Introduction to Machine Learning - Lab 6

Gustav Sternelöv

Thursday, November 26, 2015

Assignment 1

(a)

The implementation of a function that simulates from the posterior distribution *f(x)* by using the squared exponential kernel is done in two steps. In the first step a function that computes the squared exponential kernel is created. The formula for the squared exponential kernel can be seen below:

$$K(x,x') = \sigma_f^2 \times exp(-0.5 \times (\frac{x-x'}{\iota})^2)$$

The second step is to build the function *PosteriorGP*. The aim with this function is to calculate the posterior mean and variance of f over a grid of x-values. The two formulas used for calculating this are presented below:

$$\bar{f_*} = K(x_*,x)[K(x,x)+\sigma^2 l]^{-1}y$$
$$cov(\bar{f_*}) = K(x_*,x_*) - K(x_*,x)[K(x,x)+\sigma^2 l]^{-1}K(x,x_*)$$

The code that has been used to implement the functions can be seen in the appendix *R-code* at the end of the report.

The prior mean of f is assumed to be zero for all x, which gives the following prior distribution for *f(x)*:

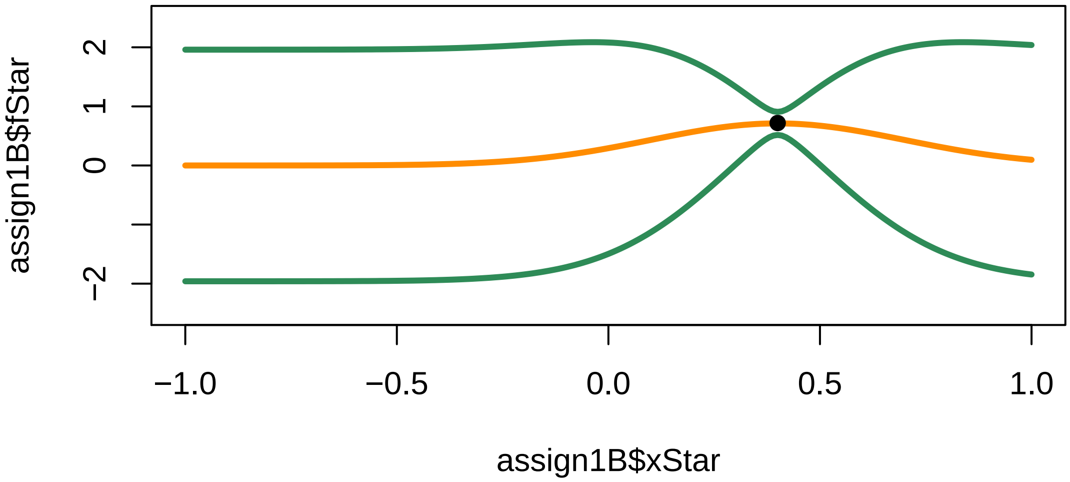
$$f(x) \sim GP(0, K(x,x'))$$

Then, the posterior guassian distribution looks as following:

$$f_* \mid x, y, x_* \sim N(\bar{f_*}, cov(\bar{f_*})$$

(b)

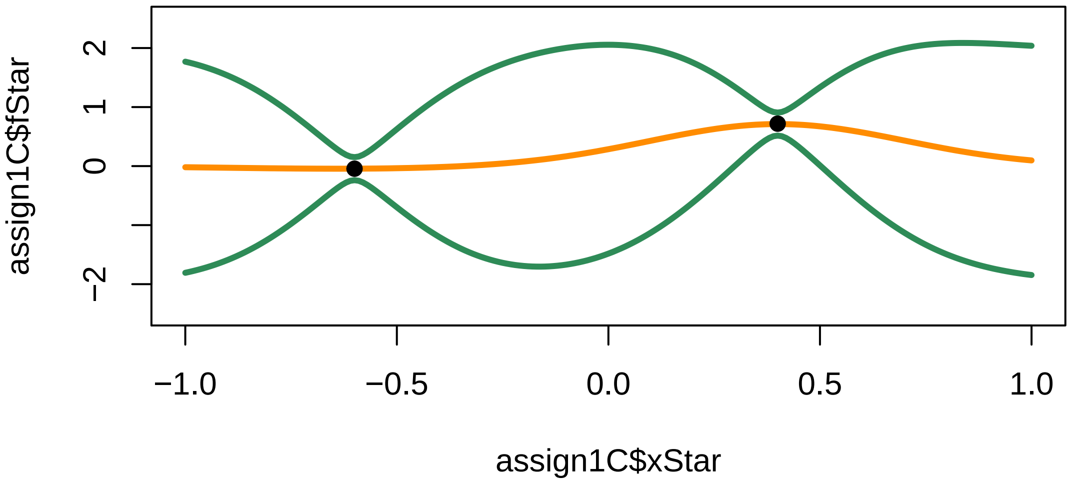
Since the noise standard deviation is assumed to be known the parameter σ_n is set to 0.1. The prior hyperparameter σ_f is set to 1 and the second prior hyperparameter ι is set to 0.3. Furthermore the prior is updated with one observation, *(x,y)*=(0.4, 0.719). A plot over the posterior mean of f over the interval $x \in$ [-1,1] with 95 % probability bands for f can be seen below.



The black dot is the observed value and it can be seen that the probability bands are more narrow around this value.

(c)

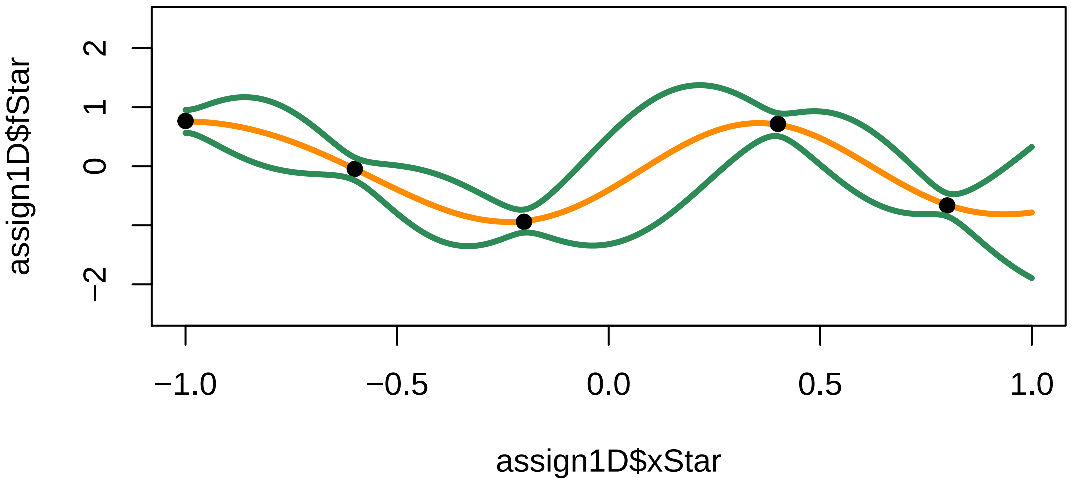
The posterior from *b)* is updated with another observation, *(x,y)*=(-0.6, -0.044).



Again it can be seen that the probability bands are more narrow around the observed values, and that they are quite wide for the other values.

(d)

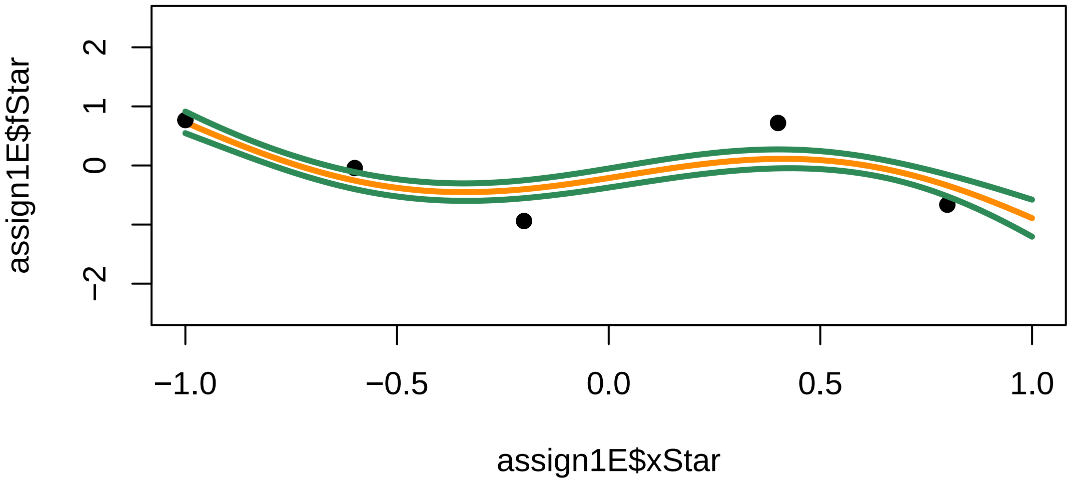
In *d)* the number of observations rises to five, resulting in the following plot over the posterior mean of f and its 95 % probability intervals.



Compared to the plots in *b)* and *c)*, the curve for the posterior mean of f is less straight/ more curvaceous than before. The probability bands has also changed and are thanks to the rise from two to five observed values more narrow, but also quite wiggly.

(e)

The hyperparameter ι is now set to 1. The other parameters are unchanged and the same observations as in *d)* are used.

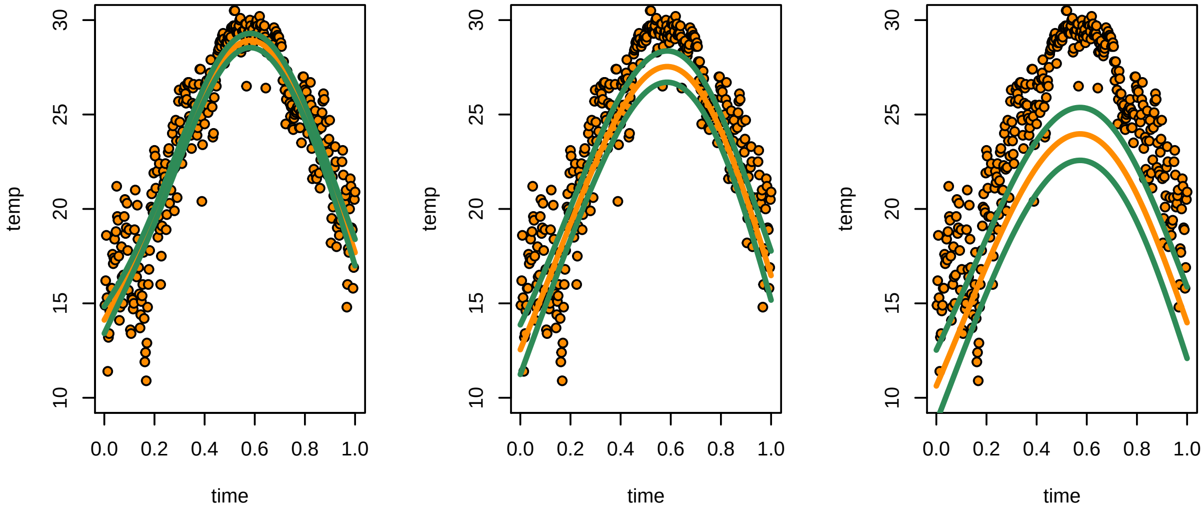


Compared to the plot in *d)*, the probability bands obtained with ι set to 1 looks much smoother and lies much closer to the curve for the posterior mean of f. Another change is that curve for the posterior mean of f no longer goes through all of the observed values. Instead the fitted curve appears to be an average between the observed values that lies closest to each other.

Assignment 2

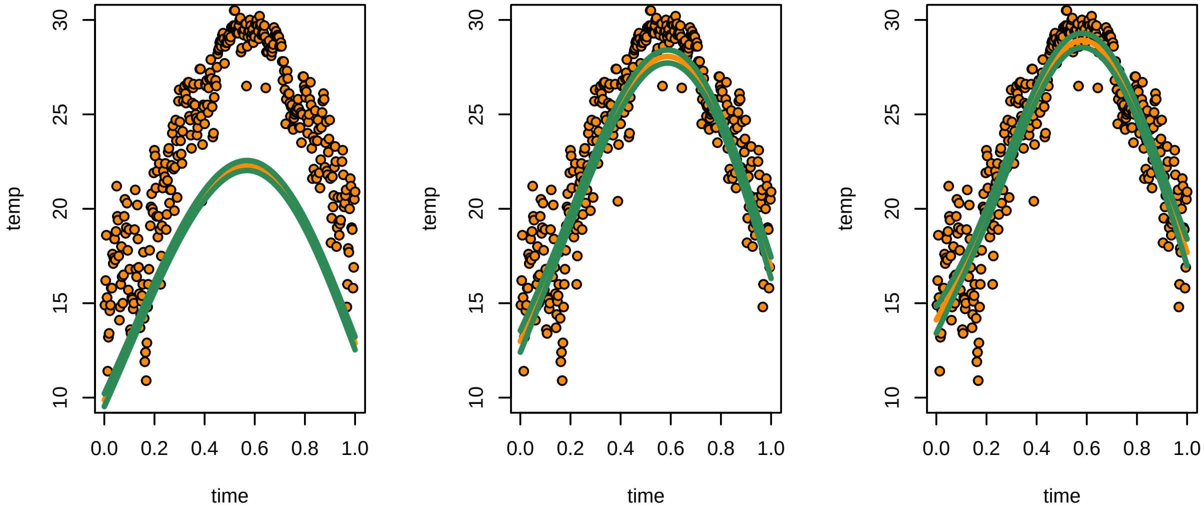
The implemented functions in assignment 1 are now tested on the data set *JapanTemp*. This data set contains information about the daily temperatures during a year for some place in Japan. What that is mainly investigated in this assignment is the effect on the posterior for different values of the parameters σ_n, σ_f, and ι.

First the effect of changing the value for σ_n is investigated. The tested values for σ_n are 2, 5 and 10. The prior hyperparameters are fixed with σ_f set to 1.5 and ι set to 0.3.



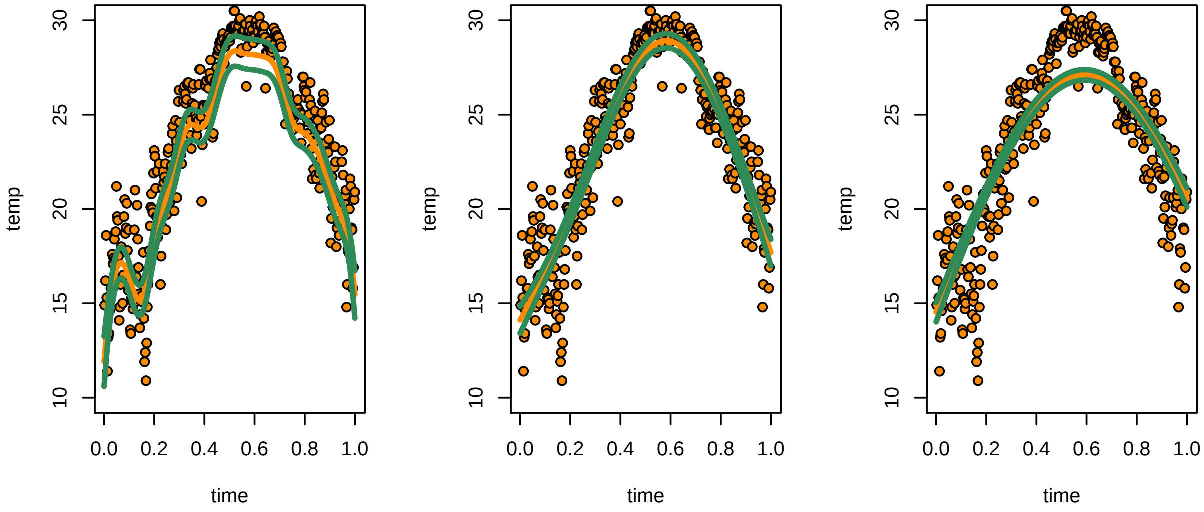
It can be seen that better fits are given for lower values of σ_n. For the highest value tested, σ_n=10, the curve is to smooth and the probability bands are quite wide. For lower values the curve becomes more flexible and the probability bands more narrow. A low value for the noise standard deviation, in this case somewhere around 2, therefore seem to give better results both in terms of an improved fit and narrower probability bands.

The effect of changing the value for σ_f is examined by setting the parameter equal to 0.25, 0.75 or 1.5 whilst the other parameters are fixed (σ_n=2 and ι=0.3).



The effect of changing the value of σ_f looks quite similar to the effect of changing the value of σ_n with the difference that the probability bands remains practically unchanged. Also higher values of σ_f seem to give better fits with more flexible curves, compared to σ_n where lower values resulted in better fits.

Finally, the effect on the posterior when changing the value for the prior hyperparameter ι is examined. The tested values for ι are 0.05, 0.30 and 0.75 while the other parameters are fixed (σ_n=2 and σ_f=1.5).



For a ι value of 0.05 the obtained curve is too flexible, it follows the observed values too well. For a little higher value, 0.30, the curve is more smooth and seem to be a quite good fit. When a even higher value is tested, 0.75, the curve instead becomes too inflexible.

Appendix

R-code

```
SqExpKernel <- function(x1, x2, hyperParam){
  K <- matrix(nrow=length(x1), ncol=length(x2))
  for (i in 1:length(x2)){
    K[, i] <- hyperParam[1]^2 * exp(-0.5 *( (x1-x2[i])/hyperParam[2]) ^2)
  }
  return(K)
}
PosteriorGP <- function(x, y, xStar, hyperParam, sigmaNoise){
  # Calculates f star bar
  fStar <- SqExpKernel(xStar, x, hyperParam) %*% solve(SqExpKernel(x, x, hyperParam)+
                                        sigmaNoise^2*diag(length(x))) %*% y
  # Calculates cov f star
  cov_fStar <- SqExpKernel(xStar, xStar, hyperParam) - SqExpKernel(xStar, x, hyperParam)%*%
    solve(SqExpKernel(x, x, hyperParam)+sigmaNoise^2*diag(length(x))) %*%
    SqExpKernel(x, xStar, hyperParam)

  # Store all values in a list
  val_list <- list(fStar=fStar, cov_fStar=cov_fStar, xStar=xStar)

  return(val_list)
}
assign1B <- PosteriorGP(x=0.4, y=0.719, xStar=seq(-1,1, 0.01), hyperParam=c(1, 0.3),
            sigmaNoise=0.1)
```

```
Upper1B <- assign1B$fStar + 1.96 * sqrt(diag(assign1B$cov_fStar))
Lower1B <- assign1B$fStar - 1.96 * sqrt(diag(assign1B$cov_fStar))

plot(y=assign1B$fStar, assign1B$xStar, ylim=c(-2.5,2.5), type="l", lwd=3, col="darkorange")
points(x=0.4, y=0.719, pch=21, bg="black")
lines(y=Upper1B, assign1B$xStar, col="seagreen", lwd=3)
lines(y=Lower1B, assign1B$xStar, col="seagreen", lwd=3)
assign1C <- PosteriorGP(x=c(0.4, -0.6), y=c(0.719, -0.044), xStar=seq(-1,1, 0.01), hyperParam=c(1, 0.3)
                        sigmaNoise=0.1)
Upper1C <- assign1C$fStar + 1.96 * sqrt(diag(assign1C$cov_fStar))
Lower1C <- assign1C$fStar - 1.96 * sqrt(diag(assign1C$cov_fStar))

plot(y=assign1C$fStar, assign1C$xStar, ylim=c(-2.5,2.5), type="l", lwd=3, col="darkorange")
points(x=c(0.4, -0.6), y=c(0.719, -0.044), pch=21, bg="black")
lines(y=Upper1C, assign1C$xStar, col="seagreen", lwd=3)
lines(y=Lower1C, assign1C$xStar, col="seagreen", lwd=3)
assign1D <- PosteriorGP(x=c(0.8, 0.4, -0.2, -0.6, -1), y=c(-0.664, 0.719, -0.94, -0.044, 0.768), xStar=
                        hyperParam=c(1, 0.3),sigmaNoise=0.1)
Upper1D <- assign1D$fStar + 1.96 * sqrt(diag(assign1D$cov_fStar))
Lower1D <- assign1D$fStar - 1.96 * sqrt(diag(assign1D$cov_fStar))

plot(y=assign1D$fStar, assign1D$xStar, ylim=c(-2.5,2.5),type="l", lwd=3, col="darkorange")
points(x=c(0.8, 0.4, -0.2, -0.6, -1), y=c(-0.664, 0.719, -0.94, -0.044, 0.768), pch=21, bg="black")
lines(y=Upper1D, assign1D$xStar, col="seagreen", lwd=3)
lines(y=Lower1D, assign1D$xStar, col="seagreen", lwd=3)
assign1E <- PosteriorGP(x=c(0.8, 0.4, -0.2, -0.6, -1), y=c(-0.664, 0.719, -0.94, -0.044, 0.768), xStar=
                        hyperParam=c(1, 1),sigmaNoise=0.1)
Upper1E <- assign1E$fStar + 1.96 * sqrt(diag(assign1E$cov_fStar))
Lower1E <- assign1E$fStar - 1.96 * sqrt(diag(assign1E$cov_fStar))

plot(y=assign1E$fStar, assign1E$xStar, ylim=c(-2.5,2.5), type="l", lwd=3, col="darkorange")
points(x=c(0.8, 0.4, -0.2, -0.6, -1), y=c(-0.664, 0.719, -0.94, -0.044, 0.768), pch=21, bg="black")
lines(y=Upper1E, assign1E$xStar, col="seagreen", lwd=3)
lines(y=Lower1E, assign1E$xStar, col="seagreen", lwd=3)
JapanTemp <- read.delim("C:/Users/Gustav/Documents/Machine-Learning/Lab 6/JapanTemp.dat", sep="", heade
par(mfrow=c(1,3))
SigmaVal <- c(2, 5, 10)
for(i in SigmaVal){
  Assign2 <- PosteriorGP(x=JapanTemp$time, y=JapanTemp$temp, xStar=seq(0,1, 0.01),
                       hyperParam=c(1.5, 0.3),sigmaNoise=i)
Upper2 <- Assign2$fStar + 1.96 * sqrt(diag(Assign2$cov_fStar))
Lower2 <- Assign2$fStar - 1.96 * sqrt(diag(Assign2$cov_fStar))

plot(JapanTemp, ylim=c(10,30), pch=21, bg="darkorange")
lines(y=Assign2$fStar, Assign2$xStar, type="l", lwd=3, col="darkorange")
lines(y=Upper2, Assign2$xStar, col="seagreen", lwd=3)
lines(y=Lower2, Assign2$xStar, col="seagreen", lwd=3)
}
par(mfrow=c(1,1))
par(mfrow=c(1,3))
Sigma_fVal <- c(0.25, 0.75, 1.5)
for(i in Sigma_fVal){
Assign2 <- PosteriorGP(x=JapanTemp$time, y=JapanTemp$temp, xStar=seq(0,1, 0.01),
                       hyperParam=c(i, 0.3),sigmaNoise=2)
Upper2 <- Assign2$fStar + 1.96 * sqrt(diag(Assign2$cov_fStar))
Lower2 <- Assign2$fStar - 1.96 * sqrt(diag(Assign2$cov_fStar))

plot(JapanTemp, ylim=c(10,30), pch=21, bg="darkorange")
lines(y=Assign2$fStar, Assign2$xStar, type="l", lwd=3, col="darkorange")
lines(y=Upper2, Assign2$xStar, col="seagreen", lwd=3)
lines(y=Lower2, Assign2$xStar, col="seagreen", lwd=3)
}
par(mfrow=c(1,1))
par(mfrow=c(1,3))
iota <- c(0.05, 0.3, 0.75)
for(i in iota){
Assign2 <- PosteriorGP(x=JapanTemp$time, y=JapanTemp$temp, xStar=seq(0,1, 0.01),
                       hyperParam=c(1.5, i),sigmaNoise=2)
Upper2 <- Assign2$fStar + 1.96 * sqrt(diag(Assign2$cov_fStar))
Lower2 <- Assign2$fStar - 1.96 * sqrt(diag(Assign2$cov_fStar))

plot(JapanTemp, ylim=c(10,30), pch=21, bg="darkorange")
lines(y=Assign2$fStar, Assign2$xStar, type="l", lwd=3, col="darkorange")
lines(y=Upper2, Assign2$xStar, col="seagreen", lwd=3)
lines(y=Lower2, Assign2$xStar, col="seagreen", lwd=3)
}
par(mfrow=c(1,1))
##
```