# Lab 4 - Advanced Machine Learning

Simon Jorstedt

2024-10-13

```
# SETUP
```

## Problem 2.1 - Algorithm implementation

```
# Covariance function
SquaredExpKernel <- function(x1, x2, sigmaF=3, l=.3){
  n1 <- length(x1)
  n2 <- length(x2)
  K <- matrix(NA,n1,n2)
  for (i in 1:n2){
    K[,i] <- sigmaF^2*exp(-0.5*( (x1-x2[i])/l)^2 )
  }
  return(K)
}
```

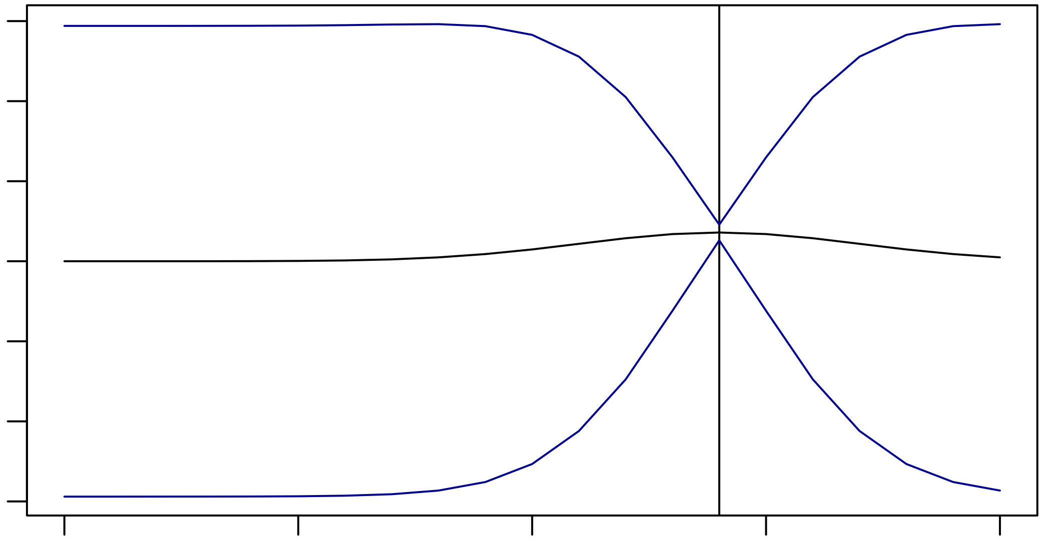
```
posteriorGP <- function(X, y, XStar, sigmaNoise, k){
  # Returns a list of sublists (for each XStar value), each containing f_mean, Variance, loglikelihood.
  K <- k(X, X)
  L <- t(chol(K + sigmaNoise**2 * diag(length(X))))

  alpha <- solve(t(L), solve(L, y))

  #return(alpha)

  # Collect outputs for each XStar value
  f_mean_vec <- c()
  Var_vec <- c()
  #loglikelihood? no?

  for (i in 1:length(XStar)){
    k_star <- matrix(k(XStar[[i]], X),
                     nrow=length(X))

    f_star <- t(k_star) %*% alpha

    v <- solve(L, k_star)

    V_of_f_star <- k(XStar[[i]], XStar[[i]]) - t(v) %*% v
```

```r
    # Save mean and variance
    f_mean_vec <- c(f_mean_vec, f_star)
    Var_vec <- c(Var_vec, V_of_f_star)
    #output[[i]] <- list(f_star, V_of_f_star, log_p_y_X)
  }

  log_p_y_X <- -1/2 * t(y) %*% alpha - sum(log(diag(L))) - length(X)/2*log(2*pi)

  return(list(means = f_mean_vec, variances = Var_vec, loglik=log_p_y_X))
}
```

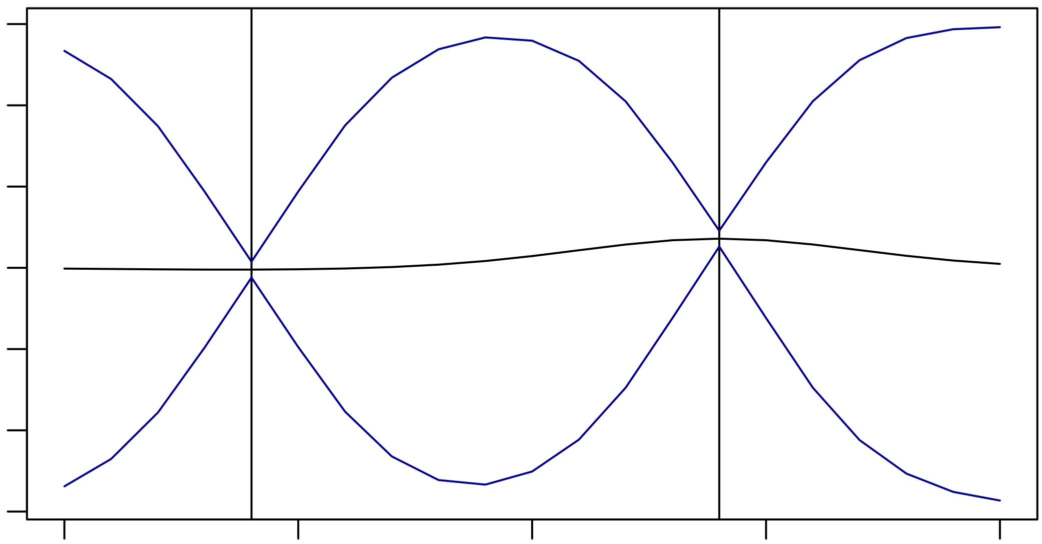
### Subproblem 2.1.2

Hey Simon, here is how you continue: You have just implemented the new posteriorGP function that can run the Algorithm for multiple XStar values. Now just call the function easy peasy makkie kakkie once for each plot they want you to make. Hvae a ncie dya!

```r
# Plot figure 1
x_star_values <- seq(-1, 1, 0.1)
result_1 <- posteriorGP(X=c(0.4),
                        y=c(0.719),
                        XStar=x_star_values,
                        sigmaNoise=0.1,
                        k=SquaredExpKernel)

lower_cfi_curve <- qnorm(p=0.025, mean=result_1$means, sd = result_1$variances**(1/2))
upper_cfi_curve <- qnorm(p=0.975, mean=result_1$means, sd = result_1$variances**(1/2))

plot(x_star_values,
     result_1$means,
     type="l",
     ylim=range(lower_cfi_curve,
                upper_cfi_curve),
     main="Fig 1: Observation (x,y) = (0.4, 0.719)")
abline(v=c(0.4))
points(x_star_values, lower_cfi_curve, type="l", col="darkblue")
points(x_star_values, upper_cfi_curve, type="l", col="darkblue")
```



```r
# Plot figure 2
x_star_values <- seq(-1, 1, 0.1)
result_2 <- posteriorGP(X=c(0.4, -0.6),
                        y=c(0.719, -0.044),
                        XStar=x_star_values,
                        sigmaNoise=0.1,
                        k=SquaredExpKernel)

lower_cfi_curve_2 <- qnorm(p=0.025, mean=result_2$means, sd = result_2$variances**(1/2))
upper_cfi_curve_2 <- qnorm(p=0.975, mean=result_2$means, sd = result_2$variances**(1/2))

plot(x_star_values,
     result_2$means,
     type="l",
     ylim=range(lower_cfi_curve_2,
                upper_cfi_curve_2),
     main="Fig 2: Two observations")
abline(v=c(0.4, -0.6))
points(x_star_values, lower_cfi_curve_2, type="l", col="darkblue")
points(x_star_values, upper_cfi_curve_2, type="l", col="darkblue")
```



```r
posteriorGP(X=c(-1, -0.6, -0.2, 0.4, 0.8),
            y=c(0.768, -0.044, -0.940, 0.719, -0.664),
            XStar=x_star_values,
            sigmaNoise=0.1,
            k=SquaredExpKernel)
```

```
## $means
##  [1]  0.76717218  0.70923382  0.54455995  0.28428771 -0.04415229 -0.39640255
##  [7] -0.71022641 -0.91244683 -0.93866801 -0.76020504 -0.40475772  0.04130301
## [13]  0.45312191  0.70549999  0.71746278  0.48603039  0.09129590 -0.33465328
## [19] -0.66263691 -0.81694733 -0.79549390
##
## $variances
##  [1] 0.009986251 0.391754389 0.679946820 0.322158496 0.009983412 0.317834726
##  [7] 0.658839421 0.369217269 0.009985895 0.585989092 1.902591335 2.619984414
## [13] 1.935681294 0.608838011 0.009986295 0.415202500 0.799607789 0.432742280
## [19] 0.009986597 0.741723734 2.822016178
##
## $loglik
##           [,1]
## [1,] -9.980549
```

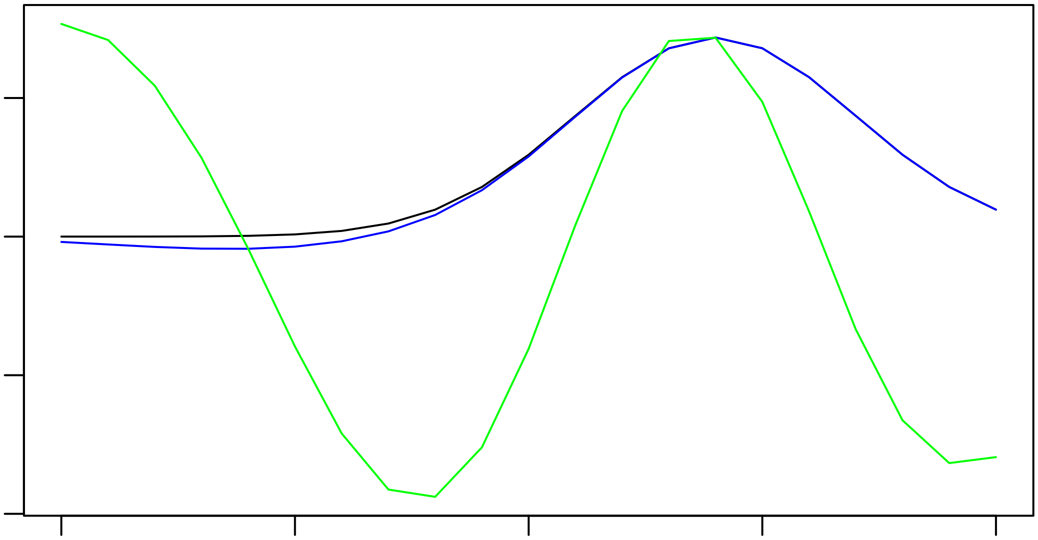
```r
#posteriorGP(X=c(-1, -0.6, -0.2, 0.4, 0.8), y=c(0.768, -0.044, -0.940, 0.719, -0.664), XStar=c(1,2,3), 
```

```r
# Plot after one observation (0.4, 0.719)
post_means_1 <- c()
x_star_values <- seq(-1,1,by=0.1)
for (xstar in x_star_values){
  res <- posteriorGP(X=0.4, y=0.719, XStar=xstar, sigmaNoise=0.1, k=SquaredExpKernel)

  post_means_1 <- c(post_means_1, res[[1]])
}


# Plot after an additional observation (-0.6, -0.044)
```

```r
post_means_2 <- c()
for (xstar in x_star_values){
  res <- posteriorGP(X=matrix(c(0.4, -0.6),nrow=2), y=c(0.719, -0.044), XStar=xstar, sigmaNoise=0.1, k=S

  post_means_2 <- c(post_means_2, res[[1]])
}


# PLOT!
#plot(x_star_values, post_means_1, type="l", ylim=c(min(post_means_1, post_means_2), max(post_means_1, p
#points(x_star_values, post_means_2, type="l", col="blue")
```

## Subproblem 2.2.4

```r
post_means_3 <- c()
x_star_values <- seq(-1,1,by=0.1)
for (xstar in x_star_values){
  res <- posteriorGP(X=c(-1, -0.6, -0.2, 0.4, 0.8), y=c(0.768, -0.044, -0.940, 0.719, -0.664), XStar=xst

  post_means_3 <- c(post_means_3, res[[1]])
}


plot(x_star_values, post_means_1, type="l", ylim=c(min(post_means_1, post_means_2, post_means_3), max(po
points(x_star_values, post_means_2, type="l", col="blue")
points(x_star_values, post_means_3, type="l", col="green")
```



```r
A <- SquaredExpKernel(1:5, 1:5, sigmaF = 0.1) #matrix(c(1,0,1,0,1,0,1,0,1), nrow=3)
#A
L <- chol(A)
t(L) %*% L == A
```

```
##      [,1]  [,2]  [,3] [,4] [,5]
```

```
## [1,] TRUE  TRUE  TRUE TRUE TRUE
## [2,] TRUE FALSE  TRUE TRUE TRUE
## [3,] TRUE  TRUE FALSE TRUE TRUE
## [4,] TRUE  TRUE  TRUE TRUE TRUE
## [5,] TRUE  TRUE  TRUE TRUE TRUE
```

```r
#matrix(c(1,0,0,2), nrow=2) %*% t(matrix(c(1,0,0,2), nrow=2))
```

## Problem 2.2 - GP Regression with Kernlab

### Subproblem 2.2.5 - Periodic kernel

## Problem 2.3 - GP Classification iwth Kernlab