^2)$. Hence, we can utilize GLMM methodology for simultaneous estimation of smooth functions and the variance parameters τ_j^2, see the next section.

The mixed model representation also enables us to examine the identification problem inherent to nonparametric regression from a different angle. Except for i.i.d. Gaussian effects (9), the design matrices $\tilde{U}_j$ for the unpenalized parts contain the identity vector. Provided that there are at least one nonlinear effect and that γ contains an intercept, the matrix $\tilde{U}$ has not full column rank. Hence, all identity vectors in $\tilde{U}$ (except for the intercept) must be eliminated to guarantee identifiability.

2.4 Inference

Bayesian inference is based on the posterior of the model. In terms of the GLMM representation of the model we obtain

$$p(\beta^{unp}, \beta^{pen}|y) \quad \propto \quad L(y, \beta^{unp}, \beta^{pen}) \prod_{j=1}^{p} \left(p(\beta_j^{pen}|\tau_j^2) \right) \tag{13}$$

where $p(\beta^{pen}|\tau_j^2)$ is defined in (12) and $L(\cdot)$ denotes the likelihood which is the product of individual likelihood contributions.

Based on the GLMM representation outlined in Section 2.3, regression and variance parameters can be estimated using iteratively weighted least squares (IWLS) and (approximate) restricted maximum likelihood (REML) developed for GLMM's. Estimation is carried out iteratively in largely two steps. Suppose $\tilde{\beta}^{unp}$, $\tilde{\beta}^{pen}$ and $\tilde{\tau}_j^2$, $j = 1, \ldots, p$ are the current estimates for the unknown parameters. Then, we obtain updated estimates $\hat{\beta}^{unp}$ and $\hat{\beta}^{pen}$ given the current variance parameters as the solutions of the linear equation system

$$\begin{pmatrix} \tilde{U}'W\tilde{U} & \tilde{U}'W\tilde{X} \\ \tilde{X}'W\tilde{U} & \tilde{X}'W\tilde{X} + \Lambda^{-1} \end{pmatrix} \begin{pmatrix} \beta^{unp} \\ \beta^{pen} \end{pmatrix} = \begin{pmatrix} \tilde{U}'W\tilde{y} \\ \tilde{X}'W\tilde{y} \end{pmatrix}. \tag{14}$$

The $(n \times 1)$ vector $\tilde{y}$ and the $n \times n$ diagonal matrix $W = diag(w_1, \ldots, w_n)$ are the usual working observations and weights in generalized linear models, see Fahrmeir & Tutz (2001), Chapter 2.2.1. In the second step, the (approximate) restricted log likelihood

$$\begin{aligned} l^*(\tau_1^2, \ldots, \tau_p^2) \quad &= \quad \tfrac{1}{2}\log(|\Sigma|) - \tfrac{1}{2}\log(|\tilde{U}'\Sigma^{-1}\tilde{U}|) \\ &\quad -\tfrac{1}{2}(\tilde{y} - \tilde{U}\hat{\beta}^{unp})'\Sigma^{-1}(\tilde{y} - \tilde{U}\hat{\beta}^{unp}) \end{aligned} \tag{15}$$

is maximized with respect to the variance parameters $\tau^2 = (\tau_1^2, \dots, \tau_p^2)$. Here, Σ is an approximation to the marginal covariance matrix of $\tilde{y}$ which is given by

$$\Sigma = W^{-1} + \tilde{X}\Lambda\tilde{X}'.$$

The restricted likelihood is maximized by Fisher scoring as described in full detail in Harville (1977) or Verbeke & Molenberghs (2000). Updated estimates $\hat{\tau}^2$ are obtained by

$$\hat{\tau}^2 = \tilde{\tau}^2 + F^*(\tilde{\tau}^2)^{-1} s^*(\tilde{\tau}^2), \tag{16}$$

where F^* is the expected Fisher information and s^* is the score vector, see e.g. Lin & Zhang (1999) for formulas. The two steps are iterated until convergence.

Note, that convergence problems may occur, if one of the parameters τ_j^2 is small. Then the maximum of the restricted likelihood may be on the boundary of the parameter space so that Fisher scoring is no appropriate way to find the REML-estimates $\hat{\tau}^2$. Therefore in the current implementation the estimation of small variances τ_j^2 is stopped, if the criterion

$$c(\tau_j^2) = \frac{||\tilde{X}_j \hat{\beta}_j^{pen}||}{||\hat{\eta}||} \tag{17}$$

is smaller than a user-specified value. This usually corresponds to small values τ_j^2 but defines "small" in a data driven way. Modifying Fisher scoring in this way guaranteed convergence in most of the analyzed models.

From a Bayesian perspective, the (approximate) covariance matrix of the regression coefficients $\hat{\beta}^{unp}$ and $\hat{\beta}^{pen}$ may be derived from (14) as

$$\text{Cov}\begin{pmatrix}\hat{\beta}^{unp}\\ \hat{\beta}^{pen}\end{pmatrix} = H^{-1} \tag{18}$$

with

$$H = \begin{pmatrix}\tilde{U}'W\tilde{U} & \tilde{U}'W\tilde{X}\\ \tilde{X}'W\tilde{U} & \tilde{X}'W\tilde{X} + \Lambda^{-1}\end{pmatrix}.$$

In a frequentist setting, this covariance matrix is given by

$$\text{Cov}\begin{pmatrix}\hat{\beta}^{unp}\\ \hat{\beta}^{pen}\end{pmatrix} = H^{-1}H_1H^{-1} \tag{19}$$

with

$$H_1 = \begin{pmatrix}\tilde{U}'W\tilde{U} & \tilde{U}'W\tilde{X}\\ \tilde{X}'W\tilde{U} & \tilde{X}'W\tilde{X}\end{pmatrix}.$$

Both expressions allow us to compute confidence intervals for the estimates $\hat{f}_j$, since $\hat{f}_j = \tilde{U}_j\hat{\beta}_j^{unp} + \tilde{X}_j\hat{\beta}_j^{pen}$ and therefore

$$\text{Cov}(\hat{f}_j) = (\tilde{U}_j, \tilde{X}_j)\,\text{Cov}\begin{pmatrix}\hat{\beta}_j^{unp}\\ \hat{\beta}_j^{pen}\end{pmatrix}(\tilde{U}_j, \tilde{X}_j)'. \tag{20}$$

The covariance matrix $\text{Cov}\begin{pmatrix}\hat{\beta}_j^{unp}\\ \hat{\beta}_j^{pen}\end{pmatrix}$ can be obtained from the corresponding blocks in (18) and (19) respectively. Pointwise confidence intervals for f_j may now be computed using the square roots of the diagonal elements from (20) as estimates of the corresponding standard deviations.

3 Splus/R-functions

3.1 Installation

The current implementation of `ggamm` consists of the files

- ggamm.s
- helpfunctions.s

for Splus and the files

- ggamm.r
- helpfunctions.r

for R, respectively. While the files `ggamm.s` and `ggamm.r` contain the two versions of the function `ggamm` for the different software packages, the files `helpfunctions.s` and `helpfunctions.r` provide several additional functions that are needed during the estimation and that allow the user to visualize the estimation results. To define `ggamm` in Splus or R, one first has to install these additional functions through executing the commands

```
> source("c:\\ggamm\\functions\\helpfunctions.s")
```

or

```
> source("c:\\ggamm\\functions\\helpfunctions.r")
```

respectively. Of course the path `c:\\ggamm\\functions` has to be changed according to the storage location of the corresponding files.

To define `ggamm` itself, execute the commands

```
> source("c:\\ggamm\\functions\\ggamm.s")
```

in Splus and

```
> source("c:\\ggamm\\functions\\ggamm.r")
```

in R. Both implementations differ neither by their call (that is: their arguments) nor their return value.

A typical call to `ggamm` has the form

```
> test<-ggamm(dep=...,fix=...,smooth=...,reg=...,regions=...,random=...,
  id=...,...)
```

Now the estimation results are stored in the object `test` and may for example be used to visualize the function estimates (compare section 3.4). The different arguments of `ggamm` are described in more detail in the following section, more information on the return value is given in section 3.3.

Note, that the R implementation of `ggamm` is usually much faster then the Splus implementation and should therefore be preferred.

3.2 Arguments

The only required argument is the dependent variable `dep`. If no other arguments are supplied, a constant fixed effect is estimated. Note that if only fixed effects are specified, `ggamm` simply calls the Splus/R-function `glm` and converts the results to the typical form of a `ggamm`-object.

`dep`	Vector containing the dependent variable y.	
`family`	Exponential family to which the distribution of the response belongs.	
	`"normal"`	Normally distributed response.
	`"binomial"`	Binomial distributed response. If one assumes $y_i \sim B(n_i, p_i)$, y_i/n_i must be specified as dependent variable. The numbers of trials n_i are supplied in the variable `weight`.
	`"poisson"`	Poisson distributed response.
	`"gamma"`	Gamma distributed response. Note, that the behavior of `ggamm` for gamma distributed response has not yet been tested properly.
	Default: `"normal"`	
`link`	Link function. For normally or Poisson distributed responses the natural link function is chosen, for gamma distributed response the logarithm is used as link function. So this option is only meaningful for binomial distributed response.	
	`"logit"`	Cdf of the logistic distribution as response function.
	`"probit"`	Cdf of the standard normal distribution as response function.
	Default: `"logit"`	
`dispers`	Whether to estimate the dispersion parameter or not. For normally or gamma distributed response this option must be `T`, for binomial and poisson distributed response `dispers=T` allows the estimation of quasi-likelihood models. Note however, that the estimation of such models has not yet been properly tested. *Default:* `F`	
`fix`	Matrix containing the realizations of the covariates that are assumed to have linear influence.	
`smooth`	Matrix containing the realizations of the covariates that are to be modelled as P-splines.	
`nknot`	Vector, containing the number of inner knots r_j for every variable in `smooth`. *Default:* 20	
`ord`	Vector, containing the order k_j of the difference penalty for every variable in `smooth`. Possible values are $k_j = 1$ and $k_j = 2$. *Default:* 2	
`deg`	Vector, containing the degree l_j of the B-spline basis for every variable in `smooth`. *Default:* 3	
`plotf`	Whether to plot the estimated functions automatically or not. *Default:* `F`	
`include.lin`	Whether to include the linear part of the function estimate in the plot or not. Only meaningful for covariates with `ord`=2. *Default:* `F`	
`reg`	Vector containing the realizations of the spatial covariate.	
`pmatrix`	Adjacency matrix K_j of the spatial effect. Please contact the author, if you if you have problems creating such a matrix.	

`regions`	Vector containing the possible (distinct) values of the spatial covariate in the order corresponding to the adjacency matrix.
`map`	Map object which allows to visualize the estimated spatial effect. Compare section 3.4 on how to define such an object.
`plotmap`	Whether to plot the estimated spatial effect automatically or not. *Default:* `F`
`random`	Matrix containing the realizations of the covariates that should be modelled as random effects. For example a random intercept is specified through an n-dimensional vector of ones.
`id`	Vector containing the identification variable of the clusters.
`weight`	Vector containing a weighting variable.
`offset`	Vector containing an offset.
`sig`	Level of the confidence intervals that are to be computed. *Default:* 0.95
`eps`	Termination condition. *Default:* 0.00001
`noprint`	Whether to print information on the estimation process or not. *Default:* `F`
`nowarnings`	Whether to print warnings or not. *Default:* `F`
`startValue1`	Starting value for the inverse smoothing parameters of the P-splines and the Markov random field. *Default:* 0.1
`startValue2`	Starting value for the variances of the random effects. *Default:* 0.5
`maxit`	Maximum number of iterations. *Default:* 400
`ranktest`	Whether to test for rank-deficiency of the fisher-information-matrix for β^{pen} and β^{unp} in each iteration or not. `ranktest=T` decreases the execution time. *Default:* `F`
`outfile`	If outfile is specified, the estimation results are written to the given directory. For example the estimation results for the fixed effects are written to `c:\\temp\\results_fixedEffects.raw` if `outfile="c:\\temp\\results"` is specified.
`log.like`	Whether to compute the (restricted) log-likelihood or not. Only implemented for normally distributed response. *Default:* `F`
`lowerlim`	Termination criterion that specifies in which case the estimation of a small variance shall be stopped. Compare section 2. *Default:* 0.001
`method`	How to estimate the variance components. `"ML"` Maximum Likelihood; `"REML"` Restricted Maximum Likelihood. *Default:* `"REML"`

3.3 Return value

The return value of `ggamm` consists of a list of Splus/R-objects that contain the estimation results for the different model components. These objects will now be briefly summarized. Note, that some of the objects like `log.like` or `spatialEffects` are only available if they are part of the analyzed model or if they are explicitly requested in the call of `ggamm`.

`fixedEffects`	Matrix containing the estimation results of the fixed effects. Confidence intervals, standard deviations and p-values are given in addition. If the P-splines have been modelled using differences of order $k_j = 2$, coefficients representing the linear parts of the P-splines are given too.
`iterations`	Number of iterations.
`log.like`	Log-likelihood or restricted log-likelihood of the estimated model.
`predict`	Matrix containing the estimation results of the linear predictor and the expectation of the different observations.
`randomEffects`	Matrix containing the estimation results of the random effects. If more than one covariate is modelled as a random effect, such a matrix is generated for each of these covariates. The matrix also contains confidence intervals, standard deviations and p-values.
`smoothedEffects`	Matrix containing the estimation results of the nonparametric effects. Confidence intervals, standard deviations and p-values are given in addition. If more than one covariate is modelled nonparametrically, such a matrix is generated for each of these covariates.
`spatialEffects`	Matrix containing the estimation results of the spatial effect. The matrix also contains confidence intervals, standard deviations and p-values.
`variances`	Matrix containing the estimation results of the variance parameters τ^2 and the dispersion parameter ϕ.

3.4 Visualizing estimated effects

Two functions allow the user to visualize the estimation results of the nonparametric and the spatial effects. If the estimation results are stored in the object `test`, the nonparametric effects may be visualized as follows:

```
> plotf(test)
```

In addition to the estimation results pointwise frequentist and Bayesian confidence intervals are drawn automatically. If the argument `include.lin=T` is supplied, the linear parts of the function estimates are included in the plot, too.

Estimates of spatial effects may be visualized through typing

```
> plotmap(test,m)
```

if object `m` contains the corresponding map.

To define a map-object, we first need information about its boundaries, which we assume to be stored in a so-called boundary file. Such a file has the following structure (The next paragraphs are taken from section 5 in Brezger, Kneib & Lang (2002). Compare this section for a more detailled description of map-objects and boundary-files):

For each region of the map the boundary file must contain the identifying name of the region, the polygons that form the boundary of the region, and the number of lines the polygon consists of. The first line always contains the region code surrounded by quotation marks and the number of lines the polygon of the region consists of. The code and the number of lines must be separated by a comma. The subsequent lines contain the coordinates of the straight lines that form the boundary of the region. The straight lines are represented by the coordinates of their end points. Coordinates must be separated by a comma.

To give an example we print a (small) part of the boundary file of (former) West Germany:

```
    ⋮

"6634",31
2319.26831,4344.48828
2375.45435,4399.50391
2390.67139,4446.32520
2470.26807,4405.35645
2576.78735,4379.60449
2607.22144,4337.46533
2627.12061,4356.19385
2662.23682,4355.02344
2691.50024,4311.71338
2726.61646,4310.54248
2716.08154,4256.69775
2710.22900,4227.43408
2680.96533,4234.45752
2583.81055,4165.39551
2568.59351,4096.33398
2520.60132,4042.48901
2535.81836,3941.82251
2490.16724,3920.75269
2451.53955,3903.19458
2437.49292,3924.26440
2369.60156,3933.62866
2359.06665,3951.18677
2285.32275,3969.91553
2258.40015,4061.21753
2197.53223,4049.51221
2162.41602,4086.96948
2204.55542,4091.65161
2192.85010,4125.59717
2284.15210,4220.41113
2339.16748,4292.98438
2319.26831,4344.48828

    ⋮
```

The map corresponding to the section of the boundary file above can be found in Figure 1. Note that the first and the last point must be identical (see the example above) to obtain a closed polygon.

In some cases it might happen that a region is separated into subregions that are not connected. As an illustrative example compare Figure 2 showing a region of Germany that is separated into 8 subregions. In this case the boundary file must contain the polygons of all subregions. The first row for each of these subregions must contain the region code and the number of lines the polygon of the respective subregion consists of. Note that it is not necessary that the polygons of the subregions are stored in subsequent order in the boundary file.

Another special case that might occur is illustrated in Figure 3. Here a region is completely surrounded by another region. In this case an additional line must be added to the boundary description of the *surrounded* region. The additional line must be placed after the first line and must contain the region code of the *surrounding* region. The syntax is:

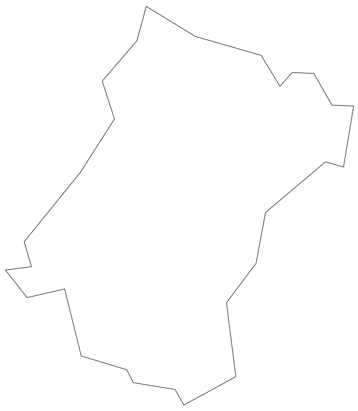

Figure 1: Corresponding graph of the section of the boundary file

is.in,"*region code*"

The following lines show a section of the boundary file of West Germany, where region "9361" is totally surrounded by region "9371":

```
⋮
"9361",7
is.in,"9371"
4155.84668,2409.58496
4161.69922,2449.38330
4201.49756,2461.08862
4224.90820,2478.64673
4250.66016,2418.94922
4193.30371,2387.34448
4155.84668,2409.58496

⋮
```

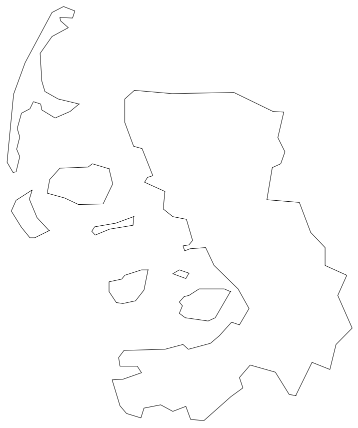

Figure 2: Example for a region that is divided into subregions

The information provided in a boundary-file is read into a map-object using the function `readbndfile`. The function has two required arguments: the filename of the boundary file to read in and the name of the map-object in Splus/R. Both arguments have to be supplied as

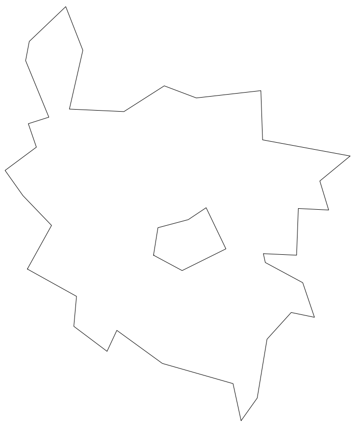

Figure 3: Example for a region that is totally surrounded by another region

string-variables. A typical call to `readbndfile` would be

```
> readbndfile("c:\\ggamm\\boundary.bnd","mymap")
```

As a result, the boundary information contained in the file `boundary.bnd` will be read into a map-object called `mymap`. Compare also section 3.5 which gives an example on the usage of `ggamm` and the visualization-functions.

Note, that the function `plotmap` mainly consists of a call to the function `drawmap`, which has been written by Andreas Brezger for the visualization of geographical information. For this function there exist many additional options which allow the user to customize the plot and which are described in section 7.4.2.3 in Brezger et al. (2002). So `plotmap` is only a possibility to get a first impression of the estimation results with less typing effort.

All the functions `plotf`, `plotmap`, `drawmap` and `readbndfile` are included in the files `helpfunctions.s` or `helpfunctions.r` and are defined through executing the commands

```
> source("c:\\ggamm\\functions\\helpfunctions.s")
```

or

```
> source("c:\\ggamm\\functions\\helpfunctions.r")
```

respectively.

3.5 Examples

Now the usage of `ggamm` shall be demonstrated through the analysis of two datasets. These datasets as well as all further files that are needed for the analysis are part of the file `examples.zip`, which is available from `http://www.stat.uni-muenchen.de/~kneib/ggamm.html`.

Example 1: Credit Scoring.

In example 1 we analyze a dataset containing information on 1000 consumers' credits from a South German bank with a generalized additive model. The aim is to predict the probability that a client with certain covariates or risk factors will not pay back his credit. Therefore the response variable is the binary variable creditability (named y in the dataset) with $y = 0$ for creditworthy clients and $y = 1$ for not creditworthy clients. As covariates we have two continuous variables and 5 categorial variables (in effect coding) which are described in Table 1.

Variable	**Description**
duration	duration of the credit in months
amount	amount of credit in 1000DM
$account1$	running account of the client with categories "no running account" ($account1$=1),
$account2$	"good running account" ($account2 = 1$) and "medium running account"
	($account1 = account2 = -1$)
payment	payment of previous credits with categories "good" (= 1) and "bad" (= -1)
intuse	intended use with categories "private" (= 1) and "professional" (= -1)
marstat	marital status with categories "married" (= 1) and "living alone" (= -1)

Table 1: Covariates in the credit scoring dataset.

To analyze the data, we first define `ggamm` in the current session and store the data in a dataframe-object.

```
> source("c:\\ggamm\\functions\\helpfunctions.r")
> source("c:\\ggamm\\functions\\ggamm.r")
> creditdata<-read.table("c:\\ggamm\\examples\\credit.raw",header=T)
```

To make the names of the variables available directly, we attach the dataframe:

```
> attach(creditdata)
```

Now we combine the covariates that are to be modelled as P-splines and the categorial covariates in two different matrices:

```
> smoothcovs<-cbind(duration,amount)
> catcovs<-cbind(account1,account2,payment,intuse,marstat)
```

Finally we call `ggamm` and store the estimation results in the object `credit`:

```
> credit<-ggamm(dep=y,fix=catcovs,smooth=smoothcovs,family="binomial")
```

As a result of this call, the following information is given on the screen:

```
 iteration: 1

 relative changes in the regression coefficients: Inf
 relative changes in the variance parameters:     0.7858781
⋮

 iteration: 11

 relative changes in the regression coefficients: 2e-006
 relative changes in the variance parameters:     5.8e-006

 variance of smooth1: 0.0045697637
 variance of smooth2: 0.0156629498
 beta0: -0.25677
 beta1: -1.09241
 beta2: 0.86104
 beta3: -0.49619
 beta4: -0.21911
 beta5: -0.25864
```

In each iteration of the estimation process the relative changes in the parameters are computed and compared with the (possibly user-specified) value of `eps`. When the estimation process has converged some estimation results are given. Namely, these are the estimators of the variance components and of the fixed effects. More information on these estimators is given in the two objects `credit$variances` and `credit$fixedEffects`:

```
> credit$variances
        coefficient stopcrit  stopped
smooth1 0.004569764 0.1488299       0
smooth2 0.015662950 0.2605143       0


> credit$fixedEffects
      coefficient  ci0p025  ci0p975     std  pvalue
beta0    -0.25677 -0.56373  0.05020 0.15662 0.05056
beta1    -1.09241 -1.33428 -0.85055 0.12340 0.00000
beta2     0.86104  0.64730  1.07477 0.10905 0.00000
beta3    -0.49619 -0.74639 -0.24599 0.12765 0.00005
beta4    -0.21911 -0.37729 -0.06092 0.08071 0.00332
beta5    -0.25864 -0.41562 -0.10166 0.08009 0.00062
fbeta1    0.03389       NA       NA      NA      NA
fbeta2    0.07633       NA       NA      NA      NA
```

In addition to the estimators, `credit$variances` also contains the values of criterion (17) and a dummy-variable that indicates, whether the estimation process has been stopped for this variance (=1) or not (=0). In this case the estimation of both variances could be performed without modification.

The matrix `credit$fixedEffects` contains the coefficients of the fixed effects, confidence intervals for these coefficients, their standard deviations and their p-values. Note, that we omitted their Bayesian counterparts in this presentation, which are also included in `credit$fixedEffects`. The last to lines of `credit$fixedEffects` contain the parameters of the linear parts of the smooth effects that were estimated for *duration* and *amount*.

These effects may be visualized by typing

```
> plotf(credit)
```

which results in the graphs that are shown in Figure 4.

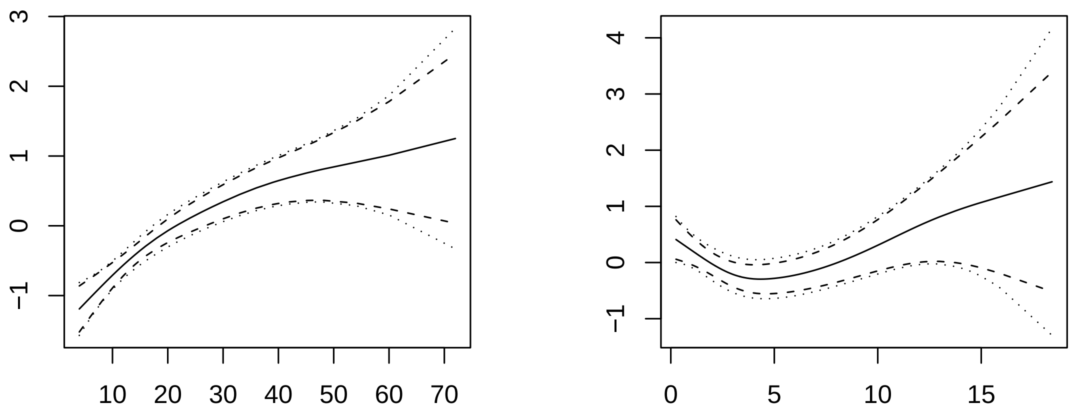

Figure 4: Estimation results for the smooth effects of duration and amount.

Example 2: Childhood undernutrition in Zambia.

In our second example we analyze the influence of certain covariates on undernutrition of children in Zambia. Here undernutrition is measured by a standardized score that represents insufficient height for age of the children. Besides some categorial covariates (in effect coding) we have two continuous covariates and information on the district in which the child lives (compare Table 2 for a description of the covariates). Therefore we can estimate a generalized geoadditive mixed model to assess the influence of the covariates on undernutrition. Note that we will split up the spatial effect into a structured and an unstructured component as described in section 2.2.2.

Variable	**Description**
hazstd	Standardized measure of stunting (height for age)
bmi	body mass index of the mother
agc	age of the child
district	district where the child lives
rcw	mother‘s employment status with categories "working" (= 1) and "not working" (= −1)
*edu*1	mother‘s educational status with categories "complete primary but incomplete
*edu*2	secondary" ($edu1 = 1$), "complete secondary or higher" ($edu2 = 1$) and "no education or incomplete primary" ($edu1 = edu2 = -1$)
tpr	locality of the domicile with categories "urban" (= 1) and "rural" (= −1)
sex	gender of the child with categories "male" (= 1) and "female" (= −1)

Table 2: Covariates in the undernutrition dataset.

Again we first define `ggamm` in the current session and then read the data into a dataframe:

```
> source("c:\\ggamm\\functions\\helpfunctions.r")
> source("c:\\ggamm\\functions\\ggamm.r")
> zambiadata<-read.table("c:\\ggamm\\examples\\zambia.raw",header=T)
> attach(zambiadata)
```

Then we combine continuous and categorial covariates in distinct matrices and define a vector of ones which serves as a covariate for the unstructured spatial effect.

```
> smoothcovs<-cbind(bmi,agc)
> catcovs<-cbind(rcw,edu1,edu2,tpr,sex)
> rancov<-rep(1,4847)
```

For the estimation of the structured spatial effect we need the adjacency matrix of the districts and a vector containing the identification numbers of the distinct districts (in the order corresponding to the adjacency matrix). Both are provided in the file `examples.zip` and must be read into the current session:

```
> adjmat<-read.table("c:\\ggamm\\examples\\Kzambia.raw")
> distinctregions<-scan("c:\\ggamm\\examples\\zambiaregions.raw")
```

Note that, since the dataset of this example contains 4847 observations and the model is rather complex, the computation may take some time (depending on the computer) and needs about 300Mb (in Splus) or 250Mb (in R) RAM. It might also be necessary (in Splus) to change `object.size` to an appropriate value:

```
> options(object.size=8000000)
```

Now we can estimate the model:

```
> zambia<-ggamm(dep=hazstd,fix=catcovs,smooth=smoothcovs,reg=district,
  regions=distinctregions,pmatrix=adjmat,random=rancov,id=district,dispers=T)
```

We obtain the following estimation results:

```
 WARNING: estimation of the variance of smooth1 was stopped after iteration 10
          because its penalized part was small relative to the linear predictor.

 phi: 0.80215
 variance of smooth1: 8.66e-008
 variance of smooth2: 0.0032290746
 variance of the spatial effect: 0.0334812023
 variance of random1: 0.0064807152
 beta0: 0.05839
 beta1: 0.00732
 beta2: -0.05988
 beta3: 0.23459
 beta4: 0.08872
 beta5: -0.05855
```

The warning is produced because the estimation of one of the variances is stopped due to the problem described in section 2.4. As it was noted in this section, this usually corresponds to small variances, which is indeed the case in our analysis for the variance of the effect of *bmi*. This is also reflected in Figure 5 which shows the estimation results of the nonparametric effects and which was produced through executing

```
> plotf(zambia)
```

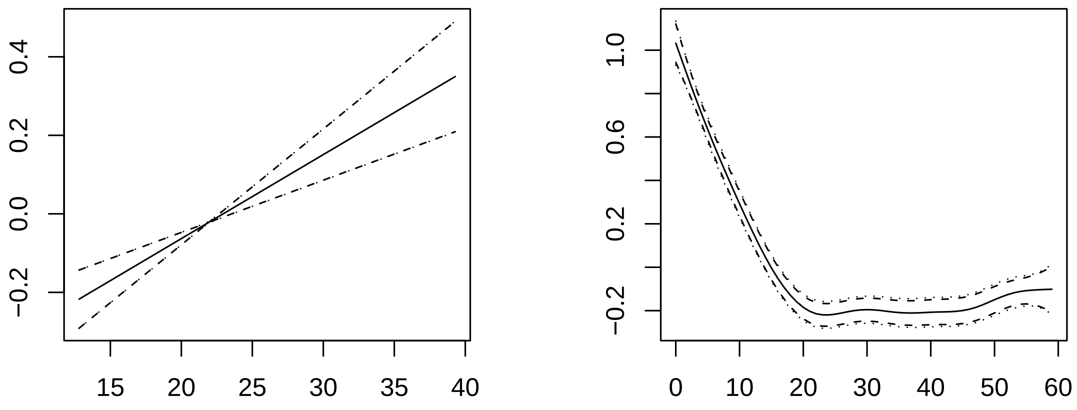

Figure 5: Estimation results for the smooth effects of bmi and agc.

Note, that the shape of the confidence intervals for the effect of *bmi* is also caused by the extremely small value for the corresponding variance.

To visualize the spatial effects, we have to create a map-object containing the geographical information about the districts in zambia. This information is provided in the boundary-file `mapzambia.bnd` and is loaded into the object `m` by a call to `readbndfile`. Then the estimation results of the structured spatial effect can be visualized using the function `plotmap`.

```
> readbndfile("c:\\ggamm\\examples\\mapzambia.bnd","m")
> plotmap(zambia,m)
```

Since the unstructered spatial effect is simply an uncorrelated random effect for each district, this effect can not be visualized using `plotmap`. Instead we have to use the original function `drawmap`. Therefore we first attach the estimation results and call `drawmap` with the following arguments:

```
> attach(zambia)
> drawmap(plotvar=randomEffects1[,2],regionvar=randomEffects1[,1],map=m,color=T)
```

Both spatial effects are shown in Figure 6.

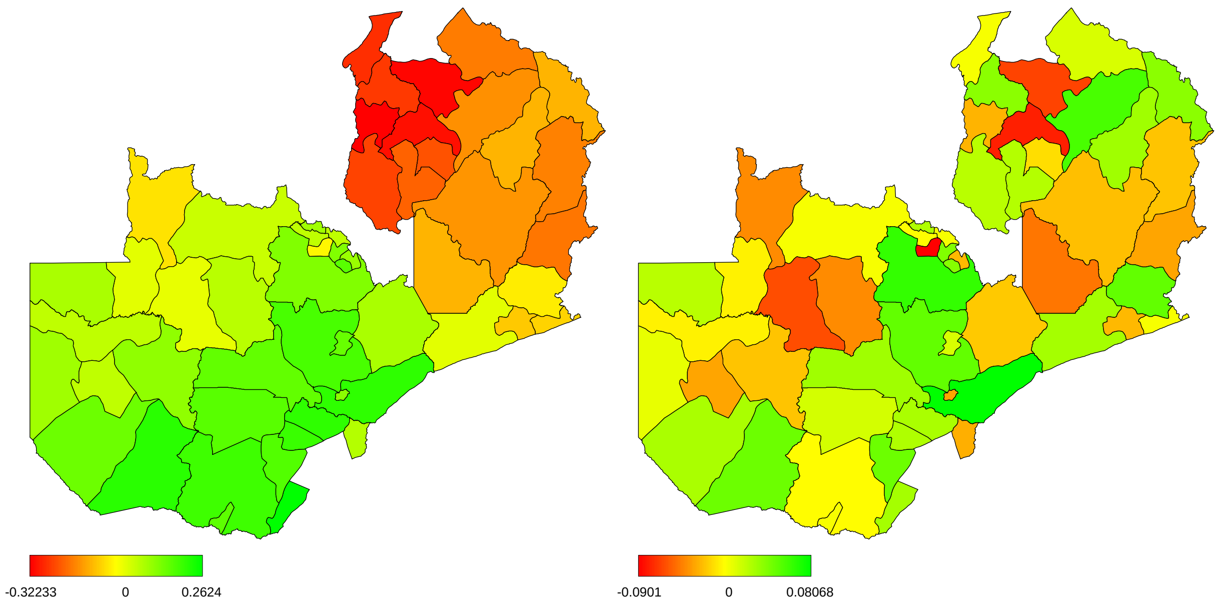

Figure 6: Structured and unstructered spatial effect.

In addition to the datasets and the files used in example 2 `examples.zip` contains adjacency matrices, boundary files and indicators of the distinct districts for some other regions. These are the districts of Germany, the districts of the western part of Germany and districts within munich. The file `readme.txt` which is also included in `examples.zip` gives more information on the different files in `examples.zip`.

References

Besag, J., York, J. & Mollié, A. (1991). Bayesian image restoration with two applications in spatial statistics (with discussion), *Annals of the Institute of Statistical Mathematics* **43**: 1–59.

Brezger, A., Kneib, T. & Lang, S. (2002). BayesX - Software for bayesian inference based on markov chain monte carlo simulation techniques. Erhältlich unter www.stat.uni-muenchen.de/∼lang/bayesx/bayesx.html.

Eilers, P. H. C. & Marx, B. D. (1996). Flexible smoothing with B-splines and penalties, *Statistical Science* **11**: 89–121.

Fahrmeir, L., Kneib, T. & Lang, S. (2003). Penalized additive regression for space-time data: A Bayesian perspective, *Statistica Sinica (under revision)* . Available from www.stat.uni-muenchen.de/∼kneib.

Fahrmeir, L. & Lang, S. (2001a). Bayesian inference for generalized additive mixed models based on markov random field priors, *Journal of the Royal Statistical Society Series C* **50**: 201–220.

Fahrmeir, L. & Lang, S. (2001b). Bayesian semiparametric regression analysis of multicategorial time-space data, *Annals of the Institute of Statistical Mathematics* **53**: 11–30.

Fahrmeir, L., Lang, S., Wolff, J. & Bender, S. (2001). Semiparametric bayesian time-space analysis of unemployment duration, *Discussion Paper 211, Sonderforschungsbereich 386, Universität München* .

Fahrmeir, L. & Tutz, G. (2001). *Multivariate Statistical Modelling based on Generalized Linear Models*, Springer, New York.

Green, P. J. (1987). Penalized likelihood for general semi-parametric regression models, *International Statistical Review* **55**: 245–259.

Harville, D. A. (1977). Maximum likelihood approaches to variance component estimation and to related problems, *Journal of the American Statistical Association* **72**: 320–338.

Hastie, T. J. & Tibshirani, R. J. (1993). Varying-coefficient models (with discussion), *Journal of the Royal Statistical Society Series B* **55**: 757–796.

Hastie, T. & Tibshirani, R. J. (2000). Bayesian backfitting, *Statistical Science* **15**: 193–223.

Kneib, T. (2003). Bayes-Inferenz in generalisierten geoadditiven gemischten Modellen. Diploma thesis, University of Munich. Available from www.stat.uni-muenchen.de/∼kneib.

Lang, S. & Brezger, A. (2003). Bayesian P-Splines, *Journal of Computational and Graphical Statistics (to appear)* . Available from www.stat.uni-muenchen.de/∼lang.

Lin, X. & Zhang, D. (1999). Inference in generalized additive mixed models by using smoothing splines, *Journal of the Royal Statistical Society Series B* **61**: 381–400.

Verbeke, G. & Molenberghs, G. (2000). *Linear Mixed Models for Longitudinal Data*, Springer, New York.