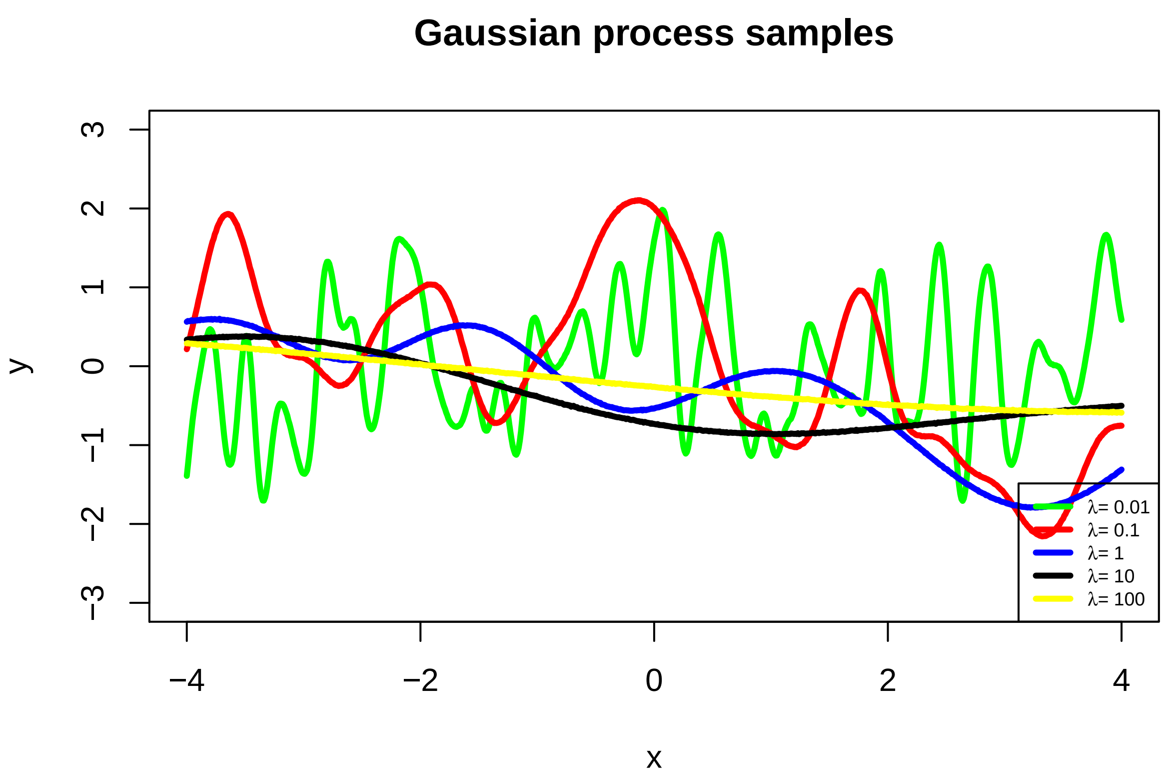
hw4

Shweta Singhal

April 23, 2016

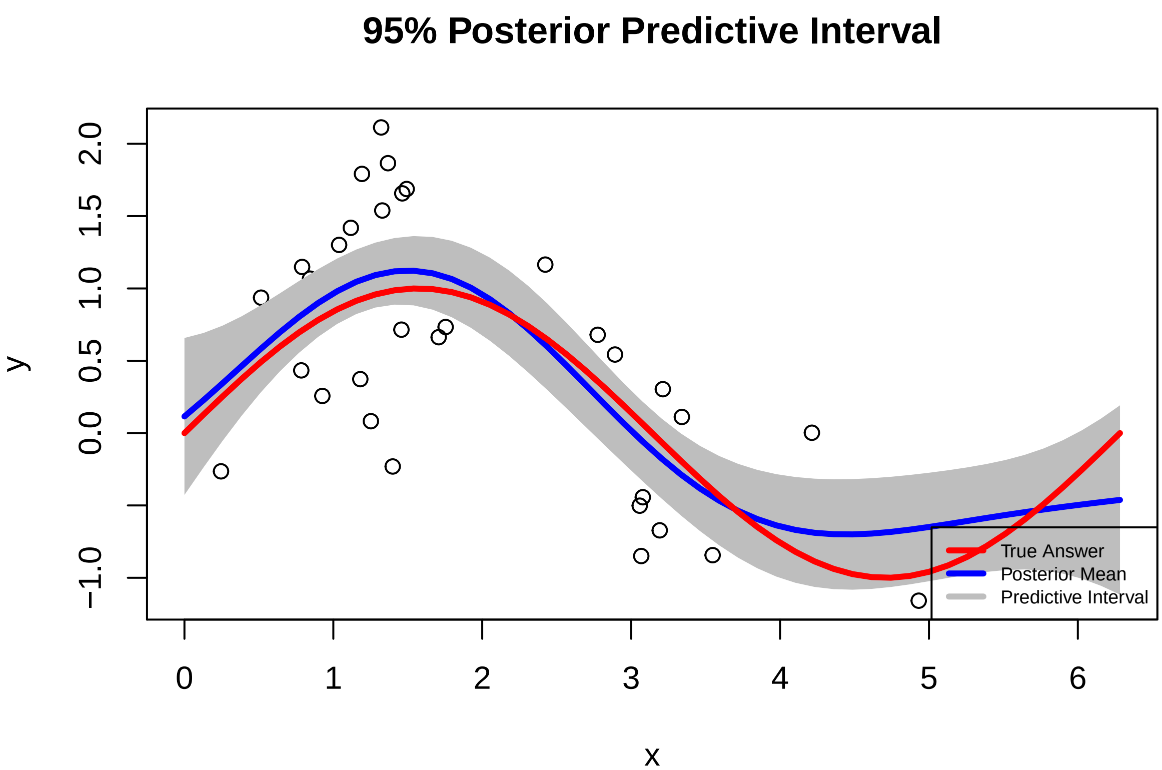
```
cov_Calculate = function(x1, x2, lambda, eps){
  res = matrix(0.0, length(x1), length(x2));
  tmp = outer(x1, x2, function(x1, x2) x2-x1)
  res = exp(-0.5*tmp*tmp/lambda) + eps*1.0*(abs(tmp)==0);
  return(res)
}

drawsample = function(x,lambda, eps){
  num = length(x);
  Sigma = cov_Calculate(x, x, lambda, eps);
  lt = t(chol(Sigma));
  u = rnorm(1:num, 0, 1);
  res = lt %*% u;
  return(res)
}
```



Ans.1)
From the plot, it is clear that the value of λ controls the smoothness of the graph,i.e, width of the kernel. With smaller value of λ the width of the Gaussian is small and it seems like it tries to overfit each and every data point, and as the λ increases, the width of individual Gaussian curve increases and it makes the regression curve more and more smoother.

Ans.2) Here $\lambda = 2.0$ with $\sigma = 0.5$ works great.

```r
num = 50
sigma = 0.5
lambda = 2
cov.eps = 1e-5

x = runif(num, min = 0, max = 2*pi)
x.test = seq(0, 2*pi, length.out=num)
eps = rnorm(1:num, 0, sigma)
y = sin(x) + eps

k_xx = cov_Calculate(x, x, lambda, eps = cov.eps)
k_xx = k_xx + sigma^2*diag(num)
k_xx.inv = chol2inv(chol(k_xx))
k_xxtest = cov_Calculate(x, x.test, lambda, cov.eps)
k_xtestx = t(k_xxtest)
k_xtestxtest = cov_Calculate(x.test, x.test, lambda, cov.eps)

post.mu <- k_xtestx %*% k_xx.inv %*% y
post.covariance <- k_xtestxtest - k_xtestx %*% k_xx.inv %*% k_xxtest
lt = t(chol(post.covariance))
u = rnorm(1:num, 0, 1)
y.posterior = post.mu + lt %*% u

post.covariance <- diag(post.covariance)

t = seq(0,2*pi,length.out=num)
up = post.mu + 1.96 * sqrt(post.covariance)
down = post.mu - 1.96 * sqrt(post.covariance)
t.rev = t[length(t):1]
down = down[length(down):1]

plot(x, y, xlim=c(0,2*pi), ylim=range(y, down, up), main="95% Posterior Predictive Interval")
polygon(x = c(t, t.rev), y=c(up, down), col="grey", border=NA)
lines(t, post.mu, col = 'blue', lwd=3)
#lines(t, y.post, col = 'green', lwd=3)
lines(t, sin(t), col='red', lwd=3)
legend("bottomright", c("True Answer", "Posterior Mean " ,"Predictive Interval"), col = c("red","blue",
```

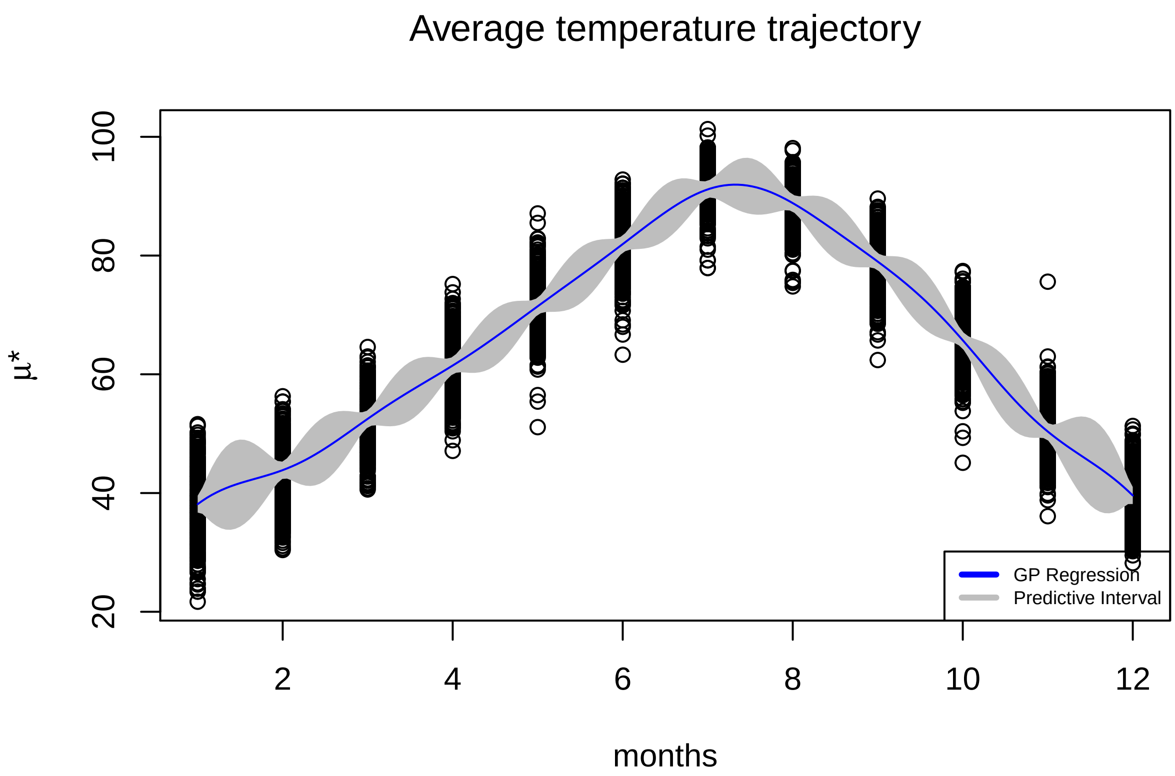


Ans.3.a) To model the data using Gaussian process regression, I have taken 200 discretized samples between [1,12] (months) and computed the posterior mean trajectory for average temperature. Parameters: $\lambda = 1.0$ and $\sigma = 0.5$

Here are the plots:

```
data = read.csv("/home/shweta/Documents/Probabilistic Modelling/hw4/SaltLakeTemperatures.csv",nrows=-1)
data = data[!data$AVEMAX == -9999,]

data["YEAR"] = 0
data["MONTH"] = 0

dummy = lapply(data$DATE, function(x) substr(x,1,4))
for(i in 1:length(dummy)){
  data$YEAR[i] = as.numeric(x=dummy[i])
}
dummy = lapply(data$DATE,function(x) substr(x,5,6))
for(i in 1:length(dummy)){
  data$MONTH[i] = as.numeric(x=dummy[i])
}

x = data$MONTH
num = length(x)
sigma = 0.5
lambda = 1.0
cov.eps = 1e-5

###a##
y = data$AVEMAX
x.test = seq(1, 12, length.out=200)
k_xx = cov_Calculate(x, x, lambda, cov.eps)
```

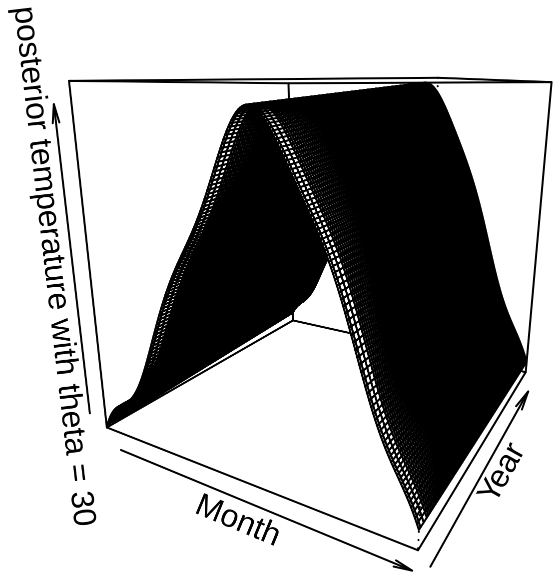
```
k_xx = k_xx + sigma^2*diag(num)
k_xx.inv = chol2inv(chol(k_xx))
k_xx1 = cov_Calculate(x, x.test, lambda, cov.eps)
k_x1x = t(k_xx1)
k_x1x1 = cov_Calculate(x.test, x.test, lambda,cov.eps)

post.mu <- k_x1x %*% (k_xx.inv %*% y)
post.covariance <- k_x1x1 - (k_x1x %*% (k_xx.inv %*% k_xx1))
post.covariance <- diag(post.covariance)*1000.0

t = seq(1, 12, length.out=200)
up = post.mu + 1.96 * sqrt(post.covariance)
down = post.mu - 1.96 * sqrt(post.covariance)
t.rev = t[length(t):1]
down = down[length(down):1]

plot(x, y, xlim = c(1, 12), ylim = range(y, down, up), main = expression("Average temperature trajector
polygon(x = c(t, t.rev), y = c(up, down), col = "grey", border = NA)
lines(t, post.mu, col = 'blue', lwd = 1)
legend("bottomright", c("GP Regression", "Predictive Interval"), col = c("blue", "grey"), lwd = c(3,3,1
```



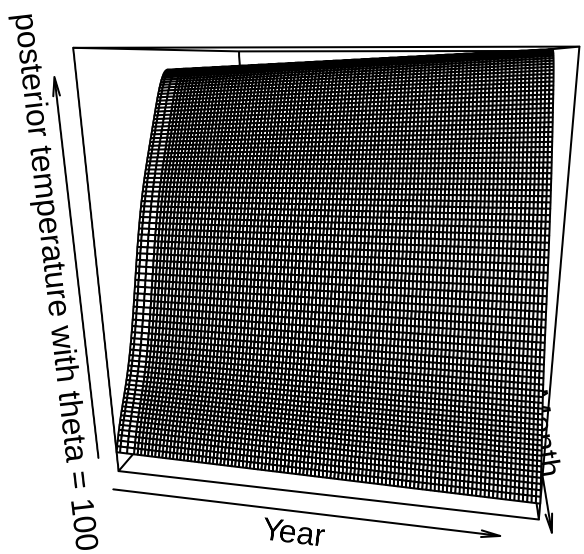
Ans.3.b)

```
#data <- data[order(data$YEAR),]
#data <- subset(data, data$YEAR >= 1906 & data$YEAR <= 1956)
#data <- subset(data, data$MONTH == 6)
x = data$MONTH
num = length(x)
sample_num = 200
```

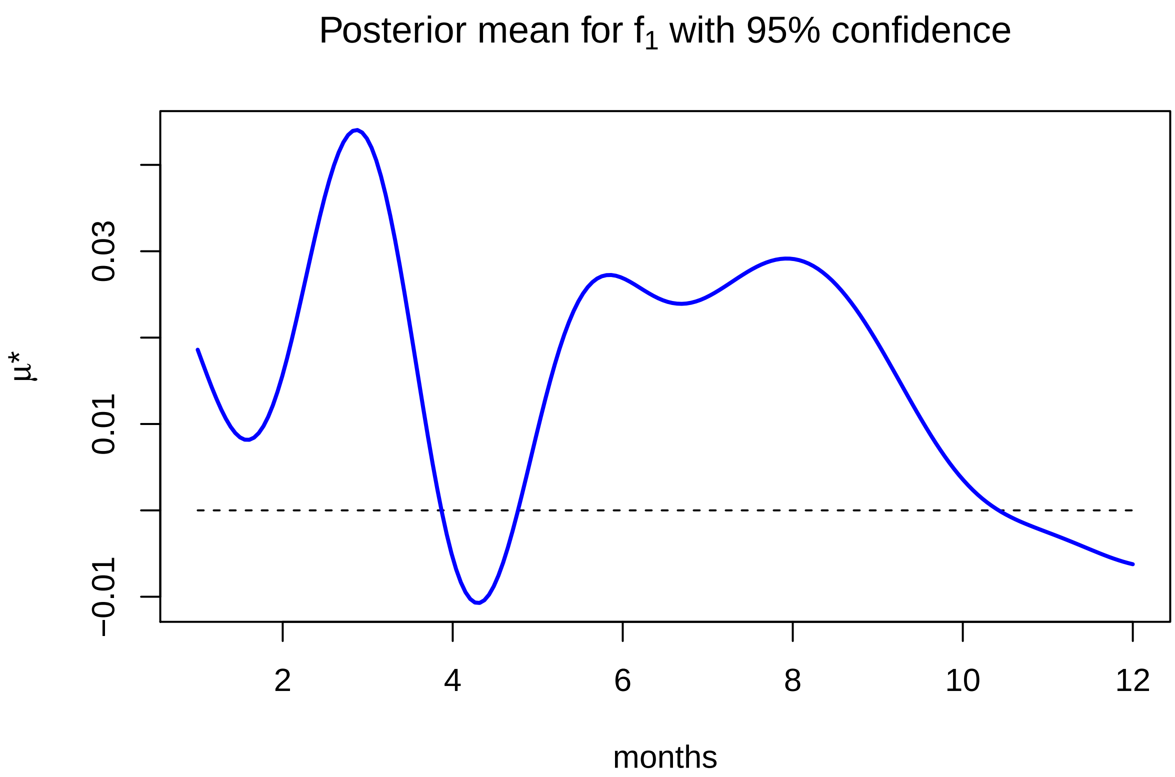
```r
x.test = seq(1, 12, length.out = sample_num)
z = data$YEAR
y = data$AVEMAX
z = z - min(z)
#Covariance between y_i and y_j is depending upon the variance of f0, f1 and epsilon
#where f1 is mutiplied by z, therefore the variance will be multiplied by z^2 using sweep
#and the same when covariance related to y and f1 is calculated.
k_yy = cov_Calculate(x, x, lambda, cov.eps)
a = sweep(x = k_yy, MARGIN=2, STATS=z, FUN='*')
b = sweep(x = a, MARGIN=1, STATS=z, FUN='*')
k_yy = b + k_yy + sigma^2*diag(num)
k_yy.inv = chol2inv(chol(k_yy))

k_yf0 = cov_Calculate(x, x.test, lambda, cov.eps)
k_yf1 = k_yf0
k_yf1 = sweep(x = k_yf1, MARGIN=1, STATS=z, FUN='*')
k_f0y = t(k_yf0)
k_f0f0 = cov_Calculate(x.test, x.test, lambda, cov.eps)
k_f0f1 = matrix(0.0, sample_num, sample_num)
k_f1y = t(k_yf1)
k_f1f0 = matrix(0.0, sample_num, sample_num)
k_f1f1 = k_f0f0

post.mu <- rbind(k_f0y,k_f1y) %*% (k_yy.inv %*% y)
post.covariance <- rbind(cbind(k_f0f0,k_f0f1),cbind(k_f1f0,k_f1f1)) - ( rbind(k_f0y,k_f1y) %*% ( k_yy.in
post.covariance <- (diag(post.covariance))

post.mu.f1 <- post.mu[(sample_num+1L):(length(post.mu)),]
post.cov.f1 <- post.covariance[(sample_num+1L):(length(post.covariance))]
post.mu.f0 <- post.mu[1:sample_num]
post.cov.f0 <- post.covariance[1:sample_num]

# ## Plot 95% confidence interval
t = seq(1,12,length.out=sample_num)
up = post.mu.f1 + 1.96 * sqrt(post.cov.f1)
down = post.mu.f1 - 1.96 * sqrt(post.cov.f1)
t.rev = t[length(t):1]
down = down[length(down):1]

z.unique = sort(unique(z))
persp(x.test, z.unique, post.mu.f0 + outer(post.mu.f1, z.unique, '*'),xlab = "Month", ylab = "Year",
      zlab = "posterior temperature with theta = 30 ",theta = 30)
```



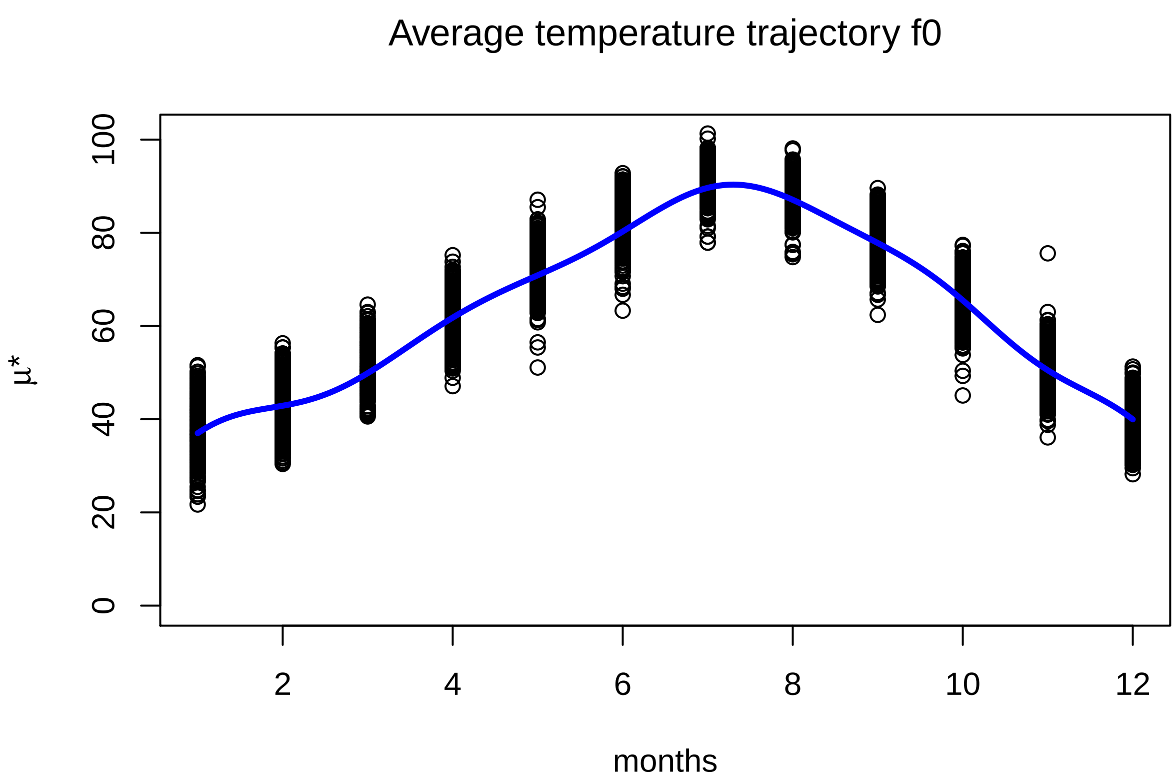

```r
persp(x.test, z.unique, post.mu.f0 + outer(post.mu.f1, z.unique, '*'),xlab = "Month", ylab = "Year",
      zlab = "posterior temperature with theta = 100 ",theta = 100)
```




```r
plot(t, rep(0, sample_num), ylim = range(min(post.mu.f1), max(post.mu.f1)), type = "l", lty = 2, main=e
lines(t, post.mu.f1, col = 'blue', lwd=2)
```



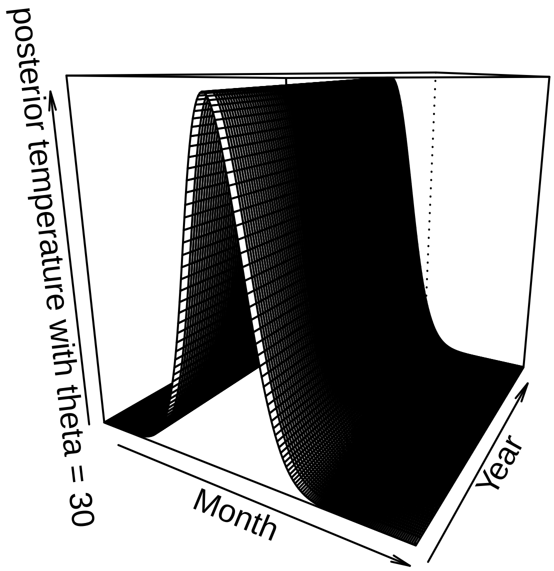

```r
plot(x,y, xlim=c(1,12),ylim=range(y,down,up), main=expression("Average temperature trajectory f0"),
lines(t, post.mu.f0, col = 'blue', lwd=3)
```



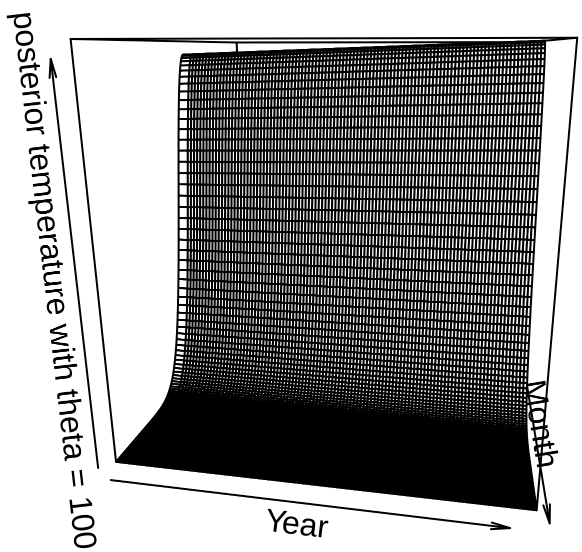
From the 3D plot after rotating the plot with theta = 100, the months are marginalized, and the graph says with time(year) the average temperature of the Salt Lake is increasing. Also, the analysis for average temperature with respect to month is also plotting in the graph as "Average temperature trajectory", which shows that maximum temperature was recorded during June/July.

Below is the same analysis for the month = 6.

```
data <- subset(data, data$MONTH == 6)
x = data$MONTH
num = length(x)
sample_num = 200
x.test = seq(1, 12, length.out = sample_num)
z = data$YEAR
y = data$AVEMAX
z = z - min(z)
#Covariance between y_i and y_j is depending upon the variance of f0, f1 and epsilon
#where f1 is mutiplied by z, therefore the variance will be multiplied by z^2 using sweep
#and the same when covariance related to y and f1 is calculated.
k_yy = cov_Calculate(x, x, lambda, cov.eps)
a = sweep(x = k_yy, MARGIN=2, STATS=z, FUN='*')
b = sweep(x = a, MARGIN=1, STATS=z, FUN='*')
k_yy = b + k_yy + sigma^2*diag(num)
k_yy.inv = chol2inv(chol(k_yy))

k_yf0 = cov_Calculate(x, x.test, lambda, cov.eps)
k_yf1 = k_yf0
k_yf1 = sweep(x = k_yf1, MARGIN=1, STATS=z, FUN='*')
k_f0y = t(k_yf0)
k_f0f0 = cov_Calculate(x.test, x.test, lambda, cov.eps)
k_f0f1 = matrix(0.0, sample_num, sample_num)
k_f1y = t(k_yf1)
k_f1f0 = matrix(0.0, sample_num, sample_num)
k_f1f1 = k_f0f0

post.mu <- rbind(k_f0y,k_f1y) %*% (k_yy.inv %*% y)
post.covariance <- rbind(cbind(k_f0f0,k_f0f1),cbind(k_f1f0,k_f1f1)) - ( rbind(k_f0y,k_f1y) %*% ( k_yy.in
post.covariance <- (diag(post.covariance))

post.mu.f1 <- post.mu[(sample_num+1L):(length(post.mu)),]
post.cov.f1 <- post.covariance[(sample_num+1L):(length(post.covariance))]
post.mu.f0 <- post.mu[1:sample_num]
post.cov.f0 <- post.covariance[1:sample_num]

# ## Plot 95% confidence interval
t = seq(1,12,length.out=sample_num)
up = post.mu.f1 + 1.96 * sqrt(post.cov.f1)
down = post.mu.f1 - 1.96 * sqrt(post.cov.f1)
t.rev = t[length(t):1]
down = down[length(down):1]

z.unique = sort(unique(z))
persp(x.test, z.unique, post.mu.f0 + outer(post.mu.f1, z.unique, '*'),xlab = "Month", ylab = "Year",
      zlab = "posterior temperature with theta = 30 ",theta = 30)
```




```r
persp(x.test, z.unique, post.mu.f0 + outer(post.mu.f1, z.unique, '*'),xlab = "Month", ylab = "Year",
      zlab = "posterior temperature with theta = 100 ",theta = 100)
```



This analysis of month = June, shows that the temperature from year 1906 to year 2015 is increased. Therefore, the whole conclusion is that the SLC is getting hotter from 1906-2015 dataset. Note: Here I haven't played with my λ and σ parameter much.