Lab 4.4: Spatio-temporal modeling of GAMs with mgcv

We often seek more flexible models that can accommodate nonlinear structure in the mean function. One successful approach to this problem has been through the use of *generalized additive models* or GAMs. In general, these models consider a transformation of the mean response to be an additive form, in which the additive components are smooth functions (e.g., splines) of the covariates. For example, we might consider a data model

$$z(\mathbf{s};t)|Y(\mathbf{s};t),\gamma \;\sim\; ind.\; EF(Y(\mathbf{s};t),\gamma), \quad \mathbf{s}\in D_s, t\in D_t, \tag{1}$$

and a mean response that is modeled additively as

$$g(Y(\mathbf{s};t)) = \mathbf{x}(\mathbf{s};t)'\boldsymbol{\beta} + \sum_{i=1}^{n_f} f_i(\mathbf{x}(\mathbf{s};t)) + \nu(\mathbf{s};t), \tag{2}$$

where $g(\cdot)$ is a specified monotonic link function, $\mathbf{x}(\mathbf{s};t)$ is a p-dimensional vector of covariates for spatial location $\mathbf{s}$ and time t, $f_i(\cdot)$ are functions of the covariates, the spatial locations, or the time index, and $\nu(\mathbf{s};t)$ is a spatio-temporal random effect. Typically, the functions $f_i(\cdot)$, $i = 1,\ldots,n_f$, are modeled in terms of a truncated basis expansion, for example, $f_i(x_1(\mathbf{s};t)) = \sum_{k=1}^{r_1}\phi_k(x_1(\mathbf{s};t))\alpha_{ik}$, where ϕ_k, $k = 1,\ldots,r_1$, are the basis functions. Further, just as one can add random effects to generalized linear models to get generalized linear mixed models, we can also add random effects to the mean response in (2) to get generalized additive mixed models (GAMMs).

GAMs can also be considered as the process model in a hierarchical statistical model. The primary difference between the GAM approach and a random-effects model using spatio-temporal basis functions is the additional flexibility that one can get by considering nonlinear functions of spatio-temporal covariates. Note also that GAMs typically assume that the basis functions are smooth functions, whereas there is no such requirement for spatio-temporal basis function models. That being said, there may also be advantages in considering these models directly in the GAM framework due to efficient computation and estimation approaches that have been established in the GAM literature.

Generalized additive models (GAMs) and generalized additive mixed models (GAMMs) can be implemented quickly and efficiently with the package **mgcv** and the functions `gam` and `gamm`, respectively. For a comprehensive treatment of GAMs and GAMMs and their implementation through **mgcv**, see Wood (2017).

In this Lab we aim to predict the expected counts at arbitrary spatio-temporal locations, from the vector of observed counts $\mathbf{Z}$. The data we use are from the North American Breeding Bird Survey.[1] In particular, we consider yearly counts of the Carolina wren (*Thryothorus ludovicianus*) at BBS routes for the period 1994–2014. We require the package **mgcv** as well as **dplyr, tidyr, ggplot2** and **STRbook**.

[1] https://www.pwrc.usgs.gov/bbs/rawdata/

```
library("dplyr")
library("ggplot2")
library("mgcv")
library("STRbook")
library("tidyr")
data("MOcarolinawren_long", package = "STRbook")
```

GAMs and GAMMs rely on constructing smooth functions of the covariates, and in a spatio-temporal context these will inevitably include space and time. In this Lab we consider the following simple GAM:

$$g(Y(\mathbf{s};t)) = \beta + f(\mathbf{s};t) + \nu(\mathbf{s};t), \tag{3}$$

where $g(\cdot)$ is a link function, β is an intercept, the function $f(\mathbf{s};t)$ is a random smooth function of space and time, and $\nu(\mathbf{s};t)$ is a spatio-temporal white-noise error process.

In **mgcv**, the random function $f(\mathbf{s};t)$ is generally decomposed using a separable *spline* basis. Now, there are several basis functions that can be used to reconstruct $f(\mathbf{s};t)$, some of which are knot-based (e.g., B-splines). For the purpose of this Lab, it is sufficient to know that splines, of whatever order, are decomposed into a set of basis functions. Thus, $f(\mathbf{s};t)$ is decomposed as $\sum_{i=1}^{r_1}\phi_{1i}(\mathbf{s};t)\alpha_{1i}$, where the $\{\alpha_{1i}\}$ are unknown random effects that need to be predicted, and the $\{\phi_{1i}\}$ are given below.

There are a number of basis functions that can be chosen. Those derived from thin-plate regression splines are convenient, as they are easily amenable to multiple covariates (e.g., functions of $(\mathbf{s};t) \equiv (s_1, s_2; t)$). Thin-plate splines are isotropic and invariant to rotation but not invariant to covariate scaling. Hence, the use of thin-plate splines for fitting a curve over space *and* time is not recommended, since units in time are different from those in space.

To combine interacting covariates with different units, such as space and time, **mgcv** implements a tensor-product structure, whereby the basis functions smoothing the individual covariates are combined productwise. That is,

$$f(\mathbf{s};t) = \sum_{i=1}^{r_1}\sum_{j=1}^{r_2}\phi_{1i}(\mathbf{s})\phi_{2j}(t)\alpha_{ij} \equiv \boldsymbol{\phi}(\mathbf{s};t)'\boldsymbol{\alpha}.$$

The function `te` forms the product from the marginals; for example, in our case this can achieved by using `te(lon,lat,t)`. Other arguments can be passed to `te` for added functionality; for example, the basis-function class is specifed through `bs`, the number of basis functions through `k`, and the dimension of each spline through `d`. In this case we employ a thin-plate spline basis over longitude and latitude (`"tp"`) and a cubic regression spline over time (`"cr"`). A GAM formula for (3) is implemented as follows

```
f <- cnt ~ te(lon, lat, t,          # inputs over which to smooth
              bs = c("tp", "cr"),   # types of bases
              k = c(50, 10),        # knot count in each dimension
              d = c(2, 1))          # (s,t) basis dimension
```

We chose $r_1 = 50$ basis functions for the spatial component and $r_2 = 10$ for the temporal component. These values were chosen after some trial and error. The number of knots could have been set using cross-validation. In general, the estimated degrees of freedom should be considerably lower than the total number of knots; if this is not the case, probably the number of knots should be increased.

In Lab 3.4 we saw that the Carolina wren counts are over-dispersed. To account for this, we use the negative-binomial distribution to model the response (a quasi-Poisson model would also be suitable). The **gam** function is called in the code below, where we specify the negative-binomial family and a log link (the function $g(\cdot)$ in (3)):

```
cnts <- gam(f, family = nb(link = "log"),
            data = MOcarolinawren_long)
```

The returned object is a `gam` object, which extends `glm` and `lm` objects (i.e., functions that can be applied to `glm` and `lm` objects, such as **residuals**, can also be applied to `gam` objects). The negative-binomial distribution handles over-dispersion in the data through a size parameter r, such that, for a fixed mean, the negative-binomial distribution approaches the Poisson distribution as $r \to \infty$. In this case the estimated value for r (named `Theta` in **mgcv**) is

```
cnts$family$getTheta(trans = 1)

## [1] 5.2
```

which is not large and suggestive of over-dispersion. Several graphical diagnostics relating to the fit can be explored using the **gam.check** function.

To predict the field at unobserved locations using the hierarchical model, we first construct a space-time grid upon which to predict.

```
MOlon <- MOcarolinawren_long$lon
MOlat <- MOcarolinawren_long$lat

## Construct space-time grid
grid_locs <- expand.grid(lon = seq(min(MOlon) - 0.2,
                                   max(MOlon) + 0.2,
                                   length.out = 80),
                         lat = seq(min(MOlat) - 0.2,
                                   max(MOlat) + 0.2,
```

```
                                    length.out = 80),
                          t = 1:max(MOcarolinawren_long$t))
```

Then we call the function **predict** which, when se.fit = TRUE, returns a list containing the predictions and their associated prediction standard errors.

```
X <- predict(cnts, grid_locs, se.fit = TRUE)
```

Specifically, the predictions and prediction standard errors are available in X$fit and X$se.fit, respectively. These can be plotted using **ggplot2** as follows.

```
## Put data to plot into data frame
grid_locs$pred <- X$fit
grid_locs$se <- X$se.fit

## Plot predictions and overlay observations
g1 <- ggplot() +
    geom_raster(data = grid_locs,
                aes(lon, lat, fill = pmin(pmax(pred, -1), 5))) +
    facet_wrap(~t, nrow = 3, ncol = 7) +
    geom_point(data = filter(MOcarolinawren_long, !is.na(cnt)),
               aes(lon, lat),
               colour = "black", size = 3) +
    geom_point(data=filter(MOcarolinawren_long, !is.na(cnt)),
               aes(lon, lat, colour = log(cnt)),
               size = 2) +
    fill_scale(limits = c(-1, 5),
               name = expression(log(Y[t]))) +
    col_scale(name = "log(cnt)", limits=c(-1, 5)) +
    theme_bw()

## Plot prediction standard errors
g2 <- ggplot() +
    geom_raster(data = grid_locs,
                aes(lon, lat, fill = pmin(se, 2.5))) +
    facet_wrap(~t, nrow = 3, ncol = 7) +
    fill_scale(palette = "BrBG",
               limits = c(0, 2.5),
               name = expression(s.e.)) +
    theme_bw()
```

The plots are shown in Figures 1 and 2, respectively. One may also use the **plot.gam** function on cnts to quickly generate plots of the tensor products.

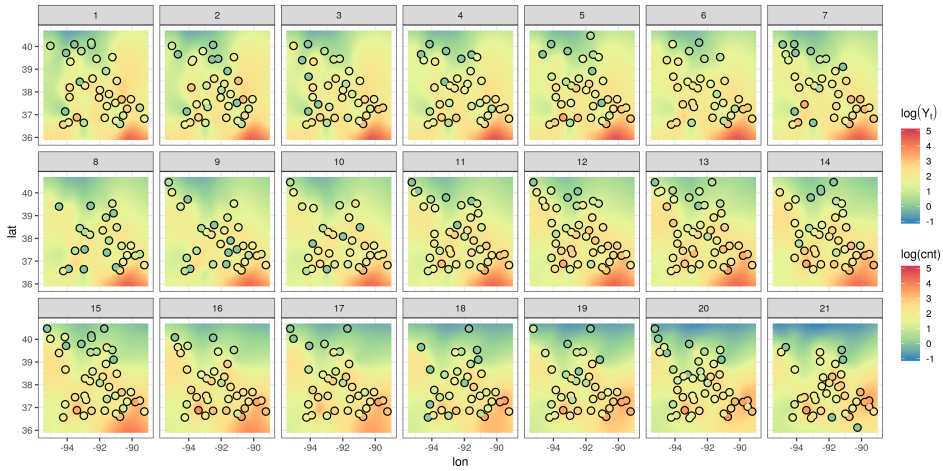

Figure 1: Posterior mean of $\log(Y(\cdot))$ on a grid for $t = 1$ (the year 1994) to $t = 21$ (the year 2014), based on a negative-binomial data model using the package **mgcv**. The log of the observed count is shown in circles using the same color scale.

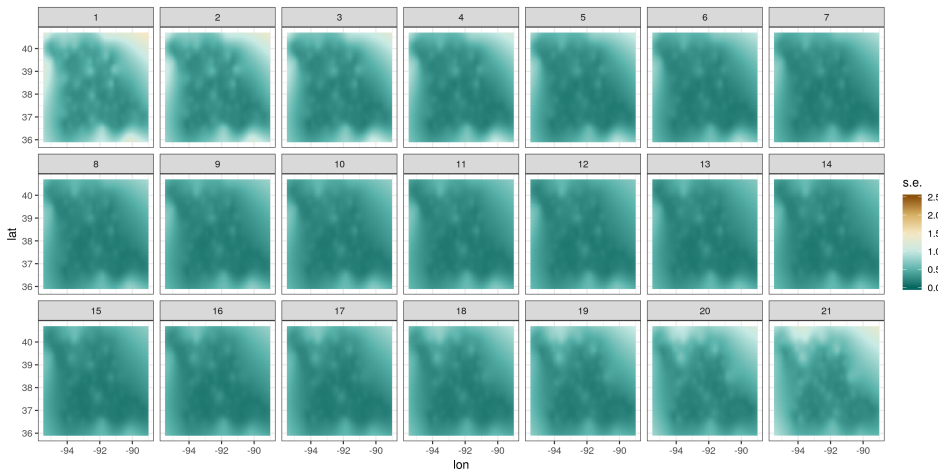

Figure 2: Posterior standard deviation (i.e., prediction standard error) of $\log(Y(\cdot))$ on a grid for $t = 1$ (the year 1994) to $t = 21$ (the year 2014), based on a negative-binomial data model using the package **mgcv**.

Bibliography

Wood, S. N. (2017), *Generalized Additive Models: An Introduction with R*, 2nd ed., Boca Raton, FL: Chapman & Hall/CRC.