Using the R package **kergp**: Group kernels

Olivier Roustant

23rd September, 2019

This report[1] illustrates the use of so-called *group kernels*, investigated in [Roustant et al., 2018], with **kergp**. They are suitable for categorical inputs when levels are gathered in predefined groups.

1 Group kernels

Let G be the number of groups, and $n_1, \ldots, n_G$ their sizes. Let $L = n_1 + \cdots + n_G$ be the total number of levels. Then a group kernel is a valid block covariance matrix $\mathbf{T}$ of size L, composed of constant off-diagonal blocks $\mathbf{T}_{g,g'}$, $(g \neq g')$. Up to a condition on diagonal blocks, such block matrices can be generated with

- one *'between-{group means}'* covariance matrix $\mathbf{B}^\star$ of size G, corresponding to the covariance matrix of the group centers,
- and G *'within'* centered covariance matrices $\mathbf{W}_1^\star, \ldots, \mathbf{W}_G^\star$ of size $n_1, \ldots, n_G$.

The link between $\mathbf{T}, \mathbf{B}^\star, \mathbf{W}_g^\star$ corresponds to a probabilistic nested model (see [Roustant et al., 2018] for details) and is given by the following equations:

$$\mathbf{T}_{g,g} = B^\star_{g,g}\, \mathbf{J}_{n_g,n_g} + \mathbf{W}_g^\star \quad (1)$$

$$\mathbf{T}_{g,g'} = B^\star_{g,g'}\, \mathbf{J}_{n_g,n_{g'}} \qquad g \neq g' \quad (2)$$

where $\mathbf{J}_{k,k'}$ is the $k \times k'$ matrix of ones. More precisely, the within covariance matrices are parameterized with full rank matrices $\mathbf{M}_1, \ldots, \mathbf{M}_G$ of size $n_1 - 1, \ldots, n_G - 1$ by

$$\mathbf{W}_g^\star = \mathbf{A}_g \mathbf{M}_g \mathbf{A}_g^\top \quad (3)$$

where $\mathbf{A}_1, \ldots, \mathbf{A}_G$ are fixed *contrast* $n_G \times (n_G - 1)$ matrices, whose columns form an orthonormal basis of centered vectors of $\mathbb{R}^{n_G}$.

In **kergp**, the *between* and *within* covariance matrices can be chosen among three parameterizated classes:

- `"Diag"`: Diagonal matrices,
- `"compSymm"`: Coumpound symmetry (or exchangeable) matrices,
- `"Symm"`: General symmetric matrices.

Furthermore, two other parameters can be used to refine parameterization settings:

[1] Compiled with **R** 3.6.1 using **kergp** 0.5.0.

- `covBet`. A character string indicating the kind of covariance matrix for $\mathbf{B}^\star$: correlation (`"corr"`), homoscedastic (`"homo"`) or heteroscedastic (`"hete"`). Partial matching is allowed.
- `covWith`. A vector of size G (recycled if smaller) whose component g indicates the kind of covariance matrix for group g : correlation (`"corr"`), homoscedastic (`"homo"`) or heteroscedastic (`"hete"`). Partial matching is allowed.

Mind that, even when $\mathbf{B}^\star$ and $\mathbf{M}$ are chosen to be correlation matrices (`covBet = "corr", covWith = "corr"`), then $\mathbf{T}$ will not be a correlation matrix in general (see Eq 1). Finally, notice that the model choice (`within = "Diag", covWith = "homo"`) implies that the corresponding diagonal block is a compound symmetry matrix.

2 Illustration on a toy example

We consider a real-valued output $y = f(x, u)$ depending on one continuous input $x \in [0, 1]$ and one categorical input u with $L = 8$ levels. As visible in Figure 1, the levels can be split in 2 groups: a group of linear functions (levels 1–4), and a group of damped sinusoidal functions (levels 5–8).

```
library(kergp)
n1 <- 4
n2 <- 6
L <- n1 + n2
cut <- c(0, n1, n1 + n2 / 2, L)
f <- function(x, u){
  (x + 0.01 * (x - 1/2) ^ 2) * u / 10 * ((u >= cut[1] + 1) & (u <= cut[2])) +
  0.9 * cos(2 * pi * (x + (u - cut[2]) / 20 )) * exp(-x) * ((u >= cut[2] + 1) & (u <= cut[3])) -
  0.7 * cos(2 * pi * (x + (u - cut[3]) / 20 )) * exp(-x) * ((u >= cut[3] + 1) & (u <= cut[4]))
}
N <- 100
t <- seq(0, 1, length.out = N)
x <- rep(t, L)
u <- rep(1:L, each = N)
col <- lty <- c(rep("blue", n1), rep("red", n2))

# a basic stratified design
m <- 3  # number of points per level
set.seed(0)
x1 <- seq(0, 1, length.out = m * L)
u1 <- rep(sample(L), m)
Y <- f(x1, u1)

# graphics
matplot(t, matrix(f(x, u), ncol = L), type = "l", xlab = "x", ylab = "f(x,u)", col = 1:L)
points(x1, Y, pch = 19, col = u1)
```

We now aim at predicting these functions from a limited number of evaluations. We have chosen here a stratified design extracted from a sequence of regularly spaced points, with m points per level. They correspond to the black points in Figure 1. Alternatively, a sliced Latin hypercube design could have been used [Ba, 2015].

First we define a group kernel. Here, we choose the simplest structure for the first group (linear functions), but the most complex one for the second one (damped sinusoidal functions)

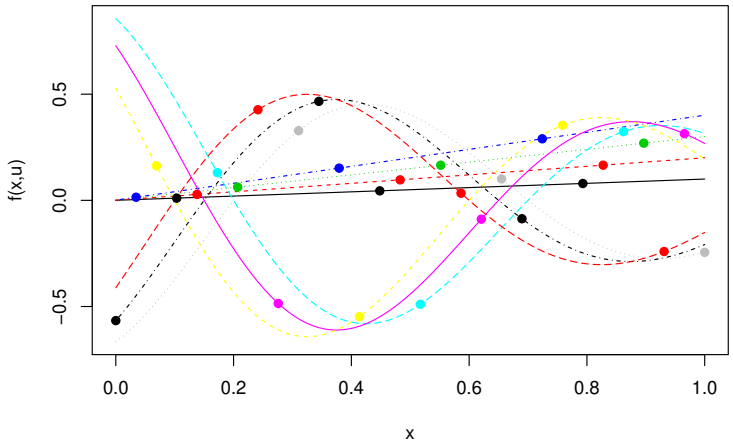

Figure 1: A toy function for group kernels.

in order to capture the negative correlations within that group. Notice that the model choice for the first group (`covBet = "Diag", covWith = "homo"`) implies that the diagonal block corresponding to the first group is a compound symmetry matrix. In general, these settings are typically motivated by a tradeoff between complexity and parsimony.

```
twoGroupsList <- list(seq(1, n1),
                      seq(n1 + 1, length.out = n2))
kcat2 <- covQualNested(input = "u1",
                       groupList = twoGroupsList,
                       between = "Symm", within = c("Diag", "Symm"),
                       covBet = "homo", covWith = c("homo", "hete"))
```

Now, a kernel for the mixed input $\mathbf{w} = (x, u)$ is obtained by multiplying the kernel for the categorical input u defined by $\mathbf{T}$ with a 1-dimensional kernel for the continuous input:

$$k(\mathbf{w}, \mathbf{w}') = k_{\text{cont}}(x, x')k_{\text{cat}}(u, u').$$

Other operations are possible, such as the sum. Below, we have chosen a Matérn kernel with $\nu = 5/2$. In order to avoid overparameterization, the variance term of the continuous kernel is fixed to 1 (by choosing `cov = "corr"`): thus the kernel variance is entirely parameterized by the categorical one.

```
kcont <- covRadial(k1Fun1 = k1Fun1Matern5_2, d = 1, cov = "corr")
inputNames(kcont) <- "x1"
coef(kcont) <- c(0.5)  # initial value

kmix2 <- covComp(formula = ~ kcont() * kcat2())
```

Next, the Gaussian process model is estimated from the data. The usage of `multistart` is recommended in order to cope with possible local maxima of the log-likelihood function.

```
myData <- data.frame(u1 = u1, x1 = x1, Y = Y)

library(doFuture)
registerDoFuture()
plan(multisession, workers = max(detectCores()-1, 1))
multistart <- 10
```

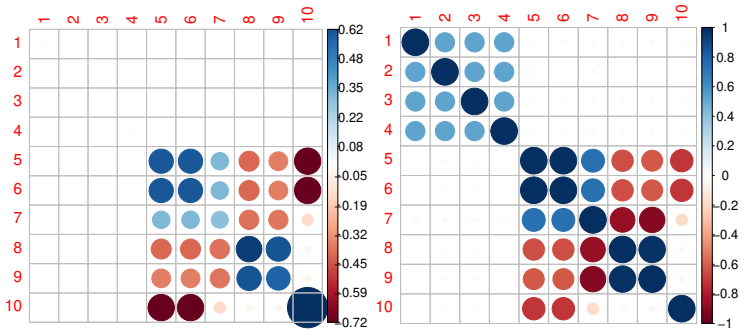

Figure 2: Color plot of the estimated *covariance* matrix $\mathbf{T}$ (left) and corresponding *correlation* matrix (right) for the categorical kernel.

```
res2 <- gp(formula = Y ~ 1, cov = kmix2,
           data = myData, inputs = c("u1", "x1"),
           multistart = multistart, noise = FALSE)
```

Then, we plot the estimated covariance matrix of the categorical part, obtained by choosing a vector of zeros for the continuous input. We also report the resulting correlation matrix in Figure 2. We can see that the model recovers the likely sign of correlations, as well as the group heteroskedascity with a smaller variance for the group of linear curves.

```
library(corrplot)
T <- as.list(res2$covariance)$kcat2
plot(T, type = "cov")
```

```
plot(as.list(res2$covariance)$kcat2, type = "cor")
```

Next we try a second model obtained by considering the two subgroups of damped sinusoidal curves as two different groups. Since the groups are now homogeneous, we choose the simplest covariance homoscedastic within-structure (`within = "Diag", covWith = "homo"`) for each group, corresponding to a compound symmetry matrix. On the other hand, for flexibility, we choose the most complex between structure (`between = "Symm"`, `covBet = "hete"`), resulting in different correlation values for each pair of groups.

```
threeGroupsList <- list(seq(1, n1),
                        seq(n1 + 1, length.out = n2/2),
                        seq(n1 + n2/2 + 1, length.out = n2/2))
kcat3 <- covQualNested(input = "u1",
                       groupList = threeGroupsList,
                       between = "Symm", within = "Diag",
                       covBet = "hete", covWith = "homo")
kmix3 <- covComp(formula = ~ kcont() * kcat3())
```

Then, we estimate this new GP model and draw the estimated covariance / correlation plot. We obtain a similar picture.

```
res3 <- gp(formula = Y ~ 1, cov = kmix3,
           data = myData, inputs = c("u1", "x1"),
           multistart = multistart, noise = FALSE)
```

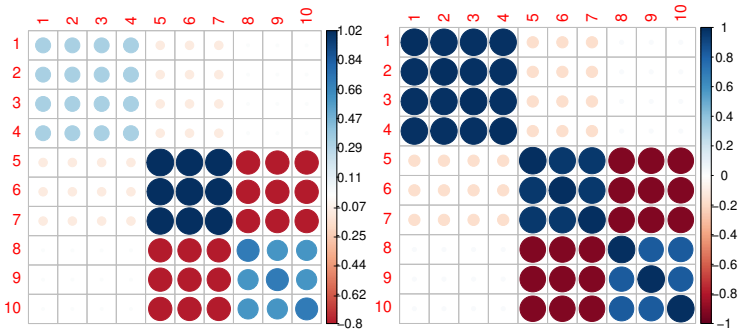

Figure 3: Color plot of the estimated *covariance* matrix $\mathbf{T}$ (left) and corresponding *correlation* matrix (right) for the categorical kernel.

```
plot(as.list(res3$covariance)$kcat3, type = "cov")
```

```
plot(as.list(res3$covariance)$kcat3, type = "cor")
```

Let us now look at the performance in prediction for the two models. We sample uniformly $N = 1000$ new data, and compute predictions at these points. The result is plotted on Figure 4. The Q^2 of each model is also indicated, which compares the predicted values to the mean. We also give the Q^2 corresponding to each level.

```
N <- 1000
newdata <- data.frame(x1 = runif(N), u1 = sample(L, N, replace = TRUE))
Q2 <- function(obs, pred){
  1 - sum( (pred - obs) ^ 2 ) / sum( (mean(obs) - obs) ^ 2 )
}
byLevelQ2 <- function(obs, pred, newdata){
  val <- c()
  for (i in 1:L){
    index <- newdata$u1 == i
    val <- c(val, Q2(obs = obs[index], pred = pred[index]))
  }
  val
}
```

```
p2 <- predict(res2, newdata, type = "UK")
matplot(t, matrix(f(x, u), ncol = L), type = "l", xlab = "x", ylab = "f(x,u)", col = 1:L)
points(x1, Y, pch = 19, cex = 1, col = u1)
points(newdata$x1, p2$mean, pch = newdata$u1, col = newdata$u1, cex = 0.3)
```

```
p3 <- predict(res3, newdata, type = "UK")
matplot(t, matrix(f(x, u), ncol = L), type = "l", xlab = "x", ylab = "f(x,u)", col = 1:L)
points(x1, Y, pch = 19, cex = 1, col = u1)
points(newdata$x1, p3$mean, pch = newdata$u1, col = newdata$u1, cex = 0.3)
```

```
Ynew <- f(x = newdata$x1, u = newdata$u1)
cat("Q2 for the model with 2 groups:\n", round(Q2(obs = Ynew, pred = p2$mean), 2))
```

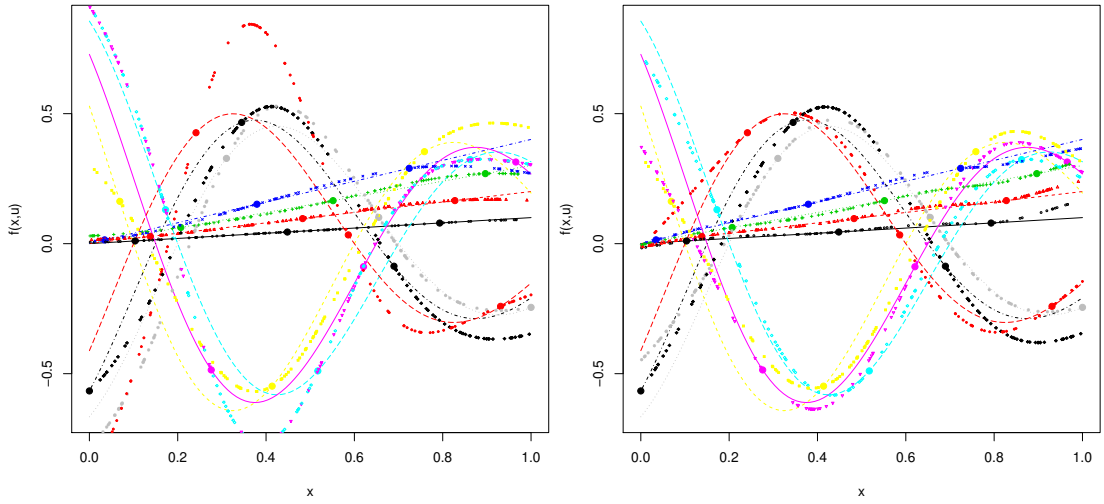

Figure 4: Prediction for the two models (left: with two groups; right: with three groups). The original data are represented by big bullets.

```
## Q2 for the model with 2 groups:
##  0.83

cat("Q2 for the model with 3 groups:\n", round(Q2(obs = Ynew, pred = p3$mean), 2))

## Q2 for the model with 3 groups:
##  0.94
```

```
cat("by-level Q2 for the model with 2 groups:\n",
    round(byLevelQ2(obs = Ynew, pred = p2$mean, newdata), 2))

## by-level Q2 for the model with 2 groups:
##  1 0.98 0.98 0.89 0.91 0.85 0.92 0.91 0.93 0.18

cat("by-level Q2 for the model with 3 groups:\n",
    round(byLevelQ2(obs = Ynew, pred = p3$mean, newdata), 2))

## by-level Q2 for the model with 3 groups:
##  0.68 0.97 0.99 0.98 0.98 0.92 0.93 0.93 0.94 0.9
```

Finally, we provide the leave-one-out diagonostic plots, in order to have a first insight of the validity of the two GP models.

```
plot(res2)
```

```
plot(res3)
```

References

[Ba, 2015] Ba, S. (2015). *SLHD: Maximin-Distance (Sliced) Latin Hypercube Designs*. R package version 2.1-1.

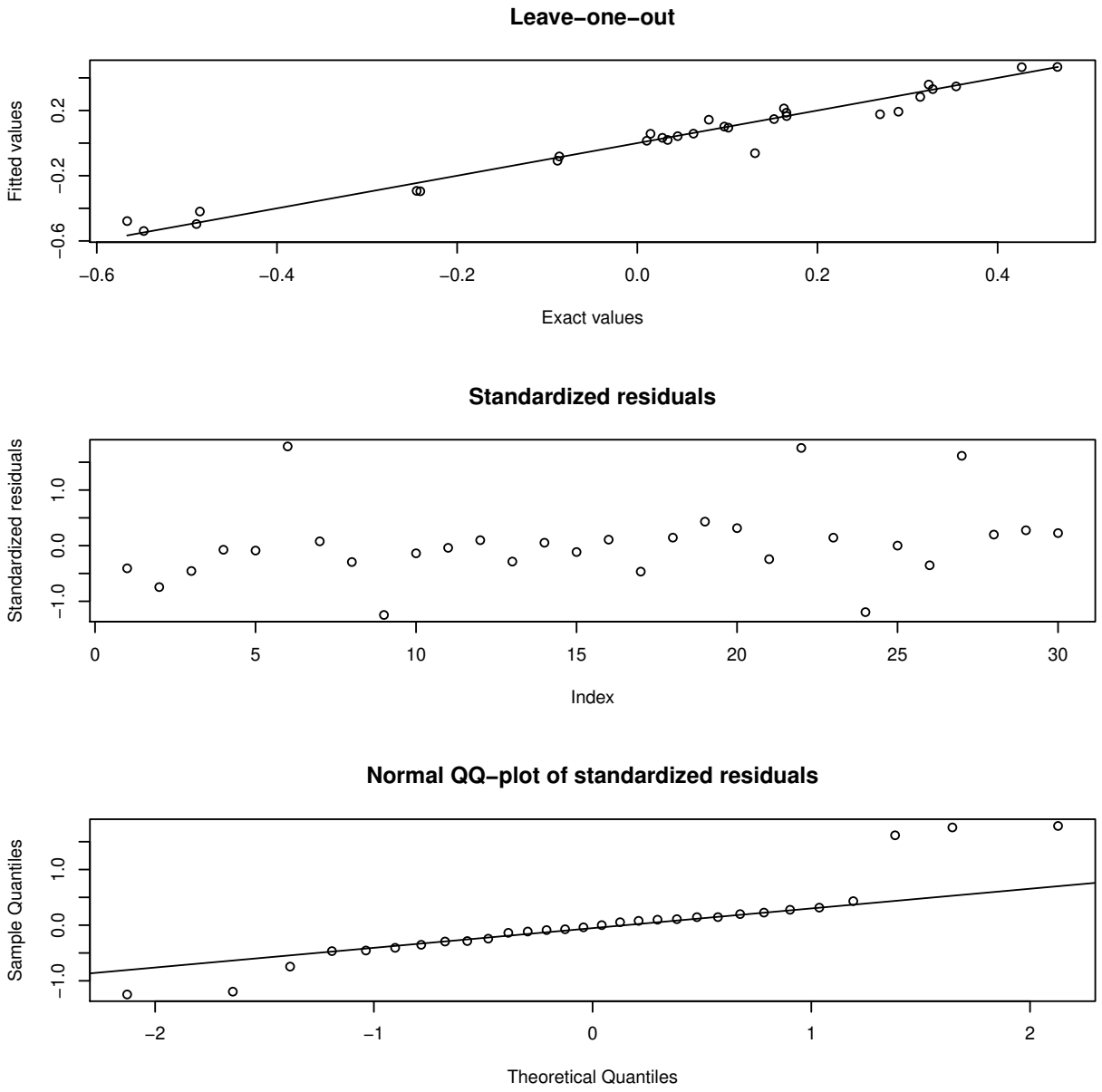

Figure 5: Leave-one-out plots for the GP models with 2 groups.

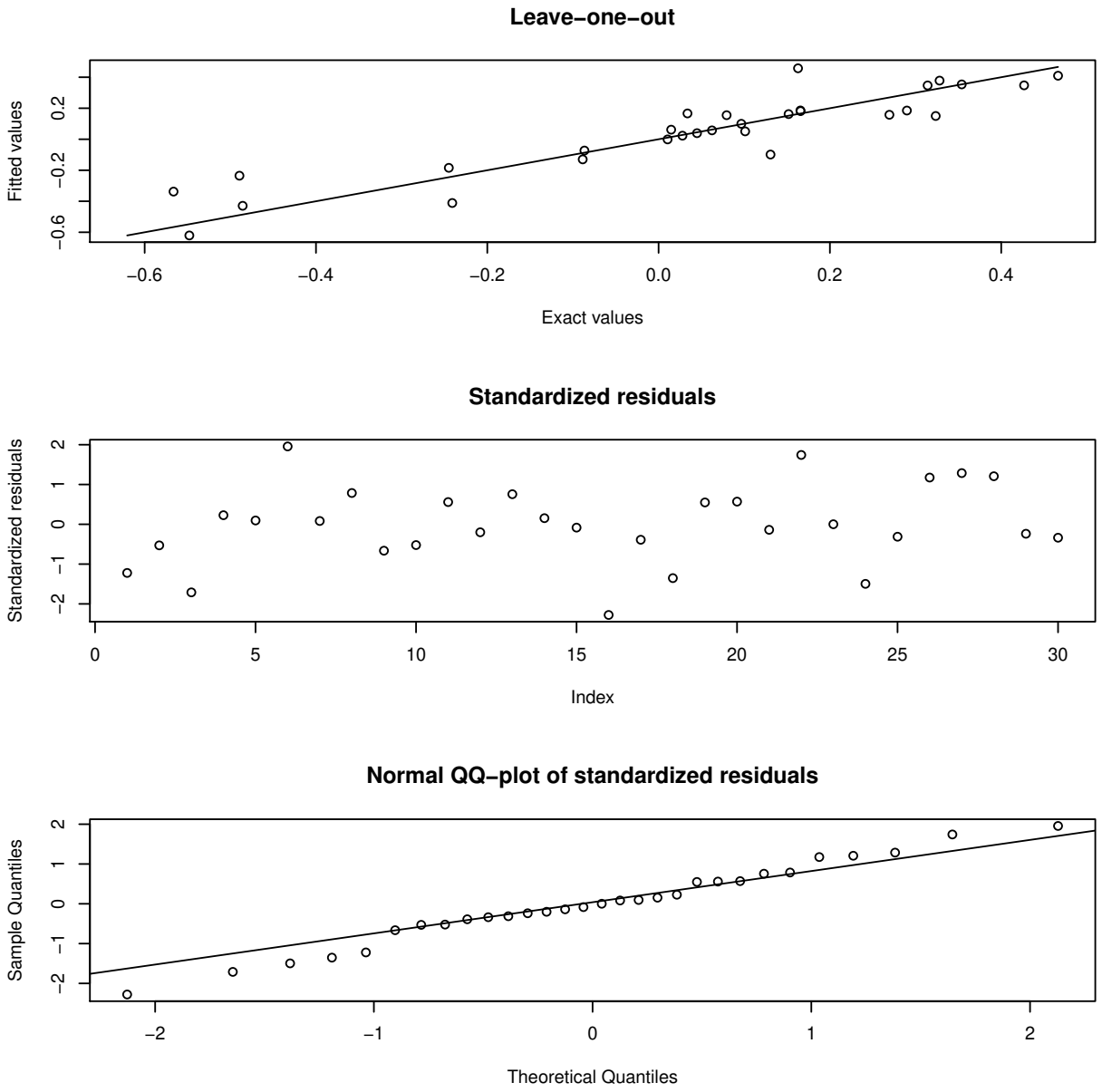

Figure 6: Leave-one-out plots for the GP models with 3 groups.

[Roustant et al., 2018] Roustant, O., Padonou, E., Deville, Y., Clément, A., Perrin, G., Giorla, J., and Wynn, H. (2018). Group kernels for Gaussian process metamodels with categorical inputs. Technical report, https://arxiv.org/abs/1802.02368.