STAT547 Homework 4

Xingche Guo

10/27/2019

Problem 1

The NW estimator has the form: $\hat{\mu}(x_0) = \frac{\sum_{i=1}^n K_h(X_i - x_0)Y_i}{\sum_{i=1}^n K_h(X_i - x_0)}$, therefore:

$$\text{Bias}(\hat{\mu}(x_0)|\mathbf{X}_n) = \frac{\sum_{i=1}^n K_h(X_i - x_0)\mu(X_i)}{\sum_{i=1}^n K_h(X_i - x_0)} - \mu(x_0); \qquad \text{Var}(\hat{\mu}(x_0)|\mathbf{X}_n) = \frac{\sum_{i=1}^n K_h^2(X_i - x_0)\sigma^2}{(\sum_{i=1}^n K_h(X_i - x_0))^2}$$

Define $S_n = \sum_{i=1}^n K_h(X_i - x_0)$, $T_n = \sum_{i=1}^n K_h^2(X_i - x_0)$, $\mu_{j,c} = \int_{-c}^1 u^j K(u)du$, $\nu_{j,c} = \int_{-c}^1 u^j K^2(u)du$, then:

$$\text{E}(S_n) = \frac{n}{h}\int_0^1 K(\frac{x - x_0}{h})f(x)dx \overset{x_0 = ch}{=} \frac{n}{h}\int_0^{hc+h} K(\frac{x}{h} - c)f(x)dx \overset{u = \frac{x}{h} - c}{=} n\int_{-c}^1 K(u)f(hu + x_0)du = n\{f(x_0)\mu_{0,c} + o(1)\},$$

$$\text{Var}(S_n) \leq \text{E}(\sum_{i=1}^n K_h^2(X_i - x_0)) = \frac{n}{h^2}\int_0^1 K^2(\frac{x - x_0}{h})f(x)dx = \frac{n}{h}\int_{-c}^1 K^2(u)f(hu + x_0)du = \frac{n}{h}\{f(x_0)\nu_{0,c} + o(1)\},$$

Therefore $\frac{S_n}{n} = f(x_0)\mu_{0,c} + o_p(1)$. Similarly, we can show that $\frac{T_n}{nh^{-1}} = f(x_0)\nu_{0,c} + o_p(1)$.
Thus:

$$\text{Var}(\hat{\mu}(x_0)|\mathbf{X}_n) = \frac{\sum_{i=1}^n K_h^2(X_i - x_0)\sigma^2}{(\sum_{i=1}^n K_h(X_i - x_0))^2} = \frac{\sigma^2 nh^{-1}\{f(x_0)\nu_{0,c} + o_p(1)\}}{n^2\{f(x_0)\mu_{0,c} + o_p(1)\}^2} = \frac{\sigma^2\nu_{0,c}}{f(x_0)\mu_{0,c}^2}\frac{1}{nh} + o_p(\frac{1}{nh}),$$

For bias, first note that:

$$\text{Bias}(\hat{\mu}(x_0)|\mathbf{X}_n) = \frac{\sum_{i=1}^n K_h(X_i - x_0)(\mu(X_i) - \mu(x_0))}{\sum_{i=1}^n K_h(X_i - x_0)} = \frac{\sum_{i=1}^n K_h(X_i - x_0)(\mu^{'}(x_0)h\frac{X_i - x_0}{h} + o_p(h))}{\sum_{i=1}^n K_h(X_i - x_0)}$$

$$\text{E}\left(\sum_{i=1}^n K_h(X_i - x_0)(\mu^{'}(x_0)h\frac{X_i - x_0}{h} + o_p(h))\right) = \mu^{'}(x_0)f(x_0)\mu_{1,c}nh + o(nh)$$

$$\text{Var}\left(\sum_{i=1}^n K_h(X_i - x_0)(\mu^{'}(x_0)h\frac{X_i - x_0}{h} + o_p(h))\right) \leq nh(\mu^{'}(x_0))^2 f(x_0)\nu_{2,c} + o(nh)$$

Therefore:

$$\sum_{i=1}^n K_h(X_i - x_0)(\mu^{'}(x_0)h\frac{X_i - x_0}{h} + o_p(h)) = \mu^{'}(x_0)f(x_0)\mu_{1,c}nh + o_p(nh)$$

Therefore:

$$\text{Bias}(\hat{\mu}(x_0)|\mathbf{X}_n) = \frac{\mu^{'}(x_0)f(x_0)\mu_{1,c}nh + o_p(nh)}{n(f(x_0)\mu_{0,c} + o_p(1))} = \mu^{'}(x_0)\frac{\mu_{1,c}}{\mu_{0,c}}h + o_p(h)$$

Problem 2

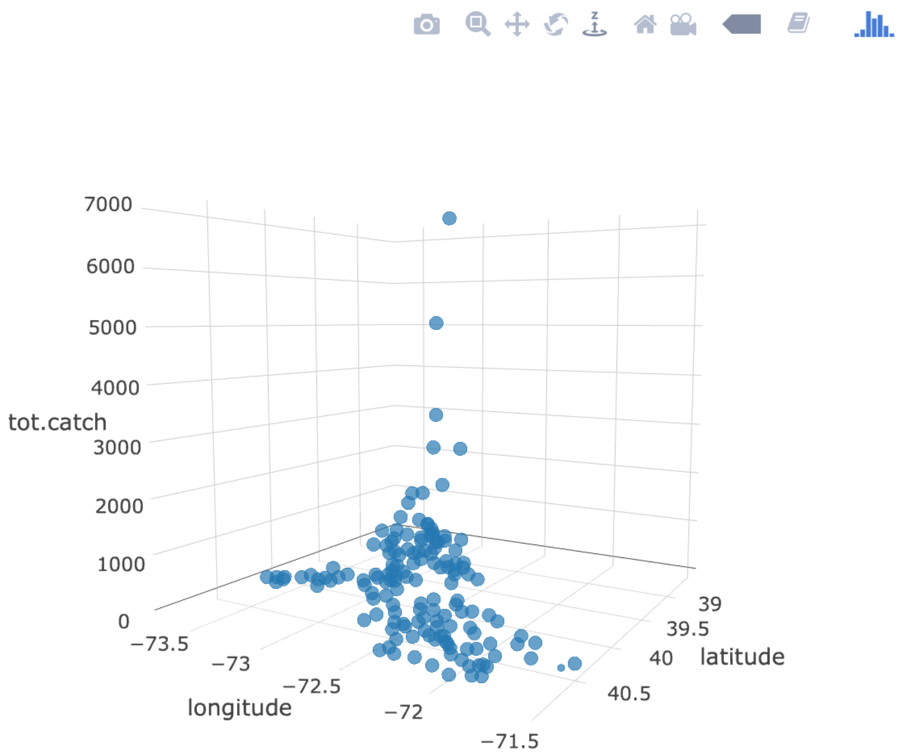


Figure 1: 3D Scatter Plot — Raw Data

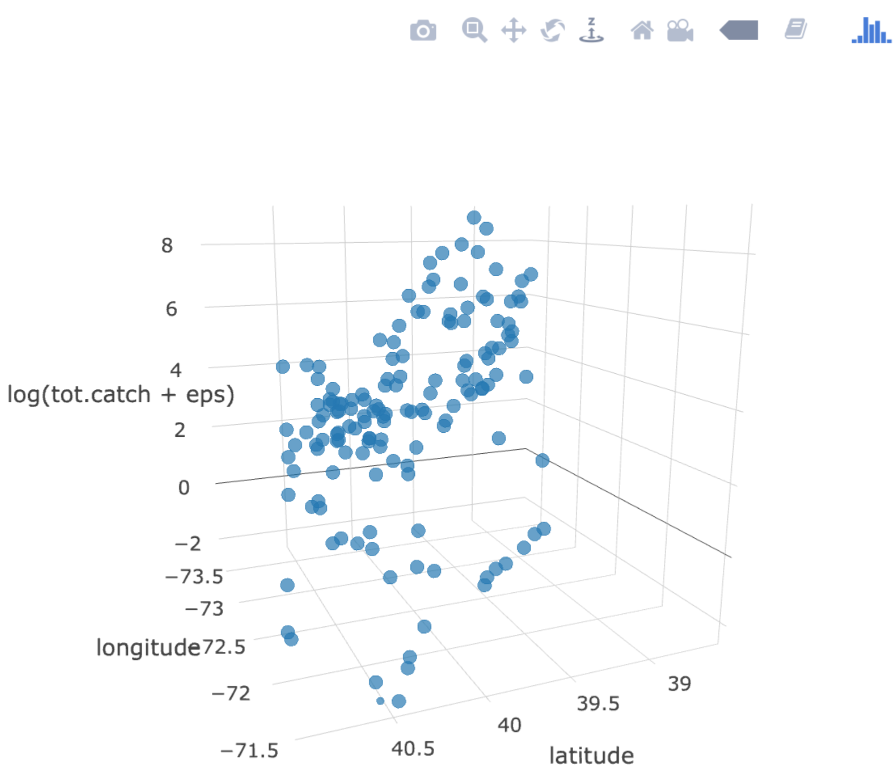


Figure 2: 3D Scatter Plot — Log Data

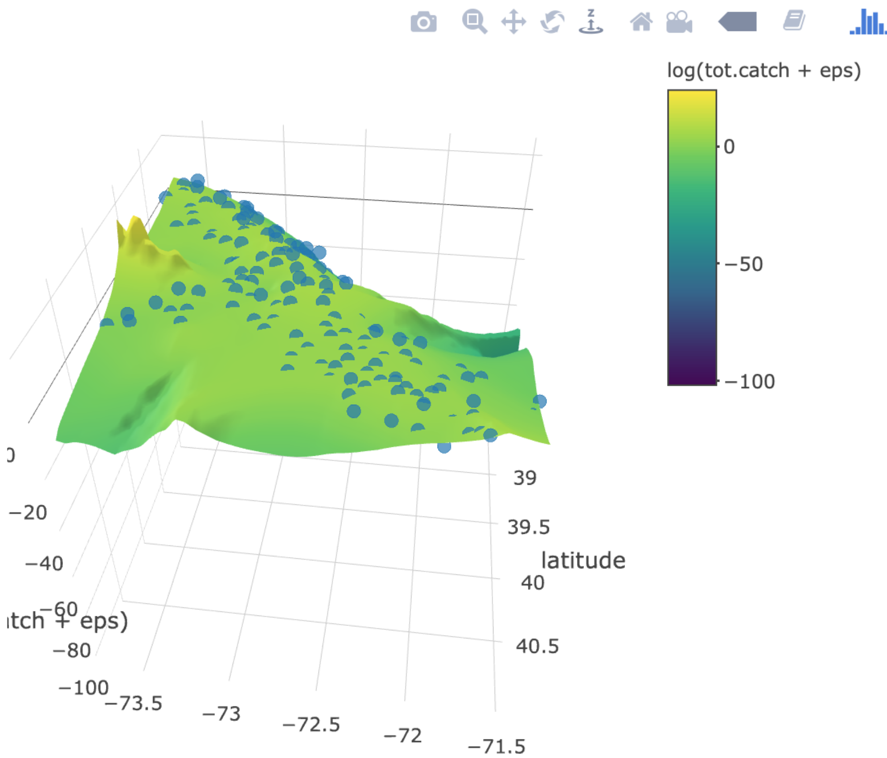


Figure 3: Local Linear Bivariate Fit — Log Data

```
library (SemiPar)
library(locfit)
```

```
## locfit 1.5-9.1    2013-03-22
```

```
library(plotly)
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

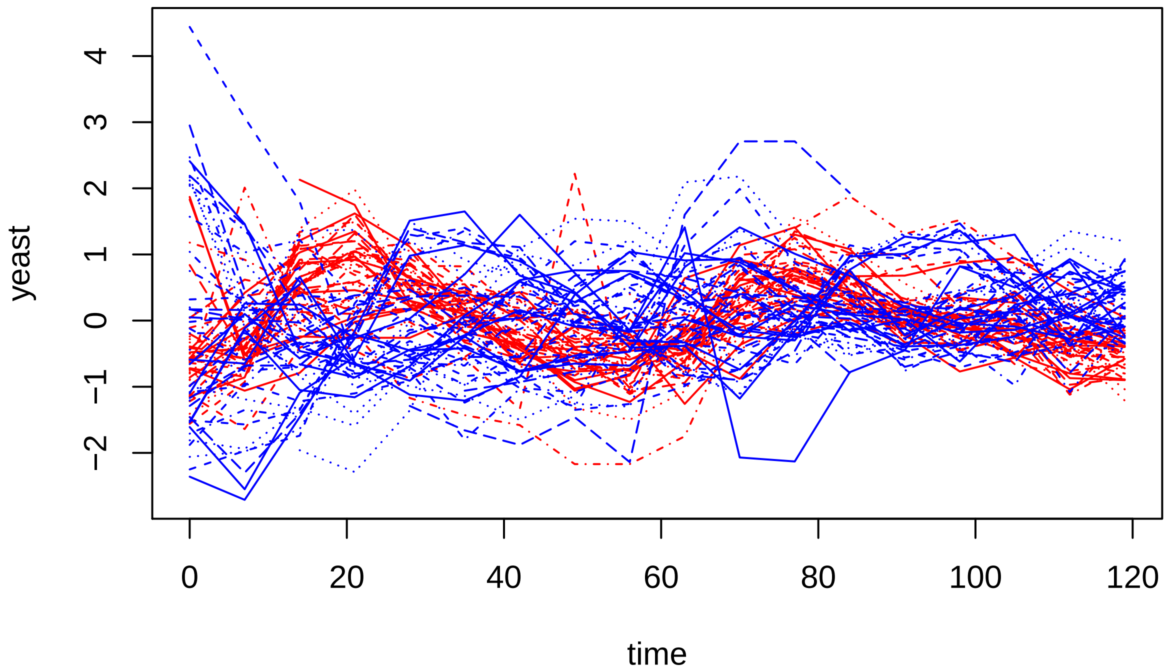
```
data(scallop)

## plot raw data
p0 <- plot_ly() %>%
```

```
  add_trace(
  data = scallop,
  x = ~latitude,
  y = ~longitude,
  z = ~tot.catch,
  type = "scatter3d",
  mode = "markers",
  size = 3
  )


## plot log data
eps <- 0.1
p <- plot_ly() %>%
  add_trace(
  data = scallop,
  x = ~latitude,
  y = ~longitude,
  z = ~log( tot.catch + eps),
  type = "scatter3d",
  mode = "markers",
  size = 3
)


## locpol fit using gcv
m <- 50

## gcv
nnCandidate <- seq(0.1, 1, 0.1)
gcvScores <- sapply(
  nnCandidate,
  function(nn) {
    gcv( log(tot.catch + eps) ~ lp(latitude, longitude, deg=1, nn=nn),
         scallop, ev=lfgrid(mg=m))['gcv']
  }
)

nnGCV <- nnCandidate[which.min(gcvScores)]

## fit
res <- locfit( log(tot.catch + eps) ~ lp(latitude, longitude, deg=1, nn = nnGCV),
               scallop, ev=lfgrid(mg=m))


xGrid <- seq(min(scallop$latitude), max(scallop$latitude), length.out=m)
yGrid <- seq(min(scallop$longitude), max(scallop$longitude), length.out=m)
z <- matrix(predict(res), m, m, byrow = TRUE)



p1 <- p %>%
  add_trace(
```

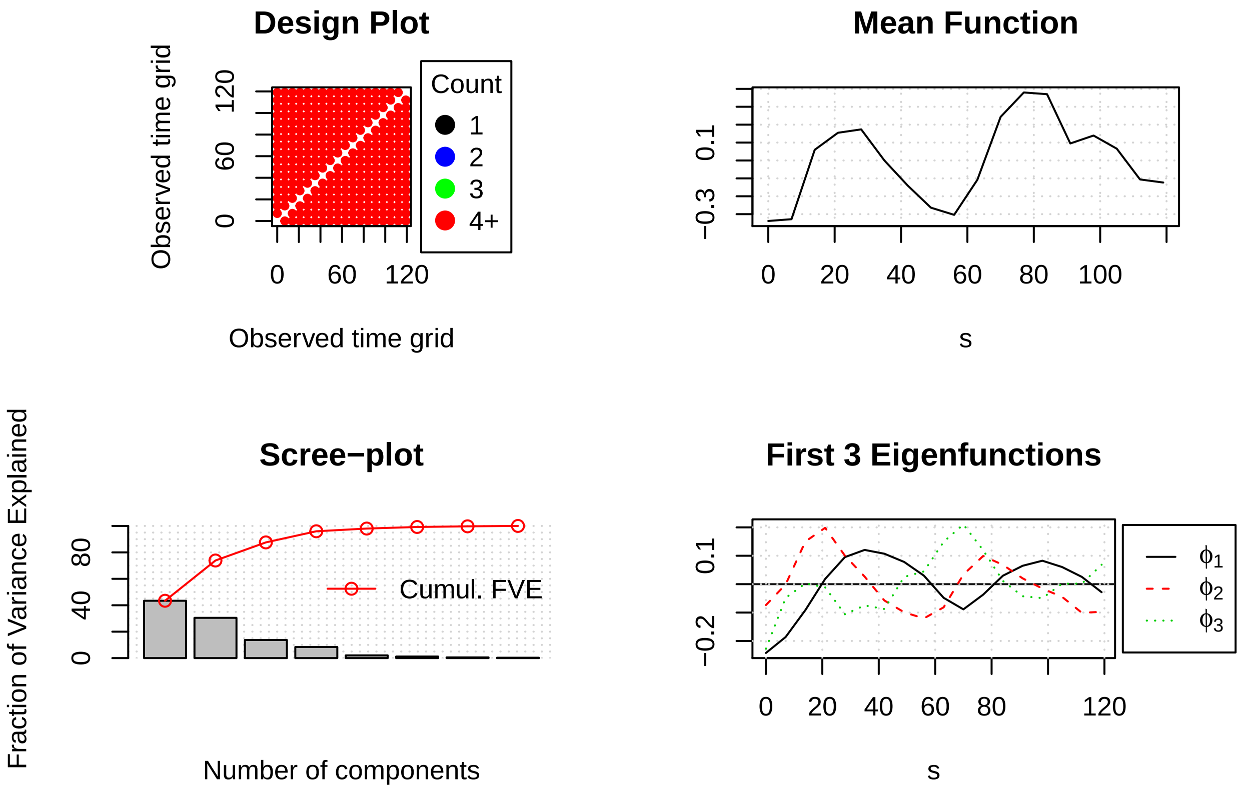
```
  x = xGrid,
  y = yGrid,
  z = z,
  type = "surface"
)
```

problem 3

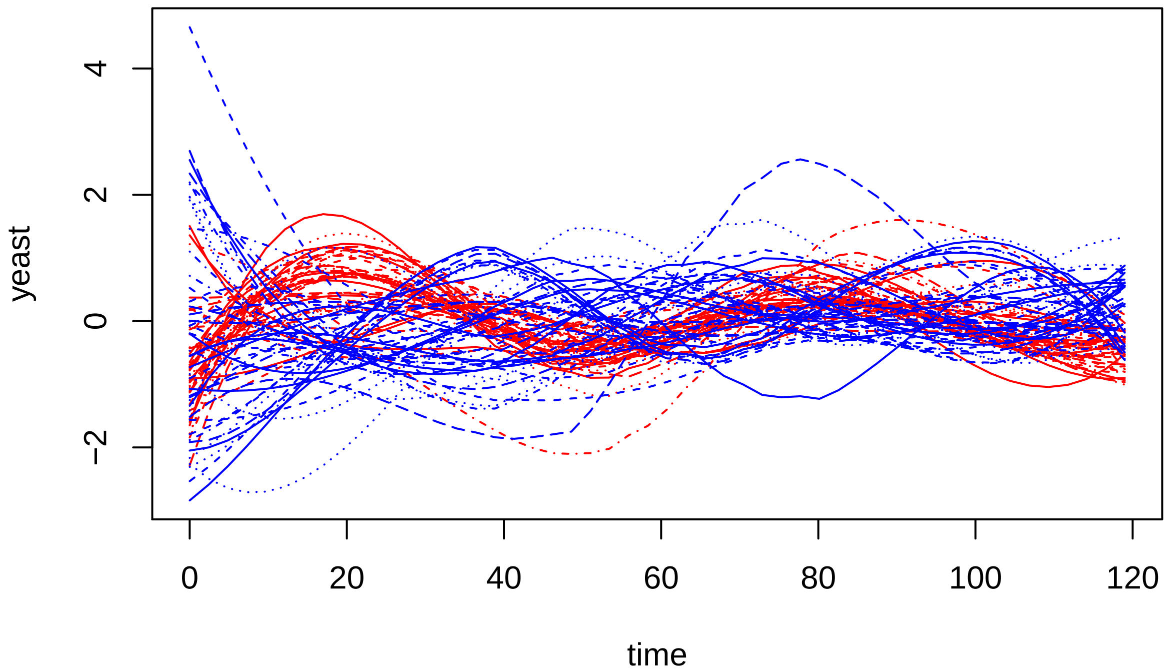
```
library(fdapace)
yeast <- read.table(file = "/Users/apple/Desktop/ISU 2019 fall/STAT547/data/yeast.txt")
time <- as.numeric( sub("alpha", "", names(yeast)) )

## original data
X <- as.matrix(yeast)
matplot(time, t(X), type='l', col=c(rep(2,44), rep(4,45)),
        xlab = "time", ylab = "yeast")
```



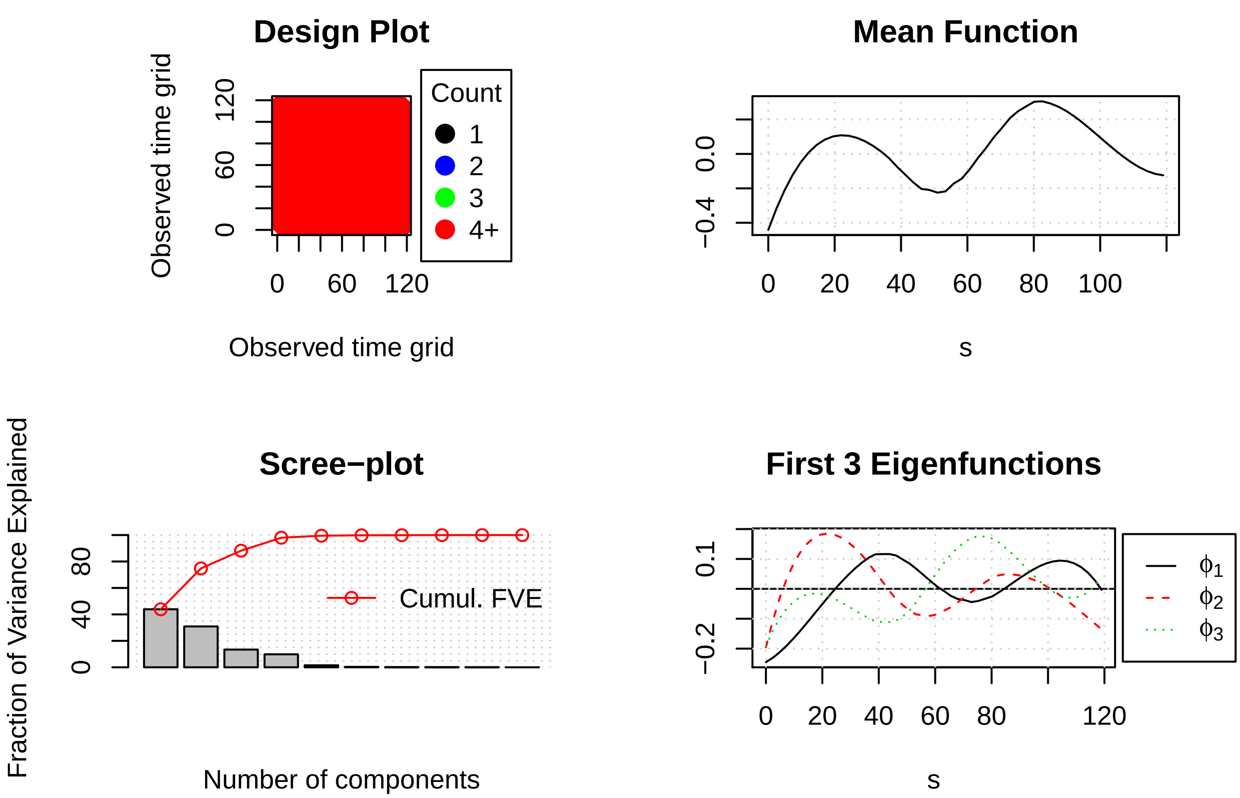

```
data_o <- MakeFPCAInputs(tVec = time, yVec = X)
res_o <- FPCA(data_o$Ly, data_o$Lt)
plot(res_o)
```




```
## smoothed data
n <- nrow(X)
m <- 50
X_smooth <- matrix(0, ncol = m, nrow = n)
tGrid <- seq(range(time)[1], range(time)[2], length.out = m)
for (i in 1:n){
  fit <- locfit( X[i,] ~ lp(time, deg=3), ev=lfgrid(mg=m))
  X_smooth[i,] <- predict(fit)
}

matplot(tGrid, t(X_smooth), type='l', col=c(rep(2,44), rep(4,45)),
        xlab = "time", ylab = "yeast")
```

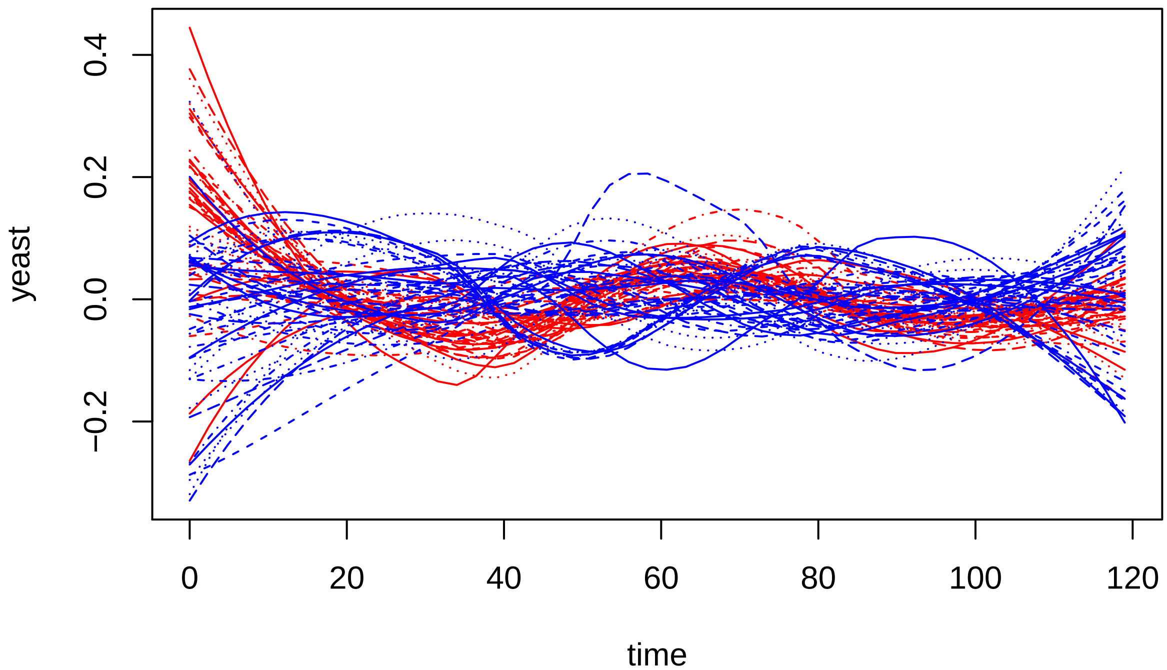



```
data_s <- MakeFPCAInputs(tVec = tGrid, yVec = X_smooth)
res_s <- FPCA(data_s$Ly, data_s$Lt)
plot(res_s)
```

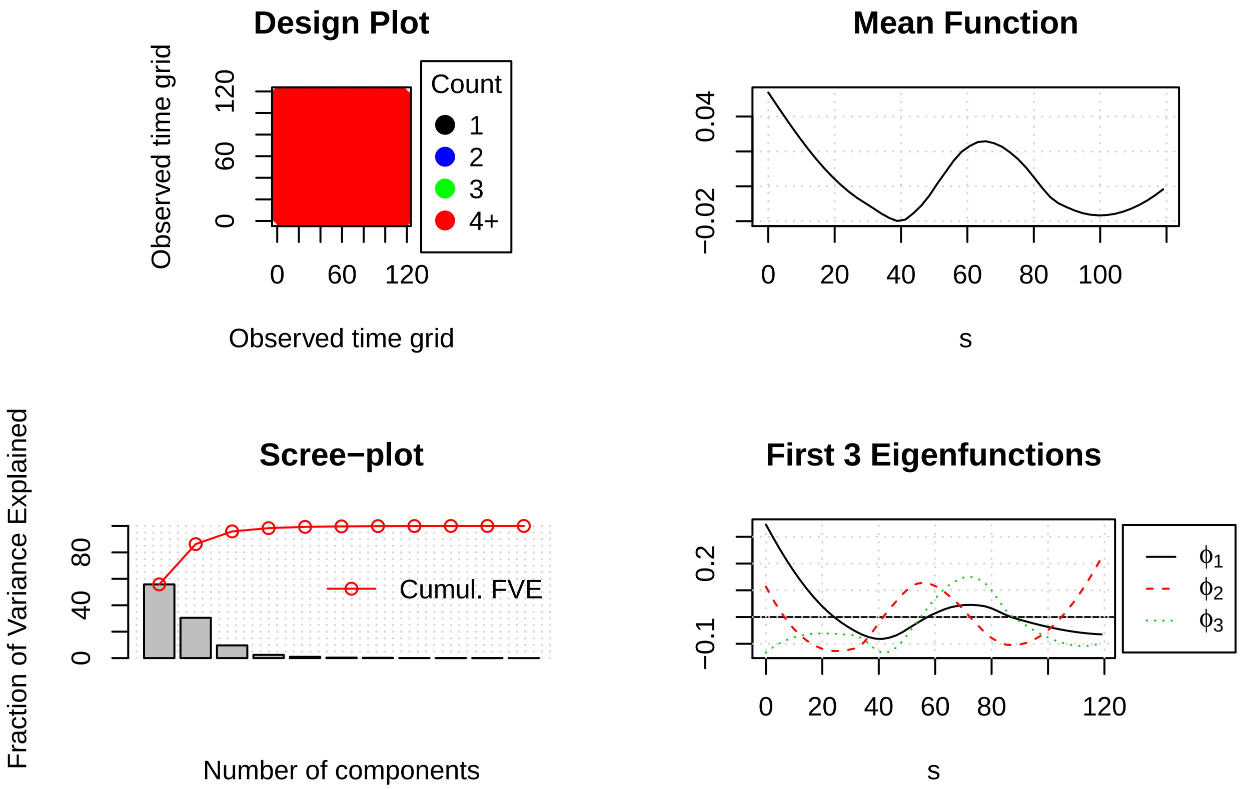



```
## derivative
X_deriv <- matrix(0, ncol = m, nrow = n)
for (i in 1:n){
  fit <- locfit( X_smooth[i,] ~ lp(tGrid, deg=4, nn=0.2), ev=lfgrid(mg=m), deriv=1)
```

```
  X_deriv[i,] <- predict(fit)
}

matplot(tGrid, t(X_deriv), type='l', col=c(rep(2,44), rep(4,45)),
        xlab = "time", ylab = "yeast")
```




```
data_d <- MakeFPCAInputs(tVec = tGrid, yVec = X_deriv)
res_d <- FPCA(data_d$Ly, data_d$Lt)
plot(res_d)
```



Design Plot
Observed time grid
0
60
120
Count
1
2
3
4+
Mean Function
0.04
−0.02
0
20
40
60
80
100
s
Scree−plot
Fraction of Variance Explained
Cumul. FVE
Number of components
First 3 Eigenfunctions
0.2
−0.1
ϕ_1
ϕ_2
ϕ_3