# Canadian climate: function-on-function regression

Sarah Brockhaus *

*Institut für Statistik,*
Ludwig-Maximilians-Universität München,
Ludwigstraße 33, D-80539 München, Germany.

The analysis is based on the analysis in the online appendix of Scheipl et al. (2015). The results of this vigentte can be found in the web appendix of Brockhaus et al. (2015).

## 1 Load and plot data

Load data and choose the time-interval.

```
library(fda)
data("CanadianWeather", package = "fda")

### use data on a monthly basis (c.f. Scheipl et. al. online supplement)
dataM <- with(CanadianWeather,
        list(   temp = t(monthlyTemp),
                l10precip = t(log10(monthlyPrecip)),
                lat = coordinates[,"N.latitude"],
                lon = coordinates[,"W.longitude"],
                region = factor(region),
                place = factor(place)
        ))
# correct Prince George location (wrong at least until fda_2.2.7):
dataM$lon["Pr. George"] <- 122.75
dataM$lat["Pr. George"] <- 53.9

# center temperature curves:
dataM$tempRaw <- dataM$temp
dataM$temp <- sweep(dataM$temp, 2, colMeans(dataM$temp))

# define function indices
dataM$month.t <- 1:12
dataM$month.s <- 1:12
```

Plot the data

```
par(mfrow=c(1,2))

# plot precipitation
with(dataM, {
  matplot(t(l10precip), type = "l", lty = as.numeric(region),
          col = as.numeric(region),
          xlab = "", ylab = "",
```

*E-mail: sarah.brockhaus@stat.uni-muenchen.de

```
          ylim = c(-1.2, 1))#, cex.axis = 1, cex.lab = 1)
  legend("bottom", col = 1:4, lty = 1:4, legend = levels(region),
         cex = 1, bty = "n")
})
mtext("time [month]", 1, line = 2)#, cex = 1.5)
mtext("log(precipitation) [mm]", 2, line = 2)#, cex = 1.5)

# plot temperature
with(dataM, {
  matplot(t(tempRaw), type = "l", lty = as.numeric(region),
          col = as.numeric(region),
          xlab = "", ylab = "")#,
#          cex.axis = 1, cex.lab = 1)
})
mtext("time [month]", 1, line = 2)#, cex = 1.5)
mtext("temperature [C]", 2, line = 2)#, cex = 1.5)
```

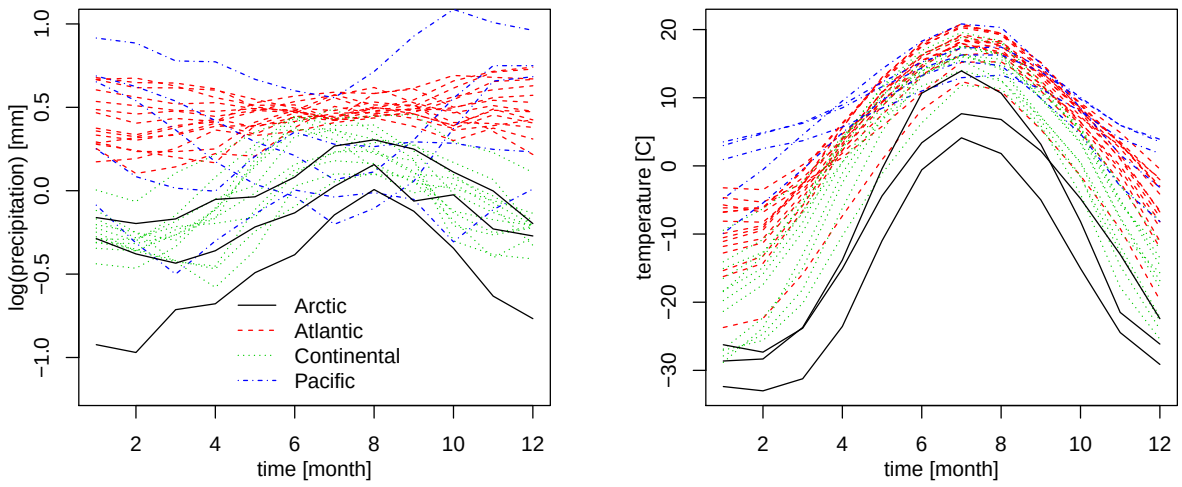


Figure 1: Monthly average temperature and log-precipitation at 35 locations in Canada. Regions are coded by colors and different line types.

## 2 Model for log-precipitation

We consider the following linear regression model:

$$E(Y_i(t)|x_i) = I(\text{rg}_i = k)\beta_k(t) + \int \text{temp}_i(s)\beta(s,t)ds + e_i(t),$$

where $Y_i(t)$ is the log-precipitation over month $t = 1, \ldots, 12$, $I(\cdot)$ is the indicator function, $\text{rg}_i$ is the region of the $i^{th}$ station, $\beta_k(t)$ are the smooth effects per region, $\text{temp}_i(s)$ is the temperature over the month $s = 1, \ldots, 12$, $\beta(s,t)$ is the coefficient surface and $e_i(t)$ are smooth spatially correlated residual curves.

Set up design matrix and penalty-matrix for spatially correlated residual curves.

```
locations <- cbind(dataM$lon, dataM$lat)
### fix location names s.t. they correspond to levels in places
rownames(locations) <- as.character(dataM$place)

library(fields)
  ### get great circle distances between locations:
  dist <- rdist.earth(locations, miles = FALSE, R = 6371)

  ### construct Matern correlation matrices as
  ### marginal penalty for a GRF over the locations:
  ### find ranges for nu = .5, 1 and 10
  ### where the correlation drops to .2 at a distance of 500/1500/3000 km
  ### (about the 10%/40%/70% quantiles of distances here)
  r.5 <- Matern.cor.to.range(500, nu = 0.5, cor.target = .2)
  r1 <- Matern.cor.to.range(1500, nu = 1.0, cor.target = .2)
  r10 <- Matern.cor.to.range(3000, nu = 10.0, cor.target = .2)
  ### compute correlation matrices
  corr_nu.5 <- apply(dist, 1, Matern, nu = .5, range = r.5)
  corr_nu1 <- apply(dist, 1, Matern, nu = 1, range = r1)
  corr_nu10 <- apply(dist, 1, Matern, nu = 10, range = r10)
  ### invert to get precisions
  P_nu.5 <- solve(corr_nu.5)
  P_nu1 <- solve(corr_nu1)
  #P_nu10 <- solve(corr_nu10)

if(FALSE){
  curve(Matern(x, nu = .5, range = r.5), 0, 5000, ylab = "Correlation(d)",
        xlab = "d [km]", lty = 2)
  curve(Matern(x, nu = 1, range = r1), 0, 5000, add = TRUE, lty = 3)
  curve(Matern(x, nu = 10, range = r10), 0, 5000, add = TRUE, lty = 4)
  legend("topright", inset = 0.2, lty = c(2, NA, 3, NA, 4),
         legend = c(".5", "", "1", "", "10"),
       title = expression(nu), cex = .8, bty = "n")
}


## for uncorrelated resiaudls
#   P_nu.5 <- diag(35)
#   print("Residuals are uncorrelated!")
```

Fit the model.

```
# use bolsc() base-learner with precision matrix as penalty matrix
set.seed(210114)
```

```
mod3 <- FDboost(l10precip ~ bols(region, df = 2.5, contrasts.arg = "contr.dummy")
                + bsignal(temp, month.s, knots = 11, cyclic = TRUE,
                          df = 2.5, boundary.knots = c(0.5, 12.5), check.ident = FALSE)
                + bolsc(place, df = 2.5, K = P_nu.5, contrasts.arg = "contr.dummy"),
                timeformula = ~ bbs(month.t, knots = 11, cyclic = TRUE,
                                    df=3, boundary.knots = c(0.5, 12.5)),
                offset="scalar", offset_control = o_control(k_min = 5),
                data=dataM)

## Warning in model.matrix.default(as.formula(fm), data = mf, contrasts.arg = args$contrasts.arg):
non-list contrasts argument ignored
## Warning in model.matrix.default(as.formula(fm), data = mf, contrasts.arg = args$contrasts.arg):
non-list contrasts argument ignored
## Warning in model.matrix.default(as.formula(fm), data = mf, contrasts.arg = args$contrasts.arg):
non-list contrasts argument ignored
```

Get optimal stopping iteration by 25-fold bootstrap over curves.

```
mod3 <- mod3[47]
```

Do not run bootstrapping.

```
set.seed(2303)
folds <- cvMa(ydim = mod3$ydim, type = "bootstrap", B = 25)
cvMod3 <- cvrisk(mod3, grid = seq(1, 1000, by=1), folds = folds, mc.cores = 10)
mod3 <- mod3[mstop(cvMod3)] # 47
# summary(mod3)
```

Base-learner for smooth residuals is not selected into the model. Look at effects of region and temperature.

```
par(mfrow=c(1,2))#, mar = c(7,4,7,1))#, cex = 1.5, cex.main = 0.9)
predRegion <- predict(mod3, which = 1,
                      newdata = list(region = factor(c("Arctic", "Atlantic",
                                                       "Continental", "Pacific")),
                                     month.t = seq(1, 12, l=20))) + mod3$offset
matplot(seq(1, 12, l = 20), t(predRegion), col = 1:4,
        type = "l", lwd = 2, lty = 1:4,
        main = "region", ylab = "", xlab = "")
mtext("t, time [month]", 1, line = 2)#, cex = 1.5)

legend("bottom", lty = 1:4, legend = levels(dataM$region), col = 1:4, bty = "n", lwd = 2)

## plot the effect of temperature
# par(mar = c(4,4,7,1))
plot(mod3, which = 2, pers = TRUE, main = "temperature", zlab = "",
     xlab = "s, time [month]", ylab = "t, time [month]")
```

Compute optimal stopping iteration by leaving-one-curve-out cross-validation.

```
mod3 <- mod3[750]
```

```
set.seed(143)
folds <- cvMa(ydim = mod3$ydim, type = "curves")
cvMod3curves <- cvrisk(mod3, grid = seq(1, 1000, by = 1), folds = folds, mc.cores = 12)
```

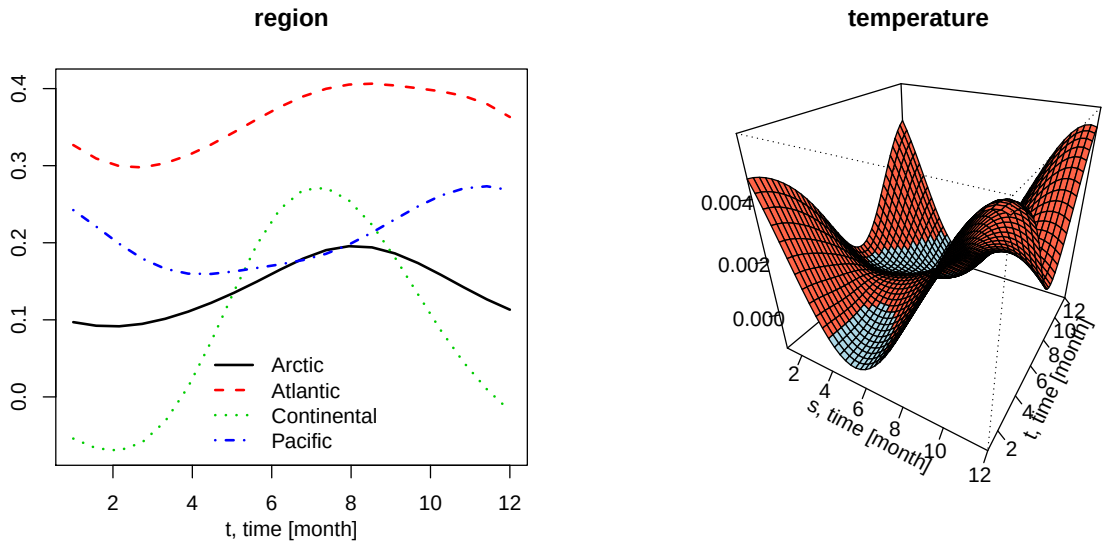


Figure 2: Coefficients for model with 49 boosting iterations (determinded by 25 fold bootstrap). The estimated coefficients for the four climatic regions are plotted with color coded regions (left panel). The coefficient function for the functional effect of temperature is colored in red for positive values and in blue for negative values (right panel).

```
## optimal stopping iteration in terms of mean
mstop(cvMod3curves)
## optimal stopping iteration in terms of median
(mStop <- which.min(apply(cvMod3curves, 2, median)) )
mod3 <- mod3[mStop]
```

Plot the coefficient functions for the effects of temperature and region.

```
par(mfrow=c(1,2))
predRegion <- predict(mod3, which=1,
                      newdata = list(region = factor(c("Arctic", "Atlantic",
                                                       "Continental", "Pacific")),
                                     month.t=seq(1, 12, l = 20))) + mod3$offset
matplot(seq(1, 12, l=20), t(predRegion), col = 1:4,
        type = "l", lwd = 2, lty = 1:4,
        main = "region", ylab = "", xlab = "")
mtext("t, time [month]", 1, line = 2)#, cex = 1.5)
#plot(mod, which=1, lwd=2, lty=1, col=c(2,3,4,1))
legend("bottom", lty = 1:4, legend = levels(dataM$region), col = 1:4, bty = "n", lwd = 2)

## plot the effect of temperature
plot(mod3, which = 2, pers = TRUE, main = "temperature", zlab = "",
     xlab = "s, time [month]", ylab = "t, time [month]")
```

Prepare data for plot of smooth residual curves. The stations a roughly ordered.

```
ord <- c("Dawson", "Whitehorse", "Yellowknife", "Uranium City", "Churchill",
         "Edmonton",  "Pr. Albert", "The Pas", "Calgary", "Regina", "Winnipeg",
         "Thunder Bay",
         "Pr. George", "Pr. Rupert", "Kamloops", "Vancouver", "Victoria",
```

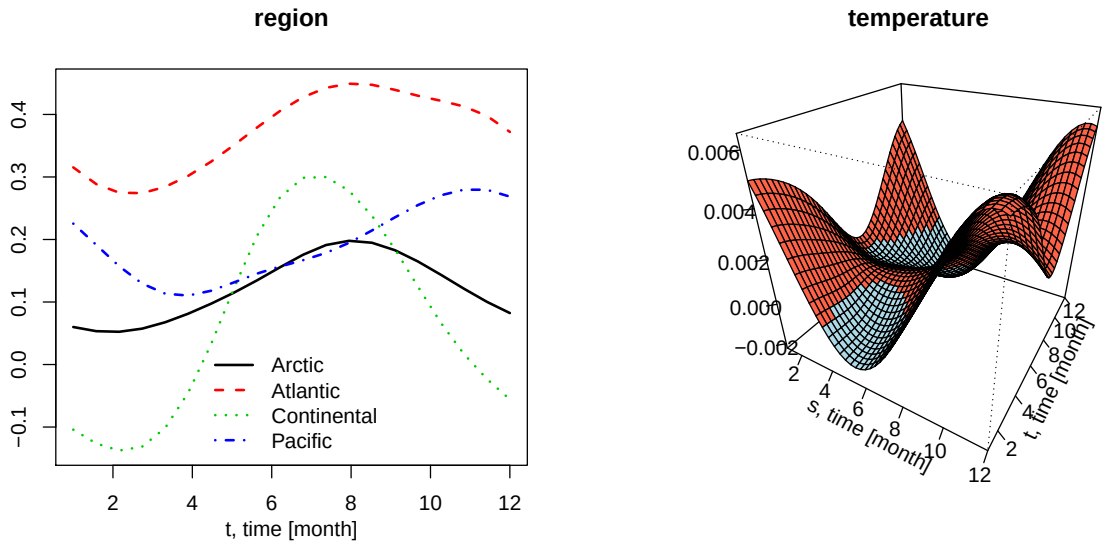


Figure 3: Coefficients for model with 750 boosting iterations (determinded by leaving-one-curve-out cross-validation). The estimated coefficients for the four climatic regions are plotted with color coded regions (left panel). The coefficient function for the functional effect of temperature is colored in red for positive values and in blue for negative values (right panel).

```
          "Scheffervll", "Bagottville", "Arvida", "St. Johns", "Quebec",
          "Fredericton", "Sydney", "Ottawa", "Montreal", "Sherbrooke", "Halifax",
          "Yarmouth", "Toronto", "London", "Charlottvl",
          "Inuvik",  "Resolute", "Iqaluit" )
ind <- sapply(1:35, function(s){ which(dataM$place == ord[s]) })
smoothRes <- predict(mod3, which=3)
if( is.null(dim(smoothRes)) ) smoothRes <- matrix(0, ncol = 12, nrow = 35)
smoothRes <- (smoothRes )[ind, ]
# smoothRes <- ( predict(mod4, which=3) )[ind, ]
regionOrd <- dataM$region[ind]

fit3 <- (predict(mod3))[ind, ]
response <- dataM$l10precip[ind, ]
```

Plot the smooth residual curves.

```
par(mar = c(2.55, 2.05, 2.05, 1.05), oma=c(0, 0, 0, 0))
layout(rbind(matrix(1:36, 6, 6), rep(37, 6), rep(37, 6)))
for(i in 1:35) {
  plot(1:12, smoothRes[i, ], col = as.numeric(regionOrd[i]), type = "l",
       ylim = range(smoothRes, response-fit3),
       main = paste(ord[i], " (", i, ")", sep = ""),
       cex = 1.2, cex.axis = .8, ylab = "", xlab = "")
  abline(h = 0, col = 8)
  lines(1:12, smoothRes[i, ], col = as.numeric(regionOrd[i]))
  points(1:12, response[i, ] - fit3[i, ], cex = 0.8)
}
plot(0, 0, col = "white", xaxt = "n", yaxt = "n", bty = "n")

if(require(maps) & require(mapdata)){
```

```
  mapcanada <- map(database="world", regions="can", plot=FALSE)
  plot(mapcanada, type = "l", xaxt = "n", yaxt = "n", ylab = "", xlab = "", bty = "n",
     xlim = c(-141, -50), ylim=c(43, 74),
     col = "grey", mar = c(0, 0, 0, 0))
  for(i in 1:35) {
  text(-dataM$lon[ind[i]], dataM$lat[ind[i]], col = as.numeric(regionOrd[i]),
       labels = as.character(i), cex = 0.8)
 }
}
```

# References


Brockhaus S, Scheipl, F., Hothor, T., and Greven, S. (2015), The functional linear array model, *Statistical Modelling*, 15(3), 279–300.

Scheipl, F., Staicu, A.-M., and Greven, S. (2015), Functional Additive Mixed Models, *Journal of Computational and Graphical Statistics*, 24(2), 477–501.

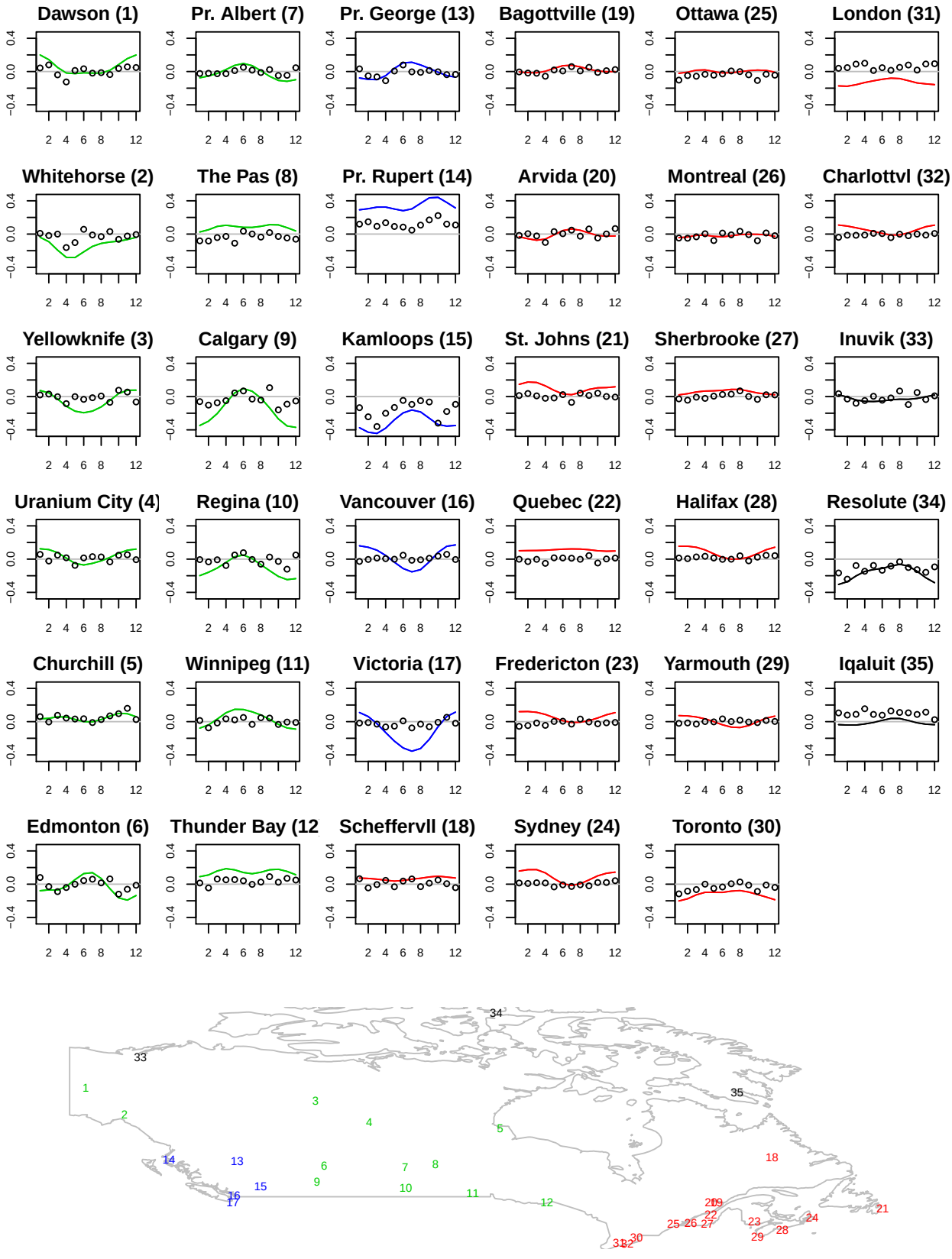


Figure 4: The estimated smooth spatially correlated residual curves (lines) and the residuals (circles) are plotted with regions color-coded. The locations of the weather stations are given in the map at the bottom.