mgcViz: computer lab exercises

Please make sure that the following packages are installed:

- `devtools` (`install.packages("devtools")`);
- `mgcViz` (`library(devtools); install_github("mfasiolo/mgcViz")`);
- `mgcFam` (`install_github("mfasiolo/mgcFam")`);
- `qgam` and `e1071` (`install.packages(c("qgam", "e1071"))`).

We have two main exercises, both based on UK electricity load data. In particular:

1. interactive GAMLSS model building: in this exercise you will start by modelling load using a simple Gaussian GAM and, by looking at several visual diagnostics, you will improve your model to take into account the fact that the shape of the conditional load distribution varies a lot depending on temperature, time of year, etc.
2. BAM modelling and looking for interactions: here you will start with a model including only the marginal effect of each covariate. Then, you will use several visual checks to determine the complexity of each marginal effect basis, and to detect interactions between the covariates.

The second exercise might be more fun. Questions marked by * should be considered extra or optional. Notice that `mgcViz` is not well documented at the moment, but the vignette and the slides should be enough to get you through both exercises.

1 Interactive GAMLSS model building

Here we consider a UK electricity demand dataset, taken from the national grid. The dataset covers the period January 2011 to June 2016 and it contains the following variables:

- `NetDemand` net electricity demand between 11:30am and 12am.
- `wM` instantaneous temperature, averaged over several English cities.
- `wM_s95` exponential smooth of `wM`, that is `wM_s95[i] = a*wM[i] + (1-a)*wM_s95[i]` with `a=0.95`.
- `Posan` periodic index in `[0, 1]` indicating the position along the year.
- `Dow` factor variable indicating the day of the week.
- `Trend` progressive counter, useful for defining the long term trend.
- `NetDemand.48` lagged version of `NetDemand`, that is `NetDemand.48[i] = NetDemand[i-2]`.
- `Holy` binary variable indicating holidays.
- `Year` and `Date` should obvious, and partially redundant.

Questions:

1. Load `mgcViz`, `qgam` and the data (`data("UKload")`). Then create a model formula (e.g. `y~s(x)`) containing: smooth effects for `wM`, `wM_s95` and `Trend` with 20, 20 and 4 knots and cubic regression splines bases (`bs='cr'`), a cyclic effect (`bs='cc'`) for `Posan` with 30 knots; and parametric fixed effects for `Dow`, `NetDemand.48` and `Holy`. Fit a Gaussian GAM using this model formula and convert it to a `gamViz` object called (say) `fit0`, by calling `getViz` with argument `nsim = 50`.

2. Use the `check1D` function together with the `l_gridCheck1D` layer to check whether the conditional mean of the residuals of `fit0` varies along `wM`, `wM_s95` or `Posan`. In the call to `l_gridCheck1D` you can set `stand = "sc"` to standardize the residuals means, thus making the residuals patterns more visible. Look at the plot for `Posan`, does the residuals mean in January ($Posan \approx 0$) differ from that in December ($Posan \approx 1$)? What does this suggest?

3. Change the model formula, by using a cubic regression spline basis also for `Posan`, and refit the model. Convert it to a `gamViz` object (called `fit1`) and compare `AIC(fit0)` with `AIC(fit1)`. As before, check the residuals along `Posan` using `check1D` and `l_gridCheck1D`. Is the pattern gone? Now use `check(fit1)` and look at the p-values. Recall that a low p-value means that an effect has not enough degrees of freedom. Also, plot all the smooth effects using `plot(fit1)`, how does the effect of `Posan` look like? Given this plot and the result of `check` can you think of a better spline basis for `Posan`?

4. Change the model formula, by using an adaptive spline basis (`bs = 'ad'`) for `Posan`, and refit the model. Convert it as before, call it `fit2` and compare its AIC with that of `fit1`. Use `check` to see if the p-value for `Posan` is still significant, and plot only the smooth for `Posan` by extracting the corresponding smooth using the `sm` function and then adding the `l_points`, `l_fitLine` and `l_ciLine` layers. Now that we are satisfied with our mean model, we start looking at the conditional variance. Use `check1D` together with the `l_densCheck` layer to compare the empirical and theoretical (Gaussian) density of the residuals along `wM`, `wM_s95` an `Posan`. Do you see any evidence of model mis-specification? Now use `l_gridCheck1D` with `gridFun = sd` to check for heteroscedasticity along the same variables. Does the variance look constant along `wM`, `wM_s95` and `Posan`?

5. Now we will fit a GAMLSS model using the `gaulss` family (see `?gaulss`). For the location use the same model formula we have used in the Gaussian GAM, while for the scale use two cubic regression spline smooths for `wM_s95` and `Posan` (10 and 20 knots respectively) and a fixed effect for `Dow`. Fit the model (to speed up things set `optimizer = 'efs'` the `gam` call) and **before** converting it using `getViz` do

   ```
   fit$family$rd <- function(mu,wt,scale){
                        rnorm(nrow(mu), mu[ , 1], sqrt(scale/wt)/mu[ , 2])
   }
   ```

   where `fit` is the output of `gam`. This adds method for simulating responses from the `gaulss` family. Then convert it using `getViz` (remember to set `nsim = 50`) and call it `fit3`. Check whether there has been any improvement in AIC, and the check the conditional variance again using `l_gridCheck1D`. Is the variance pattern as strong as before?

6. * Now that we have a satisfactory model for the conditional variance, we look at further features of the residuals distribution. Plot a QQ-plot of the residuals of `fit3` using `qq.gam` with arguments `method = "simul1"` and `nrep = 50`. Do you see significant deviations from the model-based theoretical residuals distribution? Load the `e1071` package and use `check1D` with `l_gridCheck1D` and `gridFun = skewness` to verify how the skewness of the residuals varies along `wM_s95` and `Posan`. Do you see major departures from the model-based simulations?

7. * To allow the distribution of the response to be skewed we will now consider the `shash` distribution from the `mgcFam` package (see `?shash`). The `shash` family has four parameters, so we need to specify four linear predictors (location, scale, skewness and kurtosis in that order) in the model formula. For location and scale use the same models we used for `gaulss`, for the skewness include a fixed effect for `Dow` and a smooth effect for `Posan` (with `k = 10` and `bs='cr'`), while for the kurtosis use only an intercept (`~ 1`). Fit the model (use `optimizer = 'efs'` in `gam`), convert it and call it `fit4`. Check whether the AIC has improved, relative to `fit3` and produce another QQ-plot using `QQ-gam`. Are the deviations from the theoretical distribution larger or smaller in this model?

8. * Well, if you got here... congratulations you are an `mgcViz` ninja! What one could do at these point is to check how the kurtosis changes along the covariate using `l_gridCheck1D` (`e1071` provides a function called `kurtosis`). But beware: the `shash` is still experimental and model fitting might break down if you try to fit over-complicated models.

2 BAM modelling and looking for interactions

Here we use again data from the UK grid, but this data set is 48 times larger than the one used in exercise 1. This is because it contains data corresponding to all the 48 half-hourly slots. The variable `Instant` indicates the half-hourly window corresponding to each row of the data set. All remaining covariates have the same interpretation as in exercise 1. NB: after question 4 (included) you might need a computer with at least 8GB of RAM.
Questions:

1. Load `mgcViz` and the data (`data("UKload48")`). Create a model formula with smooth effects `wM`, `wM_s95`, `Instant`, `Trend` and `Posan`. Use regression splines bases (`bs='cr'`) for all smooths apart from `Posan`, for which you should use an adaptive basis (`bs='ad'`). Use `k = 6` for `Trend` and `k = 30` for `Posan`. Leave `k` to its default for the other smooths. Use parametric fixed effects for `Dow`, `NetDemand.48` and `Holy`. Use this formula within a `bam` call to fit a Gaussian GAM. When calling `bam` set `discrete = TRUE` to speed up computations (do this in all subsequent `bam` calls).

2. Having fitted the model, convert is to a `gamViz` object (called `fit0`) using the `getViz` function with argument `nsim = 50`. Now, use `check(fit0)` to verify whether we need to change the basis dimesion `k` for any of the smooth effects. Should we increase the `k` for `Tendance` and `Posan`? Also, check for residuals patterns along `NetDemand.48`, `wM`, `wM_s95` and `Instant` using the `check1D` function with the `l_gridCheck1D` layer.

3. Having chosen an appropriate basis dimension for each effect, refit the model and convert it a `gamViz` object called `fit1`. Repeat the checks using `check` and `check1D`, any improvement? Now that we are fairly satified with the marginal effects, we start looking for interactions. Use the `check2D` function with the `l_gridCheck2D` layer to look for systematic residual patterns across pairs of covariates. Hint: we might expect the effect of temperature, time of year and lagged load to change with `Instant`.

4. Recall that smooth tensor-product interactions are specified using `ti` (see `?ti`). Fit a model which includes the interactions you have identified in the previous step using the `bam` function (setting `discrete = TRUE` might provide a lot of speed-up here). Call `fit2` the corresponding `gamViz` object, and use again `check` and `check2D` to look for residuals patterns along two variables.

5. Assuming that we are now satisfied with our model, we'll now have a detailed look at the fitted smooth effects. First, look at the marginal effects effects using the `plot.gam` function. Use

`print(plot(fit2, select = ???), pages = 1)` to plot all the marginal effects on one page (substitute `???` with the indexes of the univariate effects in your model). Do the same to plot the 2D interactions. Think about whether each effect makes physical sense to you. As an alternative to `plot.gam`, recall that you can extract any effect using the `sm` function and produce a plot with customized layers. You can use the `listLayers` function to get a list of the available layers. Then, use the `plotRGL` function to manipulate each bivariate effect interactively.

6. * The model could be improved further. For instance, use the `check` and `check2D` functions with the `l_gridCheck1D` and `l_gridCheck2D` layers to look at how the standard deviation of the residuals varies across the covariates (the `e1071` package provides a `skewness` function, then you simply need to set `gridFun = skewness` in the call to `check2D`). Do you see any pattern? At this point we could consider a GAMLSS model with linear predictors for location, variance and skewness (e.g. the `gaulss` or `shash` family). However, `bam` methods does not yet support such models, so you'll need to use `gam` which can be much slower for large models.