R experiments of GP sequntial design based on hetGP package

Zhu Huanyan

This experiment aims to compare the performance of objective functions in GP sequential design under different conditions.
Three types of train data are generated, including sequence, uniform distribution, and quasi-normal distribution. The test data follows the same distribution of the train data.
A heteroscedastic GP model is fitted by mleHetGP() function. This function will fit not only the main GP, but also a second GP for noise process.
Two objective functions are considered, IMSPE and tMSE(Picheny, Ginsbourger, Roustant, Haftka, and Kim 2010).
After training and testing, rmse is calculated to evaluate the fitted model.

1. experiment design

In the process of creating data, we assume the input domain is [0,1]. The true function is:

$$f1d2(x) = 2(e^{-30(x-0.25)^2} + sin(\pi x^2)) - 2$$

The noise has 0 mean and variance function:

$$r(x = (x_1, ..., x_m)) = \frac{1}{3} * (e^{sin(2\pi \sum_{i=1}^{n} x_i)})$$

The final observation will be:

$$y(x) = f1d2(x) + rnorm(0, r(x))$$

Assume X, Y are the initial N inputs and observations, a heteroscedastic GP model is fitted:

$$model = mleHetGP(X, Y, cov = c("Gaussian", "Matern5.2", "Matern3.2"))$$

Conditioned on the current N observations, the optimization problem is:

$$X_{N+1} = \underset{x_{new}}{\arg\min}\ IMSE(X_N, x_{new})$$

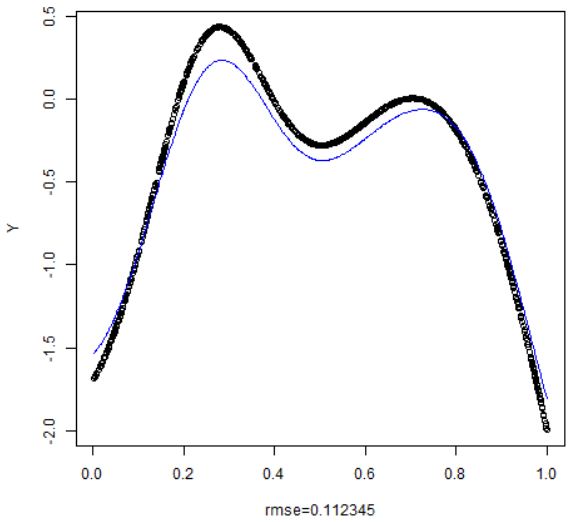
After the sequential runs, rmse is used to evaluate different objective functions.

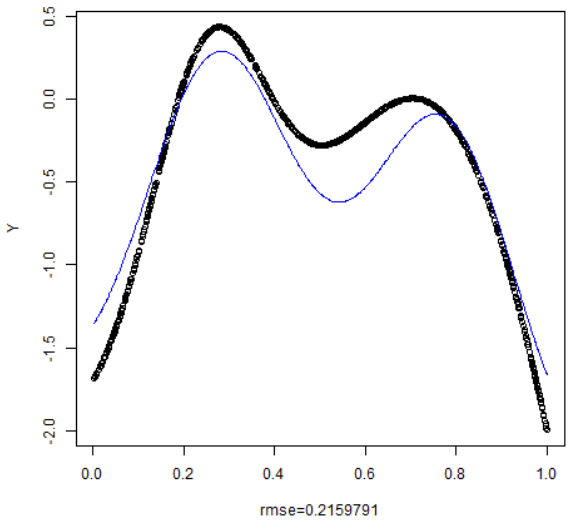
$$rmse^2 = \frac{1}{iterations}(gp.mean(X_{test}) - f1d2(X_{test}))^2$$

2. tests and reports

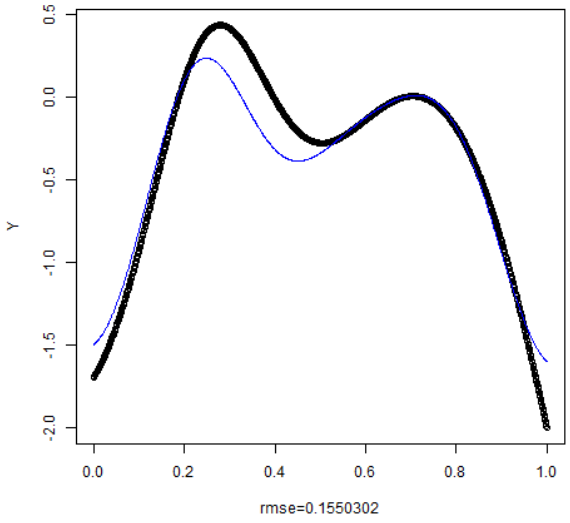
2.1 squared exponential kernel

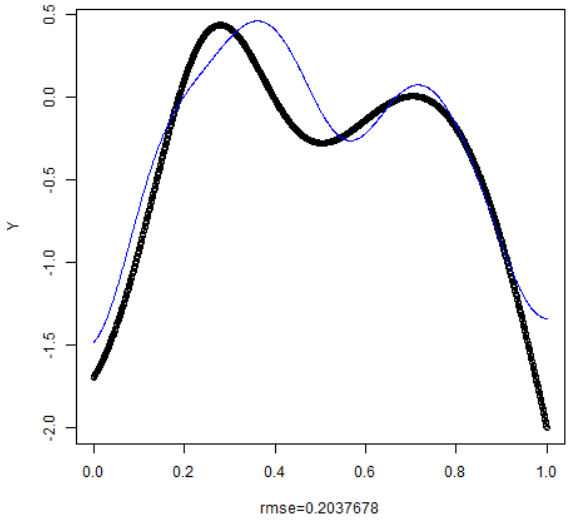
This group of test is based on squared exponential kernel. The initial input size is 10, with one replication. The train step is 100.
The results of uniform input:



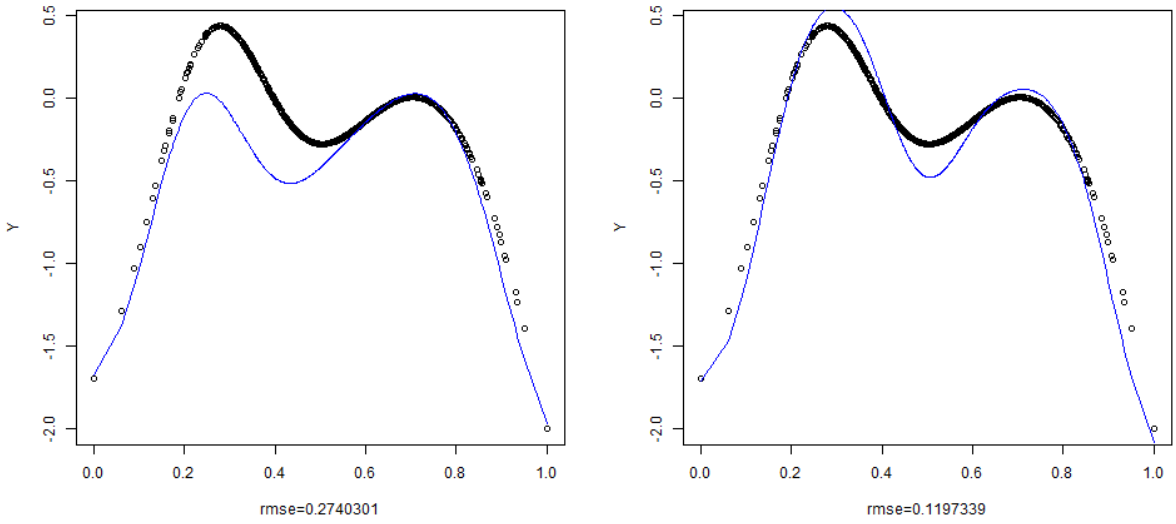


The results of sequence input:





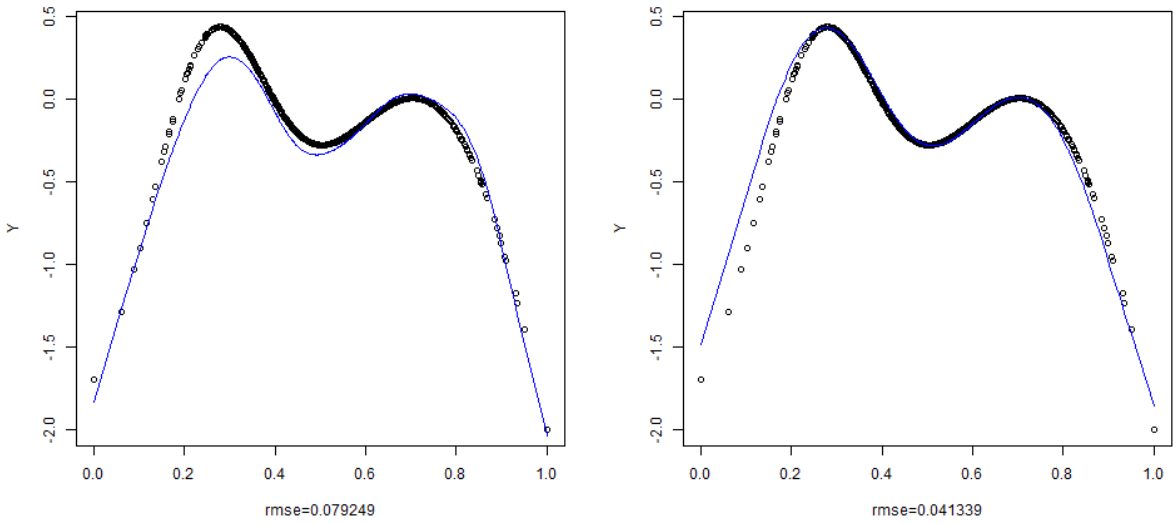
In below experiment, the input X is created from standard normal distribution. However, the data has to be normalized so that it is within the domain [0, 1], so it is a quasi-norm distribution. The results of quasi-normal input:



According to the above runs, the performance of IMSPE is better than tMSE if the input data is a sequence or uniformly distributed. However, if the input data has a more complex distribution, the tMSE criterion works better.

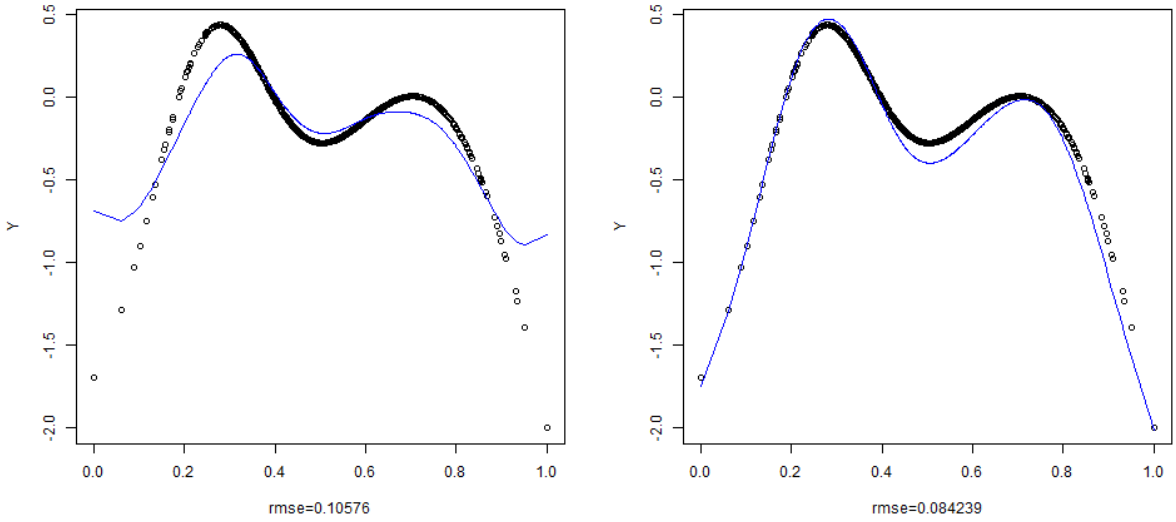
2.2 Matern kernel

Here is another run of quasi-normal data, the train step is increased to 300:



In this case, tMSE is also better than IMSPE.

In the following run, replication is removed and rmse increases, so it shows replication can lead to a better result:



In conclusion, if input data is uniformly distributed or is a sequence, rmse of IMSPE is less than tMSE. However, if input data follows a more complex distribution, tMSE works better.

3. future discussions

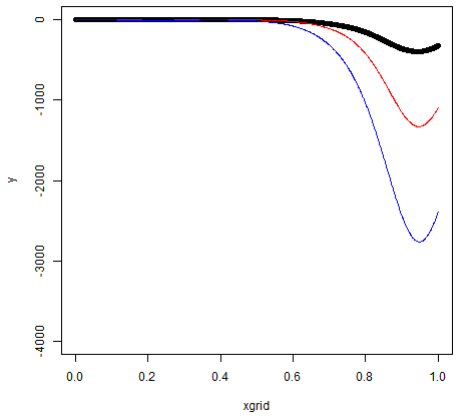
The MSE of best linear predictor:

$$MSE(X_0) = K(X_0, X_0) - K^T(X_0, X_N)K_N^{-1}K(X_0, X_N)$$

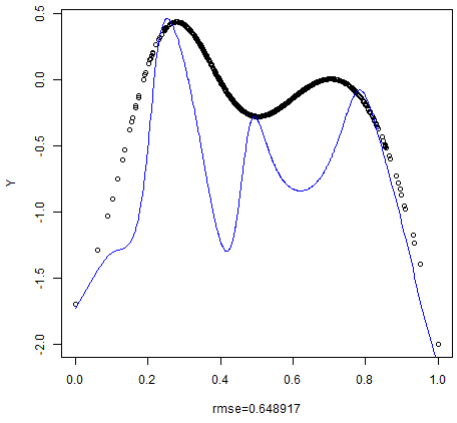
If we select MSE as the objective function:

$$X_{N+1} = \arg\min_{x_{new}} \ MSE(X_N, x_{new})$$

In experiment, one problem occured. That is the next input point of sequential design remains the same. For example, in the following image of 3 iterations, although the minimum value of MSE decreases, the critical point keeps to be 0.95.



As a result, most points in sample path will be about 0.94. And the fitted curve is unacceptable.



4. R codes

4.1 load libraries

```
set.seed(128)

library("hetGP")
library("MASS")
library("Metrics")
library("DEoptim")
```

4.2 generate train and test data

```
# noise variance function
noise.var <- function(x) {
    return(1 / 3 * (exp(sin(2 * pi * sum(x)))))
}

# real regression function
generating_func <- function(x) {
    return(f1d2(sum(x)))
}

# observations(x) = y(x) + noise
observations <- function(x) {
    generating_func(x) + rnorm(1, sd = noise.var(x))
}

generate_data <- function(type) {
    if (type == "seq") {
        X_input <- seq(0, 1, length = N)
        X_test <- seq(0, 1, length = 1000)
    }
    if (type == "unif") {
        X_input <- sort(runif(N, 0, 1))
        X_test <- sort(runif(1000, 0, 1))
    }
    if (type == "norm") {
        X_input <- sort(rnorm(N))
```

```
        X_input <- (X_input - min(X_input)) / (max(X_input) - min(X_input))
        X_test <- sort(rnorm(1000))
        X_test <- (X_test - min(X_test)) / (max(X_test) - min(X_test))
    }

    X_input <- matrix(X_input, nrow = N)
    X_input <- rbind(X_input, X_input)

    X_test <- matrix(X_test, nrow = 1000)

    return(list(X_input = X_input, X_test = X_test))
}
```

4.3 training and testing

```
train <- function(X, step, h, obj) {
    Y <- apply(X, 1, observations)
    mod <- mleHetGP(
        X = X, Z = Y,
        covtype = cov_type,
        settings = list(checkHom = FALSE))

    for (i in 1:step) {
        if (obj == "IMSPE") {
            opt <- IMSPE_optim(mod, h = h)
        }
        if (obj == "EI") {
            opt <- crit_optim(mod, crit = "crit_EI", h = h)
        }
        if (obj == "tMSE") {
            opt <- crit_optim(mod, crit = "crit_tMSE", h = h)
        }
        if (obj == "MSE") {
            opt <- MSE_optim(mod, h = h)
        }
        if (obj == "random") {
            opt <- random_optim(mod, h = h)
        }
        Xnew <- matrix(opt$par, nrow = 1)
        X <- c(X, Xnew)
```

```
        Ynew <- apply(Xnew, 1, observations)
        Y <- c(Y, Ynew)
        mod <- update(mod, Xnew = Xnew, Znew = Ynew, ginit = mod$g * 1.01)
        if (i %% 25 == 0) {
            mod2 <- mleHetGP(
                X = list(X0 = mod$X0, Z0 = mod$Z0, mult = mod$mult),
                Z = mod$Z,
                covtype = cov_type,
                settings = list(checkHom = FALSE),
            )
            if (mod2$ll > mod$ll) mod <- mod2
        }
    }
    return(mod)
}

test <- function(x_test, fit, color) {
    p <- predict(fit, x_test)
    print(rmse(p$mean, apply(x_test, 1, generating_func)))
    lines(x_test, p$mean, col = color)
}
```

Reference

M. Binois, J. Huang, R. B. Gramacy, M. Ludkovski (2019), Replication or exploration? Sequential design for stochastic simulation experiments, Technometrics, 61(1), 7-23.

Picheny V, Ginsbourger D, Roustant O, Haftka RT, Kim NH (2010). "Adaptive Designs of Experiments for Accurate Approximation of a Target Region." Journal of Mechanical Design, 132(7), 071008. doi:10.1115/1.4001873.

Binois, M. and Gramacy, R. B. (2017). hetGP: Heteroskedastic Gaussian Process Modeling and Design under Replication. R package version 1.0.0.