kokoszka FDA CH1

Noa Jeong

2024-07-12

- github : https://minitistics.tistory.com/27

1. First steps in the analysis of functional data

```
rm(list=ls())
```

You need to update R version (current: R 4.4.1) : update R (not R studio)

```
#install.packages("fda")
library(fda)
```

```
## Loading required package: splines
## Loading required package: fds
## Loading required package: rainbow
## Loading required package: MASS
## Loading required package: pcaPP
## Loading required package: RCurl
## Loading required package: deSolve
##
## Attaching package: 'fda'
## The following object is masked from 'package:graphics':
##
##     matplot
```

$$x_n(t_{j,n}) \in \mathbb{R} \quad t_{j,n} \in [T_1, T_2] \quad n = 1, 2, ..., N \quad j = 1, ..., J_n$$

N : number of curves J_n : observed time point

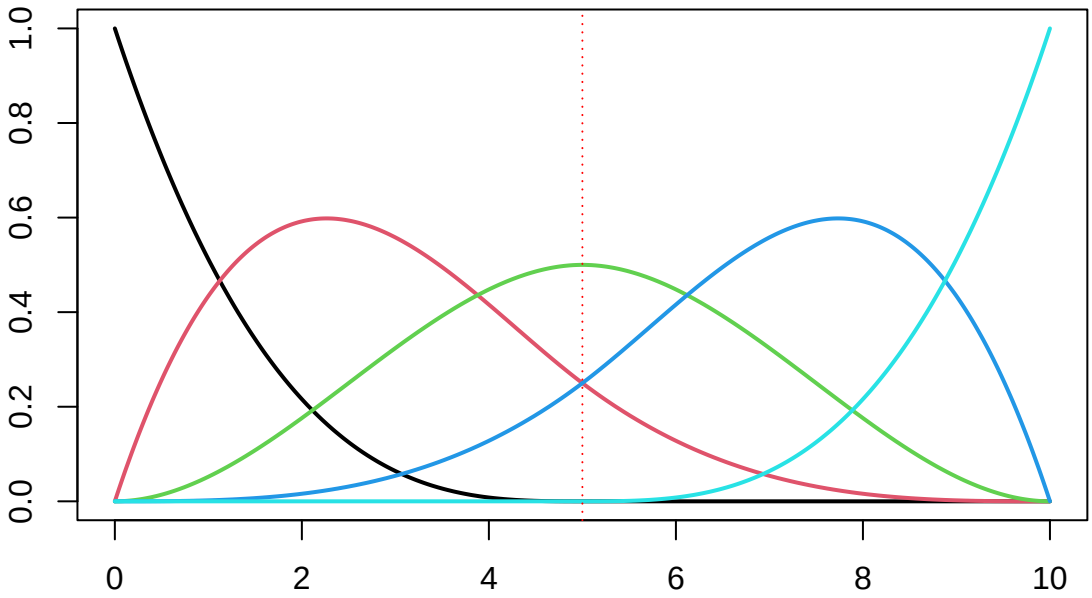
1.1 Basis Expansion

$$X_n(t) \approx \sum_{m=1}^{N} c_{nm} B_m(t)$$

B_m : basis functions (splines wavelets, sine cosine functions)

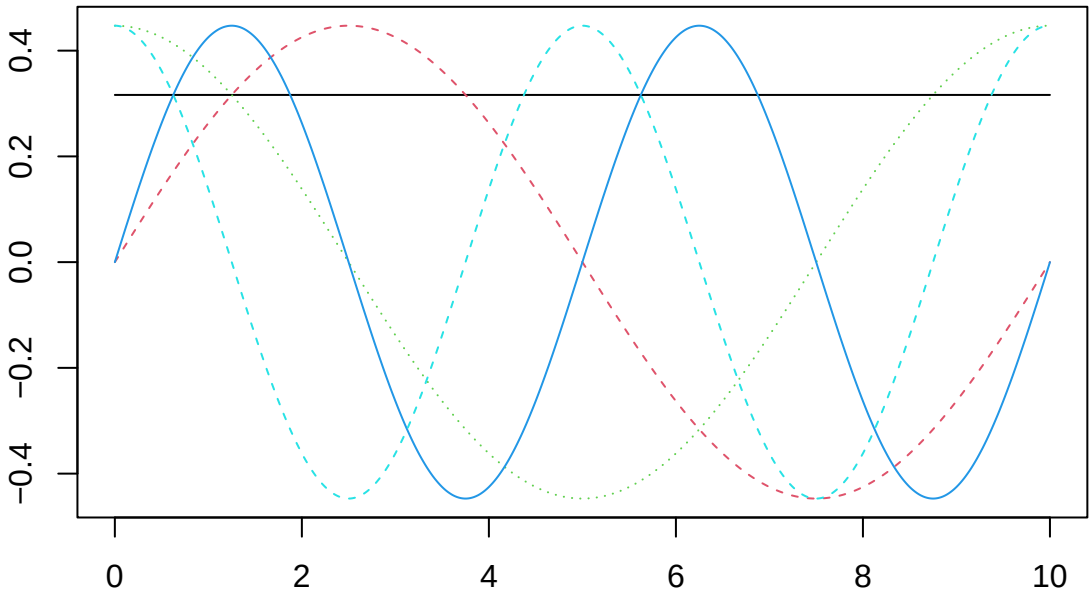
- 5 B-spline basis functions defined on the interval [0,10].

```
spline.basis = create.bspline.basis(rangeval=c(0,10), # interval
                                    nbasis=5) # number of basis functions
plot(spline.basis,lty=1,lwd=2)
```



- first five Fourier basis functions

```
fourier.basis = create.fourier.basis(rangeval=c(0,10),
                                     nbasis=5)
plot(fourier.basis)
```



Fourier system is usually only suitalbe for functions which have approximately the same values at the beginnin and the end of the interval.

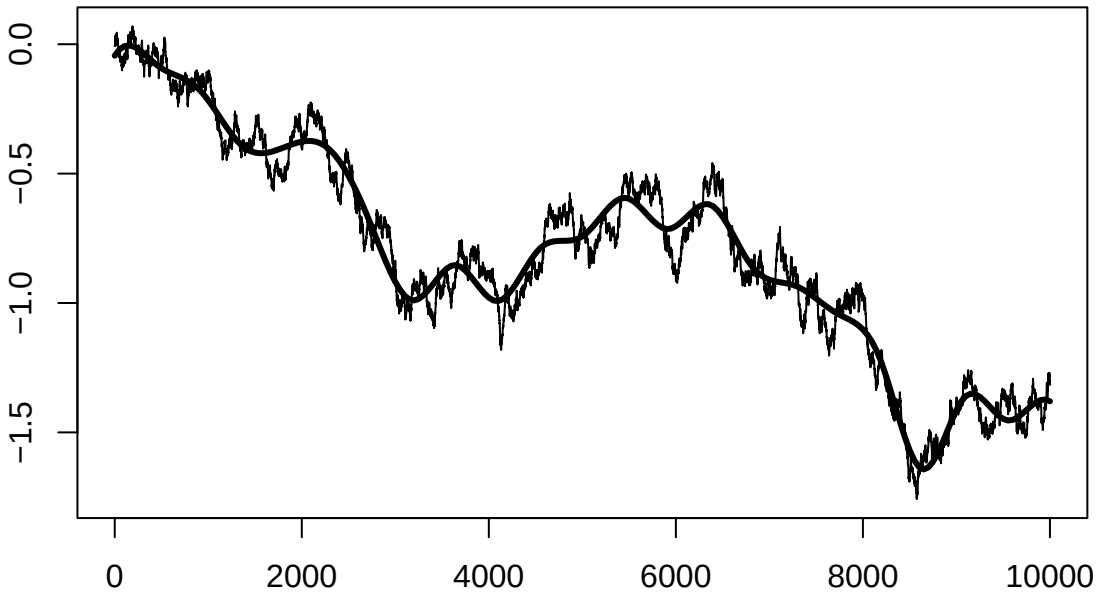
Example 1.1.1 [B-spline expansion on Wiener process]

appropriate random walk

$$S_i = \frac{1}{\sqrt{K}}, \quad N_k \sim iid\, N(0,1)$$

- Random walk and its expansion using 25 B-Spline basis functions

```
Wiener = cumsum(rnorm(10000)/100) # random walk on [0,K], K=10^4
plot.ts(Wiener, xlab="", ylab="")

B25.basis = create.bspline.basis(rangeval=c(0,10000),
                                 nbasis=25)
Wiener.fd = smooth.basis(y=Wiener, fdParobj=B25.basis) # create functional data object Wiener.fd
lines(Wiener.fd, lwd=3)
```



1.2 Sample Mean and covariance

raw data -> functional objects. by suitable basis expansion

pointwise mean & pointwise standard deviation

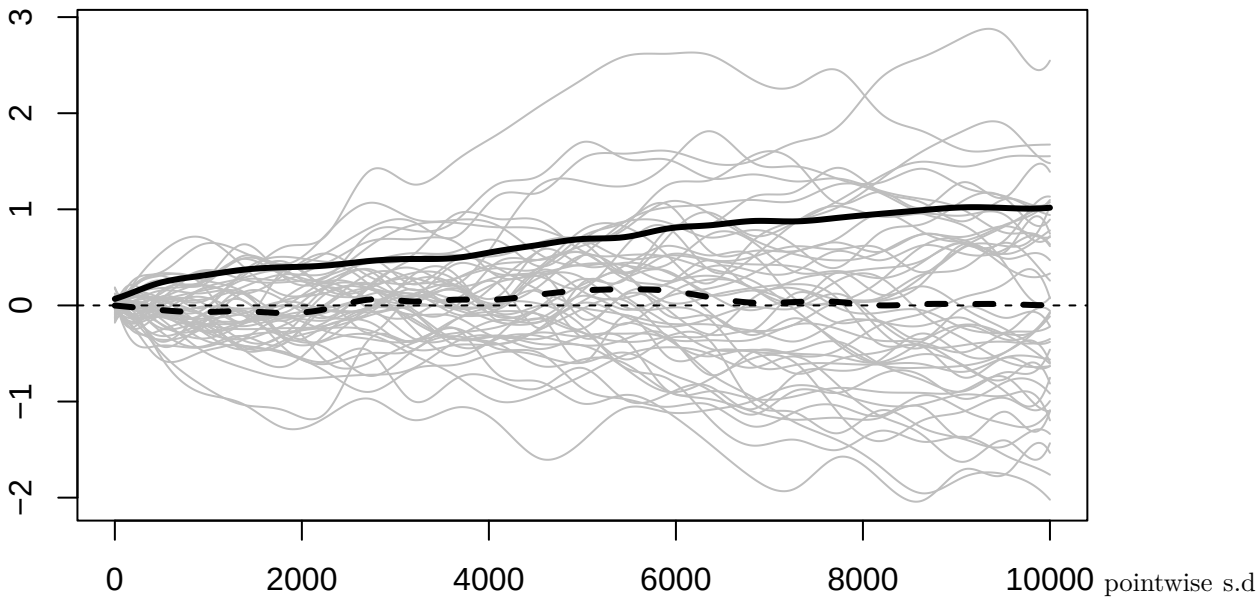
$$\bar{X}_N(t) = \frac{1}{N}\sum_{n=1}^{N} X_n(t) \quad SD_X(t) = \{\frac{1}{N-1}\sum_{n=1}^{N}(X_n(t) - \bar{X}_N(t))^2\}^{1/2}$$

Example 1.2.1 [Pointwise mean and SD]

```
N=50
W.mat=matrix(0, ncol=N, nrow=10000)
for(n in 1:N){
  W.mat[,n] = cumsum(rnorm(10000))/100
}

B25.basis = create.bspline.basis(rangeval=c(0,10000),
                                 nbasis=25)
W.fd = smooth.basis(y=W.mat,
                    fdParobj = B25.basis)

plot(W.fd,ylab='',xlab='',col='gray',lty=1)
```

```
## [1] "done"
```

```
W.mean <- mean.fd(W.fd$fd)
W.sd <- std.fd(W.fd$fd)
lines(W.sd, lwd=3);lines(W.mean,lty=2,lwd=3)
```



pointwise s.d : typical variablity of curves at time point t

But no information on how values of curve at point t relate to point s

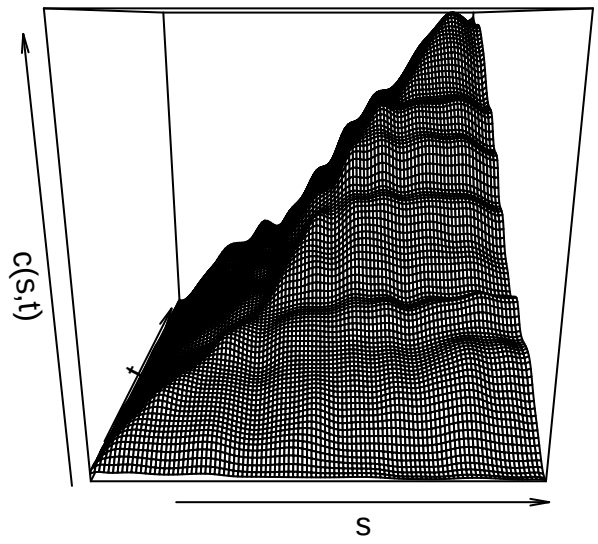
- **sample covariance function**

$$\hat{c}(t,s) = \frac{1}{N-1}\sum_{n=1}^{N}(X_n(t) - \bar{X}_N(t))(X_n(s) - \bar{X}_N(s))$$

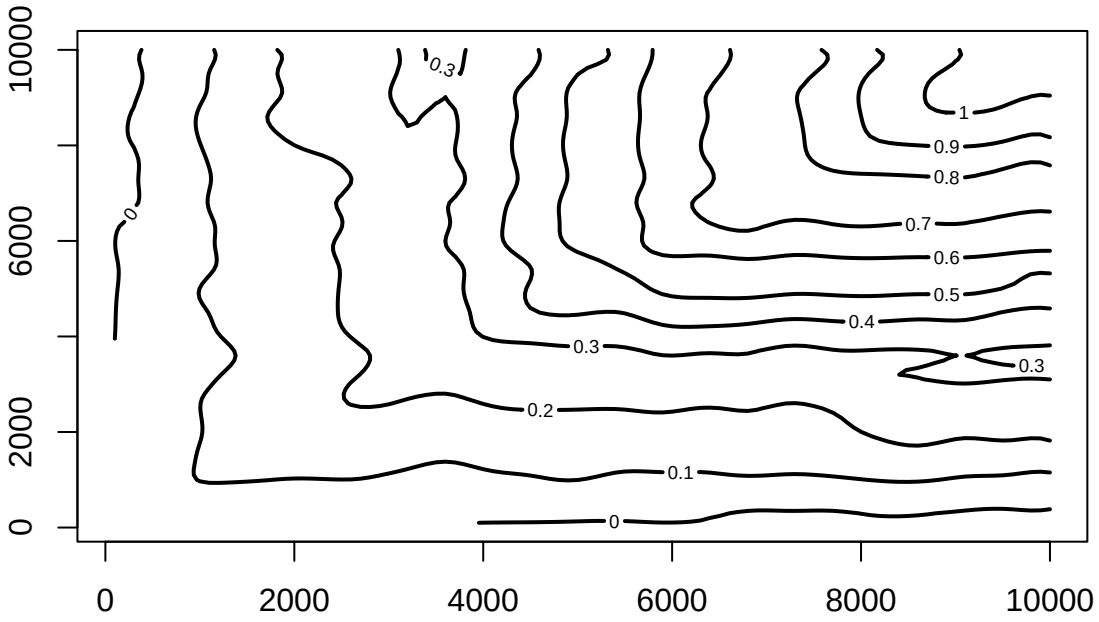
interpretaion as variance covariance matrix

Example 1.2.2 [Sample covariance function]

```
# Use the object W.fd generated in the previous example.
W.cov = var.fd(W.fd$fd) # $fd extracts function values
grid=(1:100)*100

W.cov.mat=eval.bifd(grid,grid,W.cov)
persp(grid,grid,W.cov.mat,xlab='s',
      ylab='t',zlab='c(s,t)')
```




```
contour(grid,grid,W.cov.mat,lwd=2)
```



$\hat{c}(t,s)$ is given by $c(t,s) = min(t,s)$

1.3 Principal component functions

EFPC's : Estimated fuctional principal components
related to the sample covariance function $\hat{c}(t, s)$

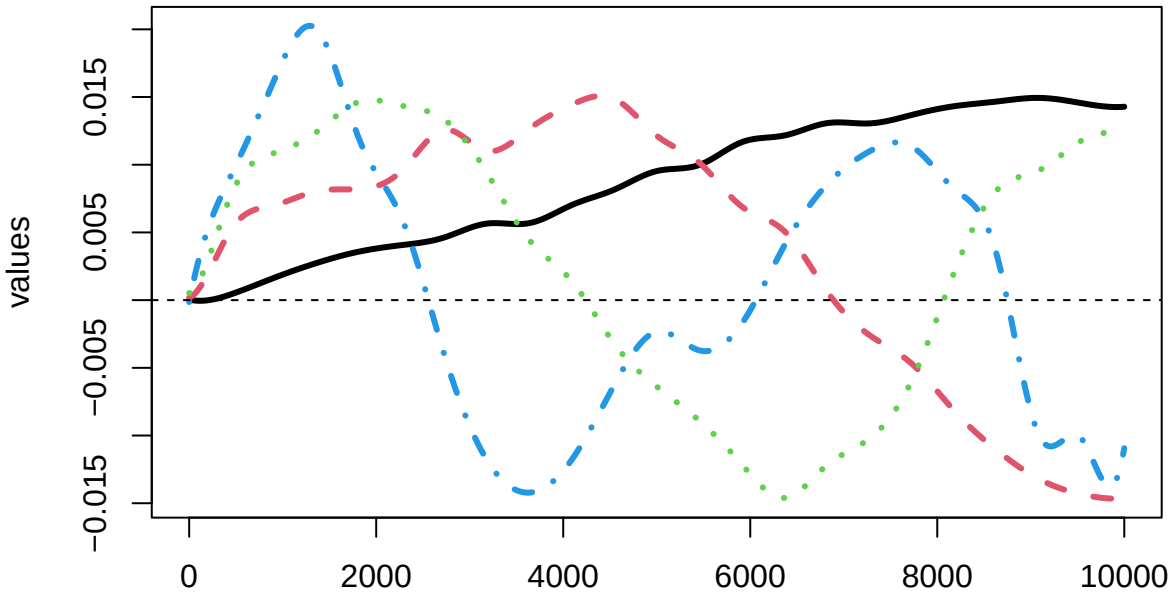
- ceneterd function

$$X_n(t) - \bar{X}_n(t) \approx \sum_{j=1}^{p} \xi_{nj} \hat{v}_j(t)$$

p is much smaller than M

$\hat{v}_j(t)$ are computed from the observed functions $X_1, X_2, ..X_N$ after converting them to functional objects

```
W.pca = pca.fd(W.fd$fd, nharm=4)
plot(W.pca$harmonics, lwd=3)
```




```
## [1] "done"
```

```
W.pca$varprop
```

```
## [1] 0.82349972 0.08368658 0.04323643 0.01610740
```

$\hat{v}_1$ black line : most pronounced pattern of the deviation from the mean function of a randomly selected trajectory.

coefficient ξ_{n1} quantifies the contribution of $\hat{v}_1$ to its shape.

$\hat{v}_2$ red line : second most important mode of mean functions of 50 random walks. It is second msot important mode of variability which is orthogonal to $\hat{v}_1$

ξ_{nj} is called the score of X_n with respect to $\hat{v}_j$.; the smaller the percentages, the smaller the scores.

EFPC's $\hat{v}_j$ are orthonormal

$$\int \hat{v}_j(t)\hat{v}_i(t)dt = \begin{cases} 0 & \text{if } j \neq i \\ 1 & \text{if } j = i \end{cases}$$

1.4 Analysis of BOA stock returns

Bank of America

1997.4.9 - 2007.4.2, 9:30 ~ 16:00 stock values recorded every minute

$t \in (0, 6.5)$

2511 days of data, each day consisting 390 measurements

functional observation : *cumulative log-return*

$$R_n(t) := log(P_n(t)) - log(P_n(0)) \approx \frac{P_n(t) - P_n(0)}{P_n(0)}$$

$R_n(t)$: how an investment, made at opening, evolves over the course of the day

$P_n(t)$: value of the stock on the day n at time t

- plot of first ten cumulative log returns for BOA

```
#install.packages('striprtf')
#install.packages('readr')
library(striprtf)
library(readr)

extracted_text <- striprtf::read_rtf('Bank of America Data.rtf')
```

```
BOA = read.table('BOA.txt', header=TRUE)
Dates <- dimnames(BOA)[[1]] # 2511 days
BOA <- data.matrix(BOA)
Outlier = which(Dates=='08/26/2004')
BOA <- BOA[-Outlier,]
```

- Plot of first 10 cumulative log returns for BOA

```
N <- dim(BOA)[1] # 2510
M <- dim(BOA)[2] # 390 measurements
Times <- seq(0,6.5,length=M)
log_BOA <- log(BOA)-matrix(log(BOA)[,1], nrow=N, ncol=M) # R_n(t)
bspline_basis <- create.bspline.basis(rangeval=c(0,6.5), norder=4, nbasis=200)
log_BOA_f <- Data2fd(Times,t(log_BOA), basisobj = bspline_basis)
plot(log_BOA_f[1:10], xlab='',ylab='',lwd=1.5)
```


```
## [1] "done"
```

- plot of mean function of BOA cumulative returns. Pointwise 95% confidence intervals are included in red.

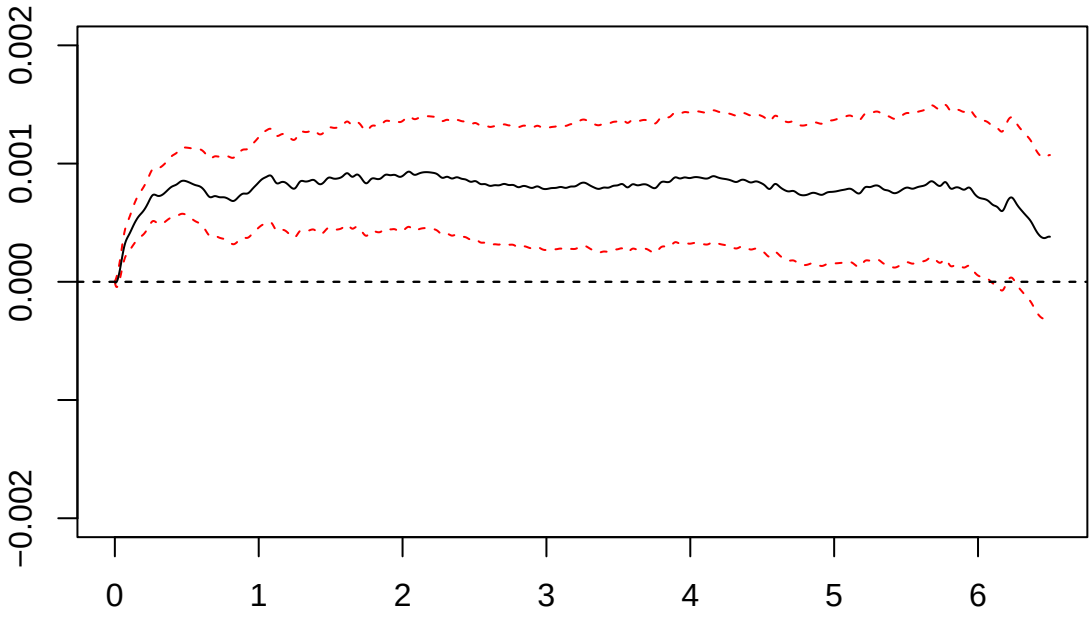
```
muhat <- mean.fd(log_BOA_f)
sdhat <- sd.fd(log_BOA_f)
SE_hat_U <- fd(basisobj = bspline_basis) # create upper CI bound
SE_hat_L <- fd(basisobj = bspline_basis) # create lower CI bound
SE_hat_U$coefs <- 2*sdhat$coefs/sqrt(N) + muhat$coefs
SE_hat_L$coefs <- -2*sdhat$coefs/sqrt(N) + muhat$coefs
plot.fd(SE_hat_U, ylim=c(-0.002,0.002), col='red',lty=2,xlab='',ylab='')
```

```
## [1] "done"
```

```
plot.fd(SE_hat_L, add=T, col='red', lty=2)
```

```
## [1] "done"
```

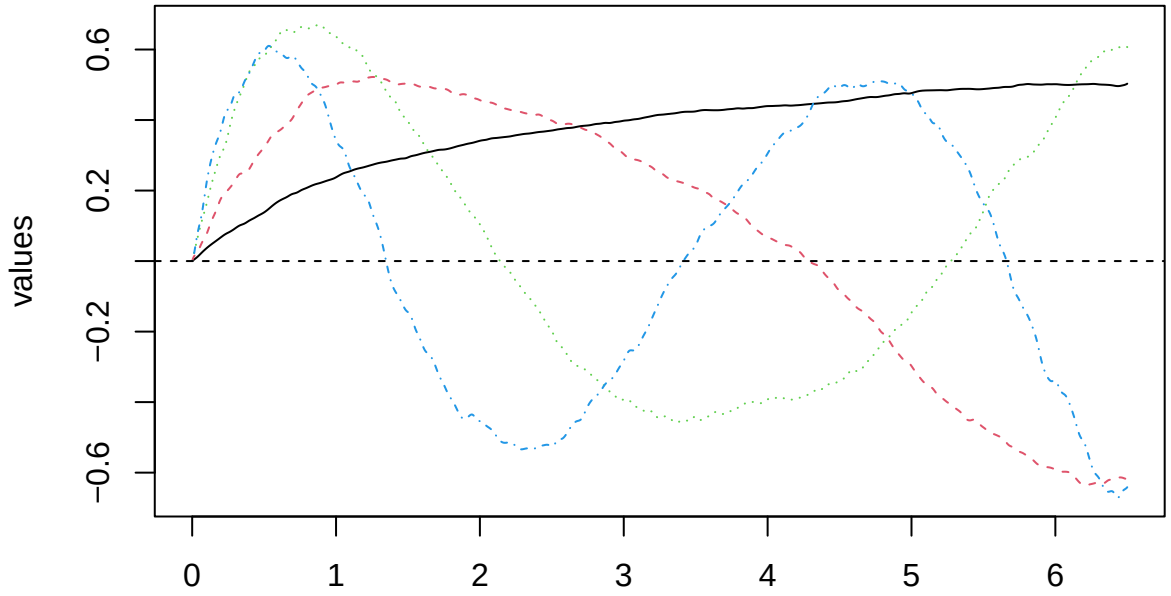
```
plot.fd(muhat, add=T)
```




```
## [1] "done"
```

- plot of first four PCs of BOA

```
log_BOA_pca <- pca.fd(log_BOA_f,nharm=4)
plot(log_BOA_pca$harmonics,lwd=1)
```




```
## [1] "done"
```

1.5 Diffusion tensor imaging

DTI : magnetic resonance imaging methodology, to measure the diffusion of water in the brain. Utilize DTI to generate image of white matter in the brain.

Fractional anisotropy is a value 0~1 which measure the level of anisotropy, therefore the quantity of white matter at a particular location.

376 patients, each tract measure at 93 equally spaced location.

$t_{j,n}$: spatial rather than temporal location.

```
#install.packages('refund')
library(refund)
data(DTI)
str(DTI)
```

```
## 'data.frame':    382 obs. of  9 variables:
##  $ ID        : num  1001 1002 1003 1004 1005 ...
##  $ visit     : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ visit.time: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Nscans    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ case      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ sex       : Factor w/ 2 levels "male","female": 2 2 1 1 1 1 1 1 1 1 ...
##  $ pasat     : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ cca       : num [1:382, 1:93] 0.491 0.472 0.502 0.402 0.402 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:382] "1001_1" "1002_1" "1003_1" "1004_1" ...
##   .. ..$ : chr [1:93] "cca_1" "cca_2" "cca_3" "cca_4" ...
##  $ rcst      : num [1:382, 1:55] 0.257 NaN NaN 0.508 NaN ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:382] "1001_1" "1002_1" "1003_1" "1004_1" ...
##   .. ..$ : chr [1:55] "rcst_1" "rcst_2" "rcst_3" "rcst_4" ...
```

```
Y <- DTI$cca
Y <- Y[-c(126,130,131,125,319,321),] # missing values
N <- dim(Y)[1]; M <- dim(Y)[2]
# N : number of patients(observations/functions)
# M : number of locations(time/spatial observed point)

argvals <- seq(0,1,length=M)
data_basis <- create.bspline.basis(c(0,1), # interval
                                   nbasis=10) # number of basis
Y.f <- Data2fd(argvals, t(Y), data_basis) # create smooth functions that fit scatter plot data
```

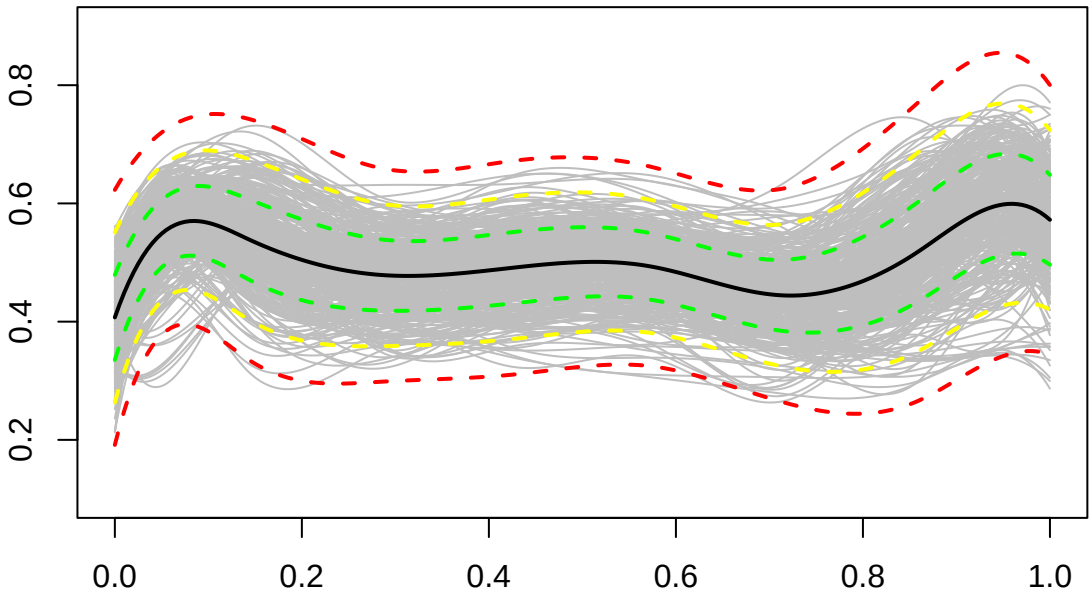
- plot of fractional anistoropy tract profiles oft he corpus collosum.

```
#dev.new(width=8,height=6)
plot(Y.f,lty=1,col='gray',xlab='',ylab='',ylim=c(0.1,0.9)) # Individuals(376 patients) tracts are plott
```

```
## [1] "done"
```

```
lines(mean.fd(Y.f), lwd=2) # mean function
# green, yellow, red indicates respecitvely one, two, three pointwise s.d. from the mean.
lines(mean.fd(Y.f)+std.fd(Y.f), lwd=2,lty=2,col='green')
lines(mean.fd(Y.f)-std.fd(Y.f), lwd=2,lty=2,col='green')
lines(mean.fd(Y.f)+2*std.fd(Y.f), lwd=2,lty=2,col='yellow')
lines(mean.fd(Y.f)-2*std.fd(Y.f), lwd=2,lty=2,col='yellow')
```

```
lines(mean.fd(Y.f)+3*std.fd(Y.f), lwd=2,lty=2,col='red')
lines(mean.fd(Y.f)-3*std.fd(Y.f), lwd=2,lty=2,col='red')
```



data follows an analog of the classic three standard deviation rule well, with nearly all curves falling between the red lines

1.6 Chapter 1 Problems

1.1

```
rm(list=ls())
library(fda)
data(pinch)

str(pinch) # 151 measurements of pinch for 20 replications(curves)
```

```
##  num [1:151, 1:20] -0.068 -0.09 0.1 -0.138 -0.053 -0.245 -0.047 -0.065 -0.123 -0.132 ...
```

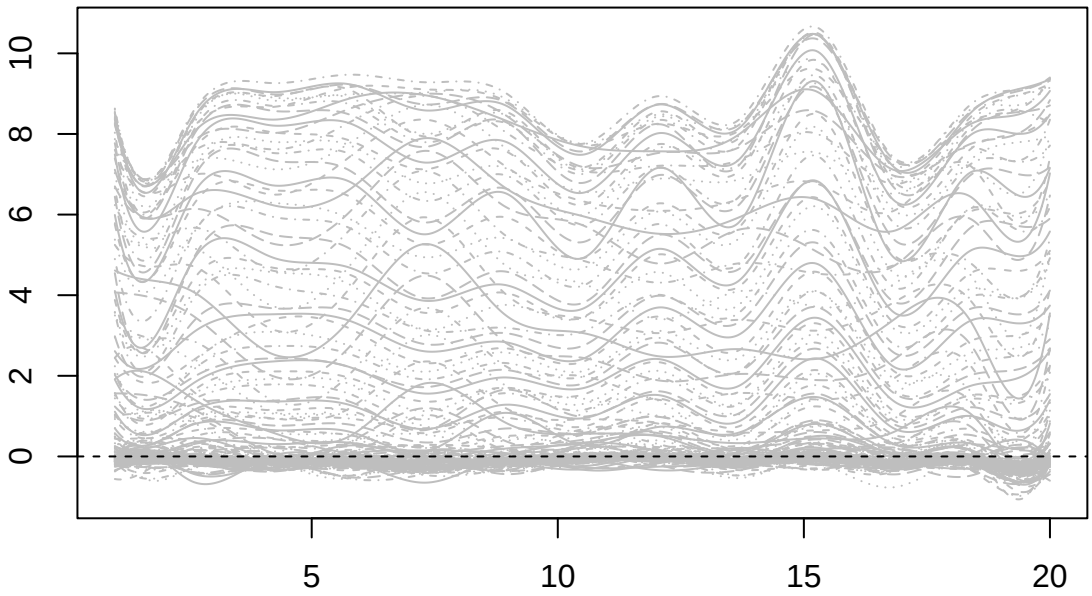
(a) Convert the pinch data to functional objects using B-splines of order four(cubic splines) and plot the 20 smoothed curves on one graph

```
dim(pinch)
```

```
## [1] 151  20
```

```
N <- dim(pinch)[1]; M <- dim(pinch)[2] # 151 20
```

```
argvals <- seq(1,20) # 20 replications
spline.basis <- create.bspline.basis(rangeval = c(1,20),
                                     nbasis=15, norder=4)

pinch.f <- Data2fd(argvals,t(pinch), spline.basis )
plot(pinch.f, col='gray',xlab='',ylab='')
```



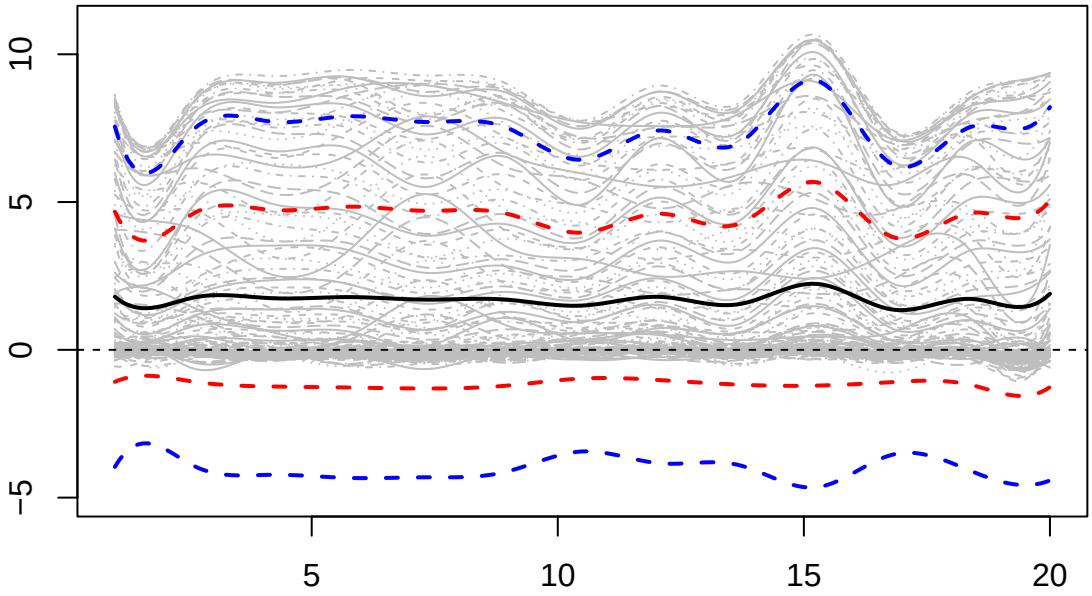

```
## [1] "done"
```

(b) Calculate the pointwise mean and SD and add them to plot.

```
mean.pinch <- mean.fd(pinch.f)
sd.pinch <- sd.fd(pinch.f)
plot(pinch.f, col='gray',xlab='',ylab='',ylim=c(-5,11))
```

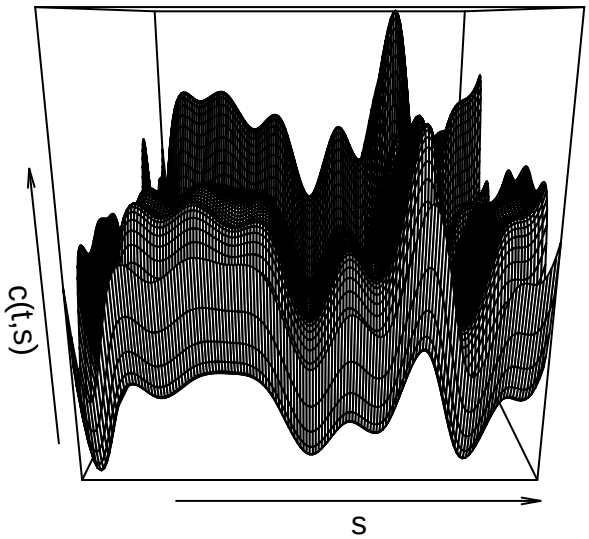
```
## [1] "done"
```

```
lines(mean.pinch, lwd=2)
lines(mean.pinch+sd.pinch, col='red',lwd=2,lty=2)
lines(mean.pinch-sd.pinch, col='red',lwd=2,lty=2)
lines(mean.pinch+2*sd.pinch, col='blue',lwd=2,lty=2)
lines(mean.pinch-2*sd.pinch, col='blue',lwd=2,lty=2)
```

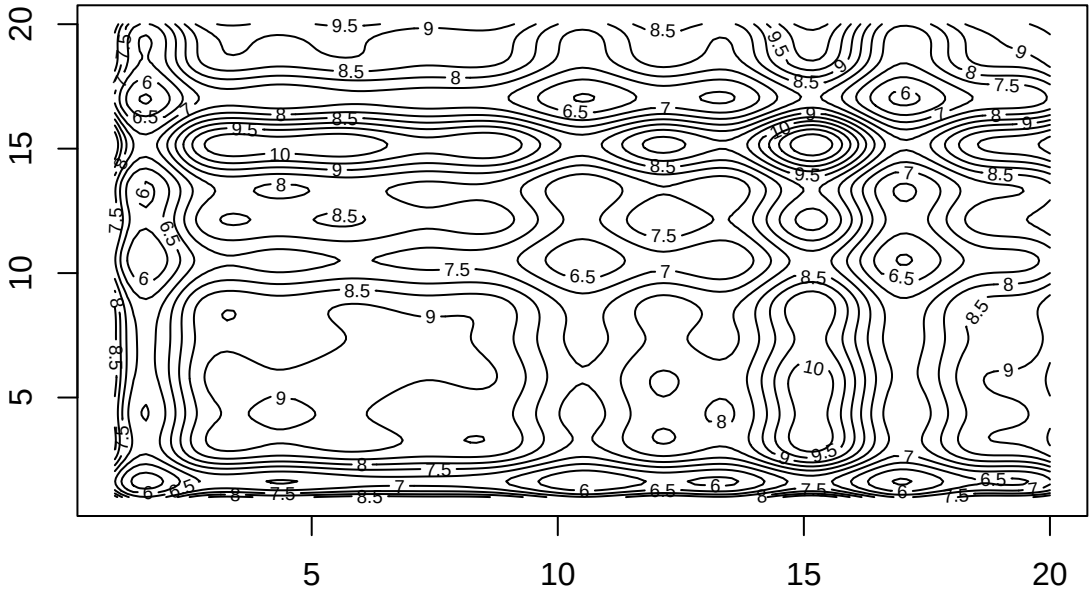


(c) Graph the persepctive and contour plots of the sample covariance function $\hat{c}(t,s)$ of the pinch curves.

```
#pinch.f$fd
cov.pinch <- var.fd(pinch.f)
#cov.pinch$coefs
grid <- seq(1,20,length=N)

cov.mat.pinch <- eval.bifd(grid,grid,cov.pinch)
persp(grid,grid,cov.mat.pinch,
      xlab='s',ylab='t',zlab='c(t,s)')
```

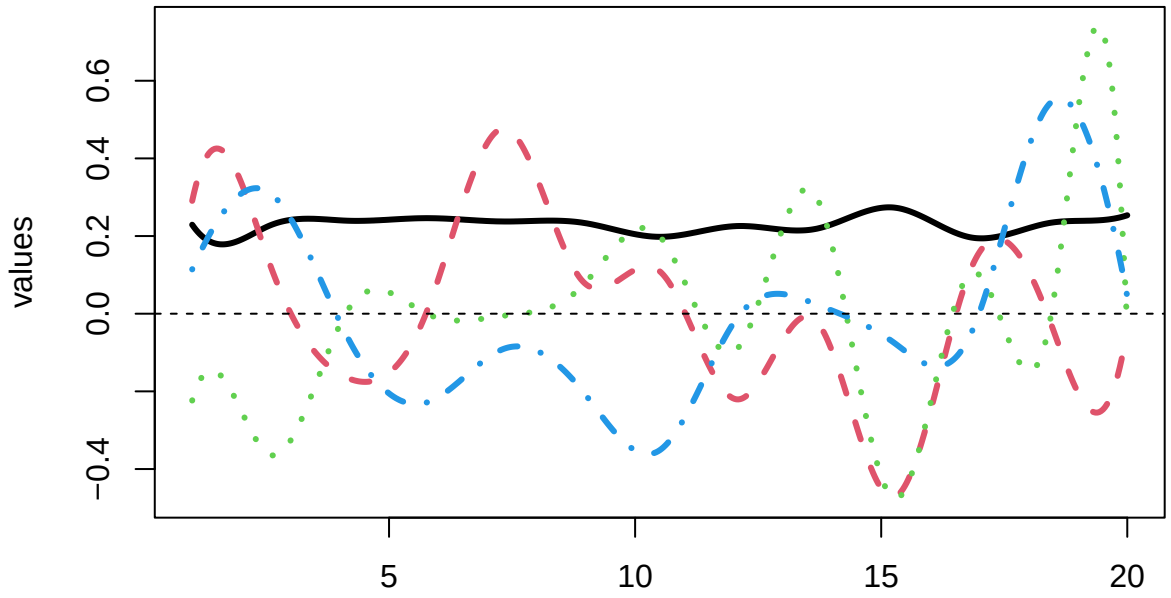


```
contour(grid,grid,cov.mat.pinch)
```



(d) Graph the first four EFPC's of the pinch data. How many components do you need to explain 90% of variation?

```
pinch.pca <- pca.fd(pinch.f, nharm=4)
plot(pinch.pca$harmonics, lwd=3)
```




```
## [1] "done"
```

```
pinch.pca$varprop # only one component needed to explain 90% of variation
```

```
## [1] 0.9868345469 0.0105586605 0.0008747298 0.0004368489
```

1.2

United states Federal Reserve interest rates `FedYiedlcurve`, which contains the monthly interest rates from January 1982 to June 2009

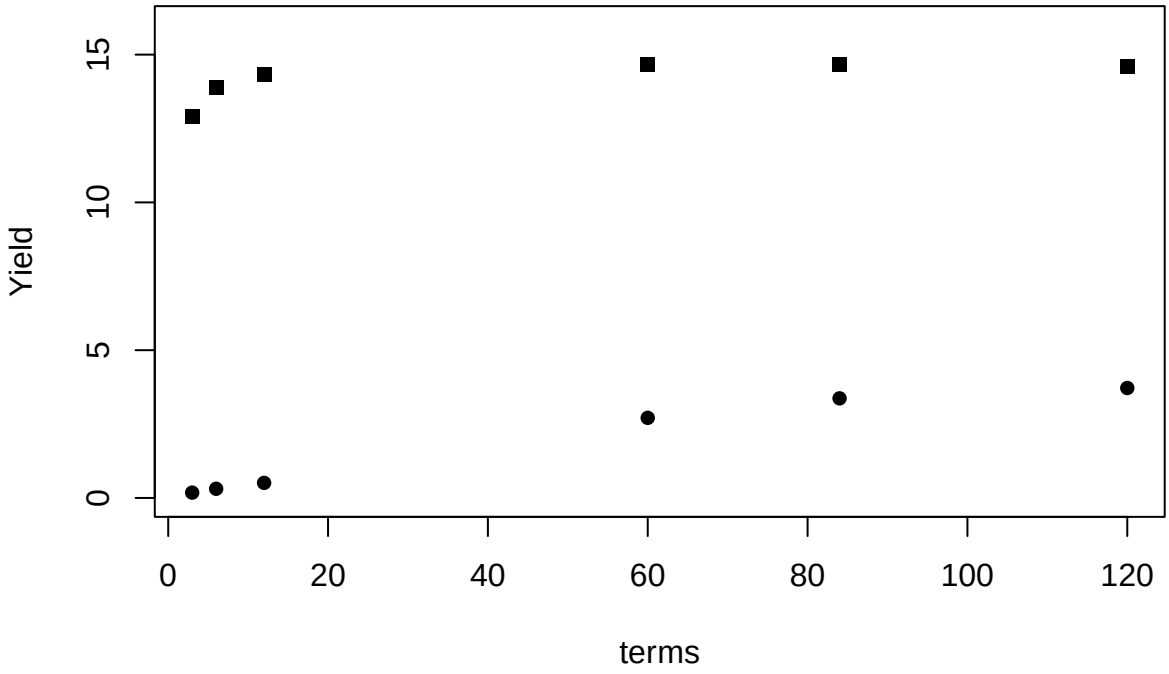
x values : maturity terms of 3,6,12,60,84,120 months. t_j

y values : interest rates of the U.S. Treasury obligations due in x months. $x_n(t_j)$, where n is a month in. the range 1982.1 to 2009.6

```
rm(list=ls())
library(fds);library(fda)
```

(a) On one graph, plot the interest rate $x(t_j)$ for Jan 1982 - Jun 2009 against the maturity terms t_j. How do the interest rates in these 2 months compare?

```
data(FedYieldcurve)
yield <- FedYieldcurve ; terms = yield$x
plot(terms, yield$y[,1], pch=15, ylab='Yield', ylim=c(0,16)) # square
points(terms, yield$y[,330], pch=16) # circle . lower interest rates at June 2009
```



(b) Convert the yield data to functional objects using bspline basis with 4 basis functions. Calculate and plot the mean yield function. What is the average behavior of interest rates as function of maturity?

```
max(yield$y)
```

```
## [1] 14.81
```

```
min(yield$y)
```

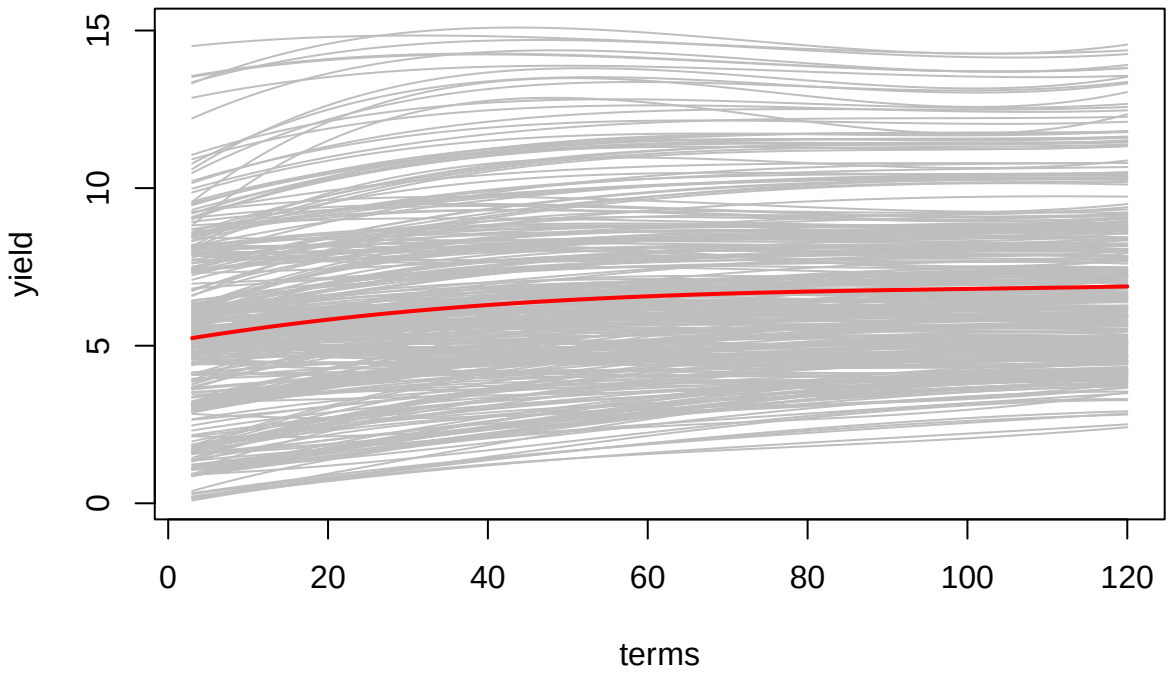
```
## [1] 0.03
```

```
M <- dim(yield$y)[1] ; N <- dim(yield$y)[2] # 6 330

bspline_basis <- create.bspline.basis(range=c(3,120), nbasis=4)
yield.fd <- Data2fd(terms,yield$y,basisobj = bspline_basis)
```

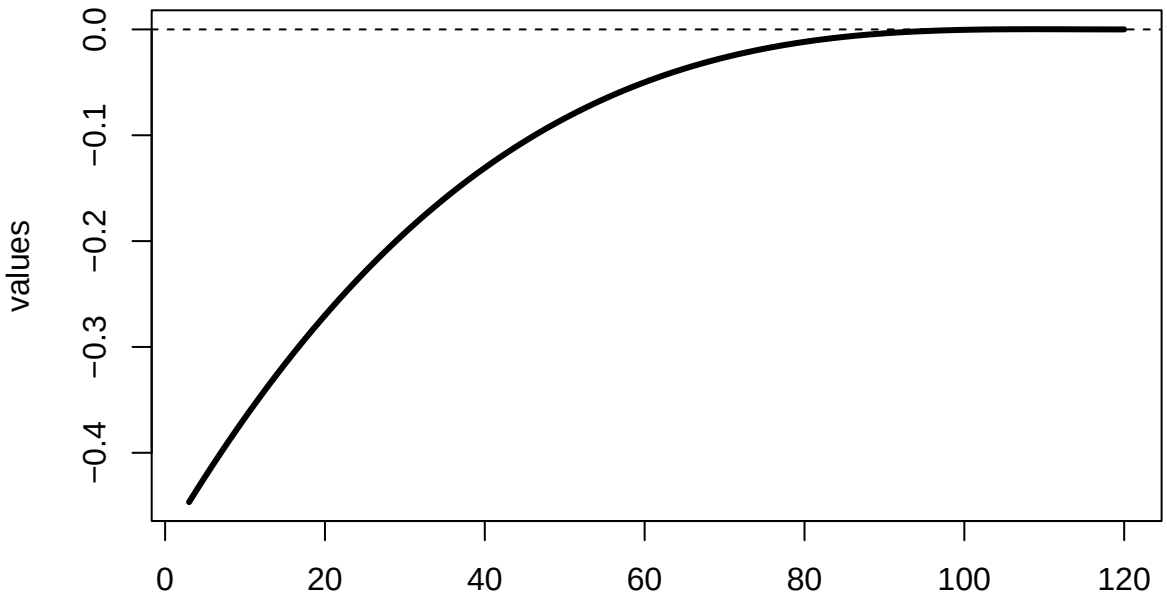
```
plot(yield.fd, lty=1, col='gray', xlab='terms', ylab='yield')
```

```
## [1] "done"
```

```
yield.mean <- mean.fd(yield.fd)
lines(yield.mean,lwd=2,col='red')
```




```
yield.pca <- pca.fd(yield.fd, nharm=1)
plot(yield.pca$harmonics,lwd=3)
```



(c)

```
## [1] "done"
```

```
yield.pca$varprop
```

```
## [1] 0.9999876
```

first component explains a lot of patterns of the deviation from the mean functions of yield.

1.3

Data for monthly SST(in degrees C) in various regions of the south Pacific ocean from 1950 to 2013. Using the `NINO3` column, treat data for each calander year as a functional observation.

```
rm(list=ls())
?sd.fd
```

```
nino <- read.table('ninoSST.txt')
head(nino)
```

```
##     V1    V2    V3    V4    V5    V6    V7    V8    V9   V10   V11   V12   V13
## 1 1950 23.56 24.89 26.36 26.44 25.69 25.51 24.91 24.47 24.01 24.17 23.86 24.15
## 2 1951 24.79 25.65 26.87 27.37 27.07 26.66 26.42 25.52 25.36 25.58 25.91 25.84
## 3 1952 25.80 26.37 27.05 27.27 26.64 25.70 24.82 24.44 24.26 24.36 24.07 24.67
## 4 1953 25.72 26.63 27.52 27.72 27.37 26.88 25.74 25.06 25.19 25.10 25.15 25.43
## 5 1954 25.50 26.27 26.56 25.61 25.62 25.04 24.35 23.56 23.17 23.32 23.64 23.68
## 6 1955 24.80 25.70 26.30 26.27 25.43 24.93 24.40 23.66 23.22 22.58 22.71 23.14
```

```
library(fda)
year <- nino[,1]
month <- 1:12

#SSTmat <- matrix(data=NA, nrow=12, ncol=64)
#count=1
#for(yr in year){
#  tmp <- nino[which(nino$V1==yr),]$V4
#  SSTmat[,count] <- tmp
#  count <- count+1
#}

SSTmat <- as.matrix(t(nino[,-1]))
rownames(SSTmat) = NULL
colnames(SSTmat) <- seq(1950,2013)
dim(SSTmat) # month, 1950-2013 years
```

```
## [1] 12 64
```

```
head(SSTmat)
```

```
##       1950  1951  1952  1953  1954  1955  1956  1957  1958  1959  1960  1961
## [1,] 23.56 24.79 25.80 25.72 25.50 24.80 24.60 24.94 26.96 25.58 25.31 25.11
## [2,] 24.89 25.65 26.37 26.63 26.27 25.70 25.70 26.20 27.44 26.35 25.93 26.20
## [3,] 26.36 26.87 27.05 27.52 26.56 26.30 26.67 27.47 27.72 27.07 26.87 26.82
## [4,] 26.44 27.37 27.27 27.72 25.61 26.27 26.92 27.86 27.81 27.45 27.15 27.31
## [5,] 25.69 27.07 26.64 27.37 25.62 25.43 26.49 27.79 27.29 26.84 26.71 26.99
## [6,] 25.51 26.66 25.70 26.88 25.04 24.93 25.77 27.31 26.57 26.07 25.86 26.29
##       1962  1963  1964  1965  1966  1967  1968  1969  1970  1971  1972  1973
## [1,] 25.09 24.85 26.02 24.70 26.69 24.93 24.26 26.11 26.08 24.08 24.93 27.44
## [2,] 25.97 25.79 26.31 25.86 26.90 25.91 24.83 26.72 26.41 24.84 25.88 27.20
## [3,] 26.47 26.91 26.72 26.93 27.18 26.30 25.76 27.42 26.83 25.90 26.91 27.29
## [4,] 26.47 27.43 26.19 27.47 27.30 26.35 26.49 27.76 27.04 26.54 27.58 26.75
## [5,] 26.17 26.97 25.42 27.56 26.20 26.59 26.04 27.71 26.27 26.13 27.46 25.93
## [6,] 25.67 26.58 24.99 27.09 25.89 26.24 26.19 26.85 25.16 25.38 27.23 25.21
##       1974  1975  1976  1977  1978  1979  1980  1981  1982  1983  1984  1985
## [1,] 23.86 25.12 23.76 26.54 25.79 25.34 26.04 24.95 25.86 28.66 25.08 24.40
## [2,] 25.07 25.56 25.30 26.96 26.41 26.12 26.44 25.52 26.40 28.71 26.26 25.50
```

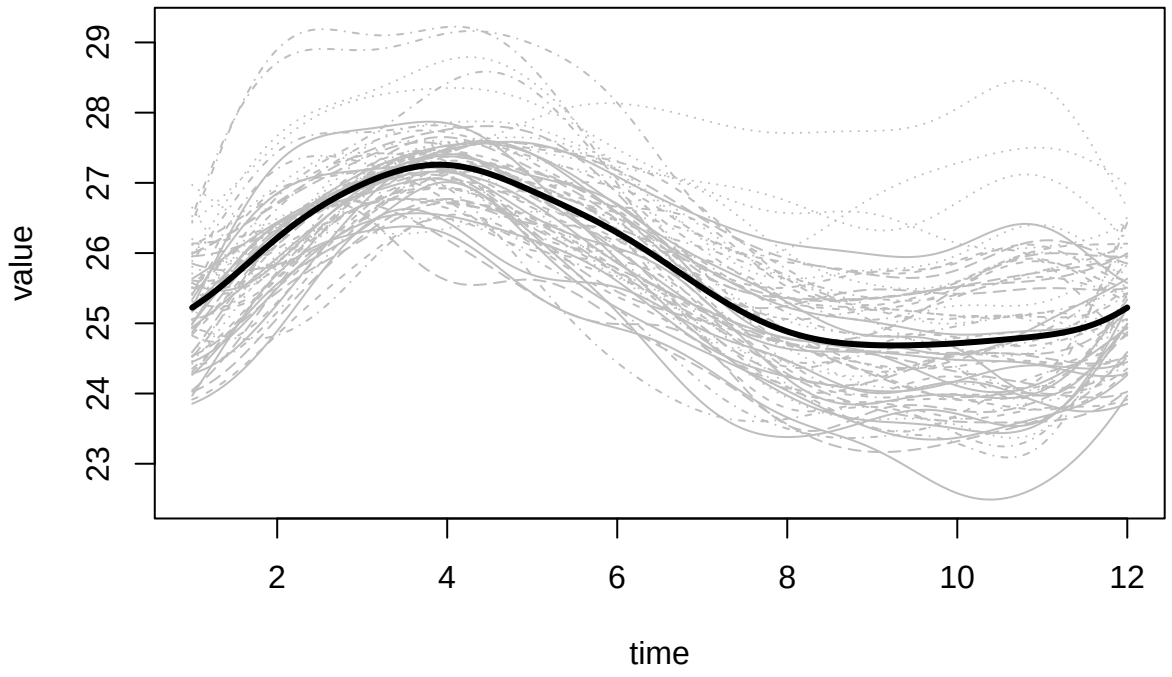
```
## [3,] 26.24 26.34 26.53 27.42 26.75 27.15 27.03 26.61 27.11 28.89 26.97 26.42
## [4,] 26.77 27.01 26.92 26.94 26.64 27.65 27.38 27.06 27.57 29.13 27.09 26.53
## [5,] 26.36 26.16 26.91 26.87 26.10 27.20 27.10 26.71 27.64 28.98 26.32 26.19
## [6,] 25.78 25.22 26.94 26.43 25.65 26.47 26.70 26.19 27.16 28.14 25.45 25.60
##       1986  1987  1988  1989  1990  1991  1992  1993  1994  1995  1996  1997
## [1,] 24.86 26.68 26.34 24.22 25.37 25.79 27.03 25.55 25.86 26.43 24.85 24.77
## [2,] 25.88 27.55 26.40 25.44 26.42 26.41 27.67 26.80 26.33 26.85 25.73 25.73
## [3,] 26.83 28.20 27.06 26.19 26.92 27.04 28.26 27.61 26.85 27.18 26.67 26.86
## [4,] 27.20 28.35 26.93 26.72 27.49 27.53 28.75 28.42 27.20 27.18 26.75 27.47
## [5,] 26.65 28.14 25.68 26.42 27.41 27.49 28.44 28.33 27.15 26.53 26.51 27.90
## [6,] 26.17 27.48 24.44 26.30 26.58 27.30 27.17 27.06 26.56 26.07 25.86 28.13
##       1998  1999  2000  2001  2002  2003  2004  2005  2006  2007  2008  2009
## [1,] 28.74 24.41 24.00 25.24 25.17 26.23 26.08 25.98 24.85 26.53 24.19 25.15
## [2,] 28.90 25.52 25.27 26.12 26.16 26.84 26.70 26.31 26.06 26.49 25.16 25.84
## [3,] 29.12 26.68 26.70 27.20 27.35 27.39 27.34 27.14 26.65 26.80 26.45 26.49
## [4,] 29.22 26.71 27.41 27.46 27.58 27.14 27.52 27.53 27.16 26.95 27.12 27.41
## [5,] 28.64 26.45 26.88 27.11 27.51 26.29 26.80 27.41 27.17 26.37 27.00 27.54
## [6,] 26.99 25.72 26.03 26.34 27.19 26.03 26.35 26.68 26.61 25.90 26.46 27.18
##       2010  2011  2012  2013
## [1,] 26.73 24.40 25.09 25.00
## [2,] 27.28 25.63 26.22 25.84
## [3,] 27.76 26.47 27.02 27.08
## [4,] 27.85 27.25 27.56 27.22
## [5,] 27.08 26.96 27.34 26.43
## [6,] 25.91 26.50 27.05 25.73
```

(a) Using Fourier basis functions, convert the data into a functional object `ninofd` containing 64 annuals curves and plot these smoothed curves.

```
N <- dim(SSTmat)[2] # 64
M <- dim(SSTmat)[1] # 12
argvals <- seq(1,12)
fourier.basis = create.fourier.basis(rangeval=c(1,12),
                                     nbasis=M)
ninofd <- Data2fd(argvals, SSTmat, fourier.basis)
plot(ninofd, col='gray')
```

```
## [1] "done"
```

```
lines(mean.fd(ninofd),col='black',lwd=3)
```



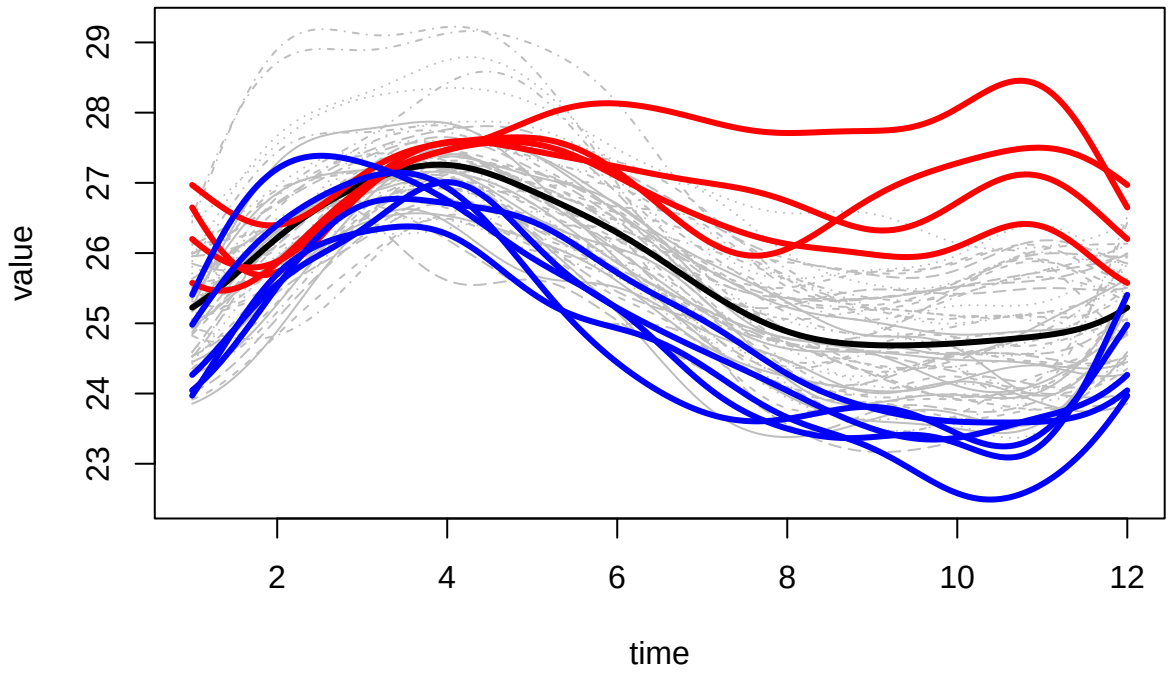
(b) The years of 1965, 1972, 1982, 1997 were pronounced El Nino years. emphasize the curves in the plot by making them thick red.

Emphasize the pronounced La Nina years of 1955, 1973, 1975, 1988, 1999 with thick blue.

```
#Pronounced ElNino Years - characterized by unusually warm SST
eln = c(1965, 1972, 1982, 1997)
#Pronounced La Nina Years - characterized by unusually cold SST
lan = c(1955, 1973, 1975, 1988, 1999)

elnn = match(eln,year)
lann = match(lan,year)
ELNmat = SSTmat[,elnn]
LANmat = SSTmat[,lann]

ELNfd <- Data2fd(argvals, ELNmat, fourier.basis)
LANfd <- Data2fd(argvals, LANmat, fourier.basis)
plot(ninofd, col='gray')
```

```
## [1] "done"
```

```
lines(mean.fd(ninofd),col='black',lwd=3)
lines(ELNfd,col='red',lwd=3,lty=1)
lines(LANfd,col='blue',lwd=3,lty=1)
```


value
23
24
25
26
27
28
29
2
4
6
8
10
12
time

1.4

1.5

Matern covariance function leads to general family of stationary Gaussian processes.

- covariance function

$$C(t,s) = \frac{\sigma^2}{\Gamma(\nu)2^{\nu-1}}(\frac{\sqrt{2\nu}|t-s|}{\rho})^{\nu} K_{\nu}(\frac{\sqrt{2\nu}|t-s|}{\rho}), \quad \nu > 0$$

σ^2 : variance parameter
ν : smoothness parameter.
ρ : range parameter
K_{ν} : modified Bessel function of the second kind.

k times continuously differentiable for any ν.

(1) Simulate plot iid mean zero Marten process with $\nu = 1/2$, $\nu = 2$, $\nu = 4$. Set $\sigma^2 = 1$ and $\rho = 1$. temporal grid with 50 evenly spaced points.

(2) Plot the first four EFPCS for each value of ν, comment on any similarities / differences

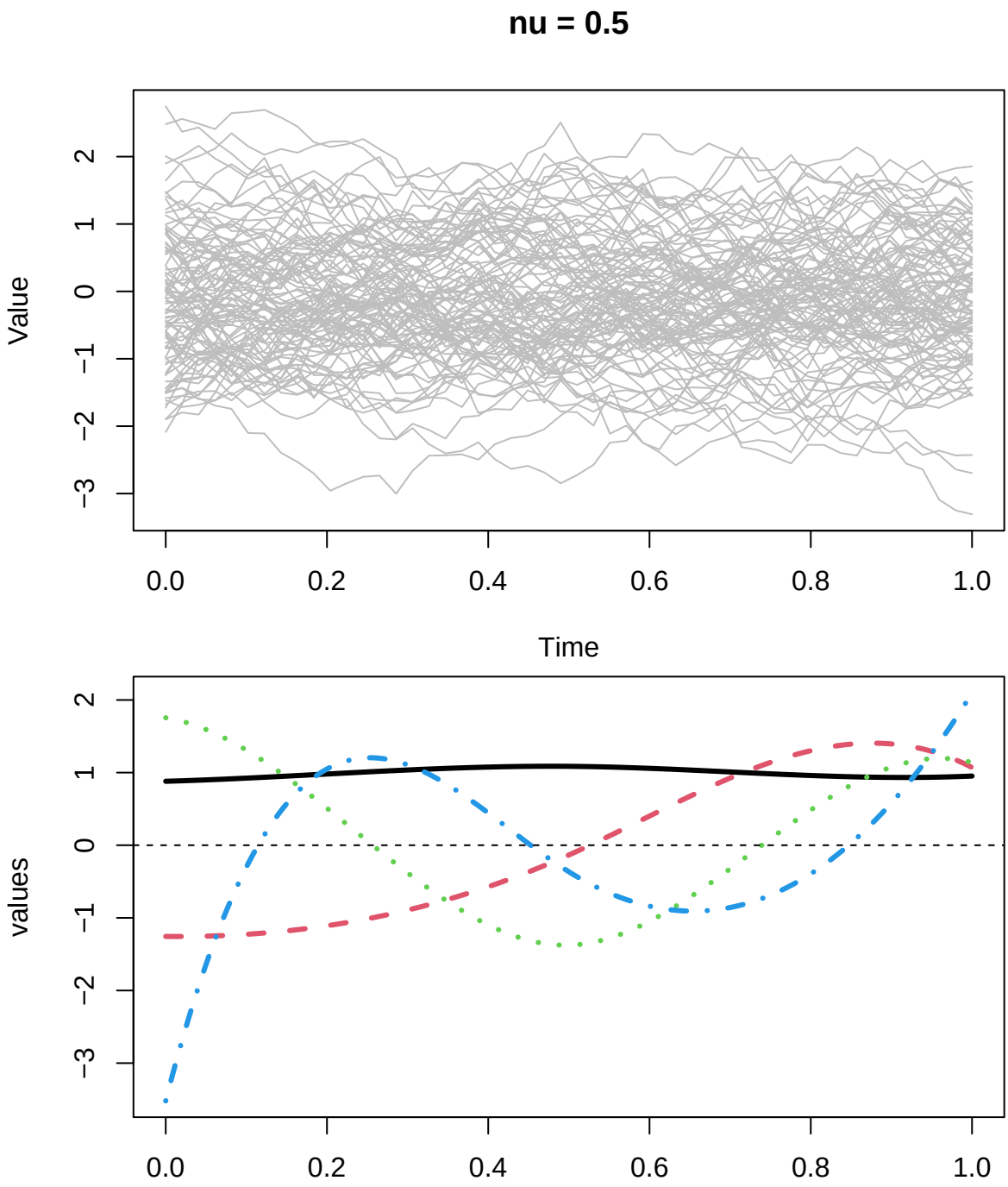
(3) Plot the explained variance for each ν, comment on any similarities/differences.

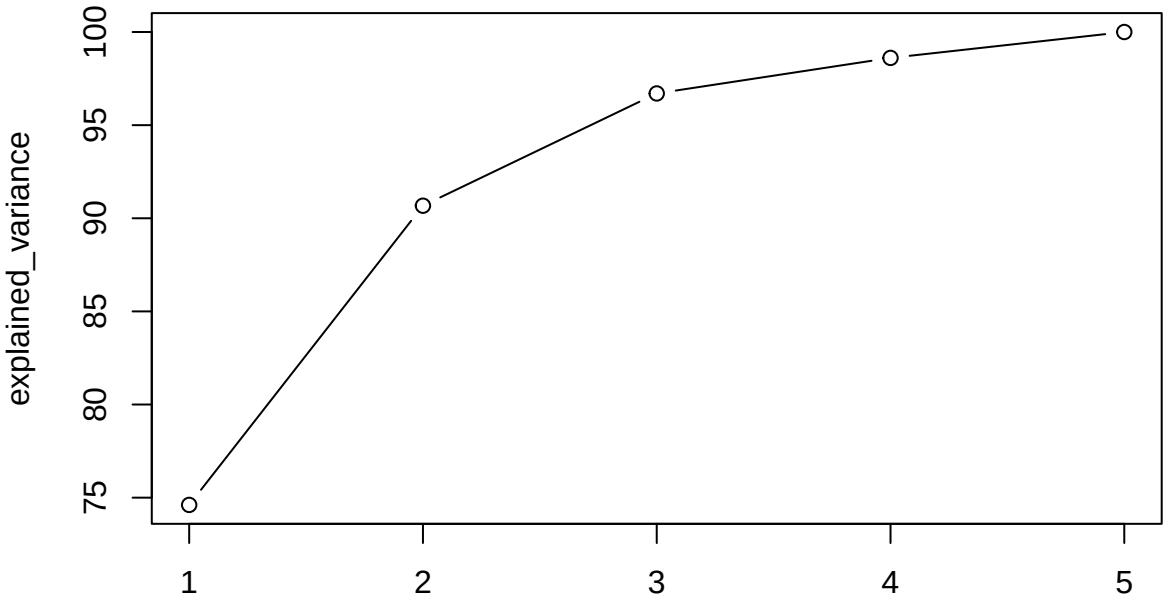
(4) Using your preferred method, create a plot of the covariance surface for each ν, comment on any similarities/differences.

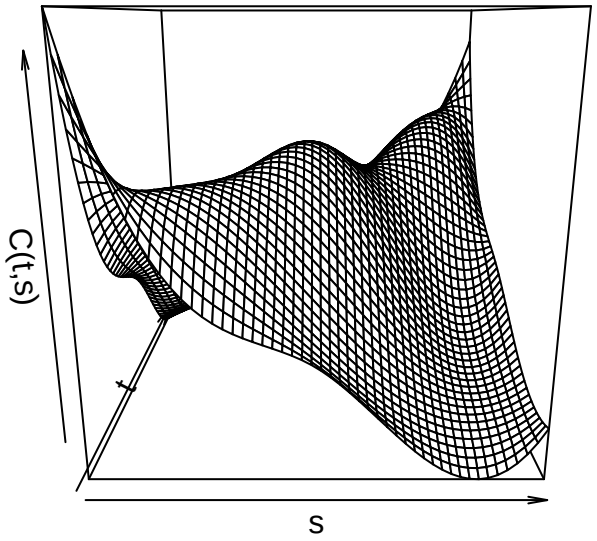
```
matern_cov <- function(t,s,sigma2,nu,rho){
  if(t==s){
    return(sigma2)
  } else{
  factor1 <- sigma2 / (gamma(nu) * 2^(nu-1))
  factor2 <- (sqrt(2*nu)*abs(t-s) / rho)^nu
  factor3 <- besselK(sqrt(2*nu)*abs(t-s)/rho, nu)
  return(factor1*factor2*factor3)
  }
} # each element in covariance

# parameters
sigma2 <- 1
rho <- 1
nu_values <- c(0.5,2,4)
time_points <- seq(0, 1, length.out = 50)

# create 50x50 matern_cov matrix
create_cov_mat <- function(time_points, sigma2, nu, rho){
  n <- length(time_points)
  cov_mat <- matrix(0,n,n)
  for(i in 1:n){
    for(j in 1:n){
      cov_mat[i,j] <- matern_cov(time_points[i], time_points[j],sigma2, nu, rho)
    }
  }
  return(cov_mat)
}

# simulation functions
```

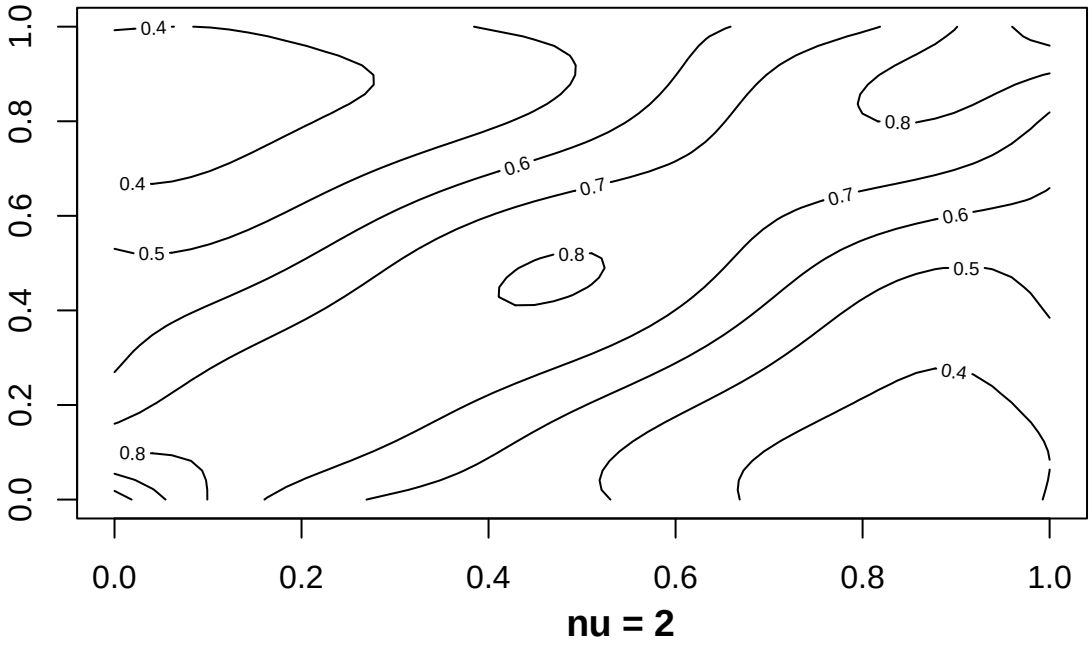
```r
simul <- function(n_sim, time_points, sigma2, nu, rho){
  n <- length(time_points)
  cov_mat <- create_cov_mat(time_points, sigma2, nu, rho)
  simulation <- mvrnorm(n_sim, mu=rep(0,n), Sigma = cov_mat)
  return(simulation)
}
```

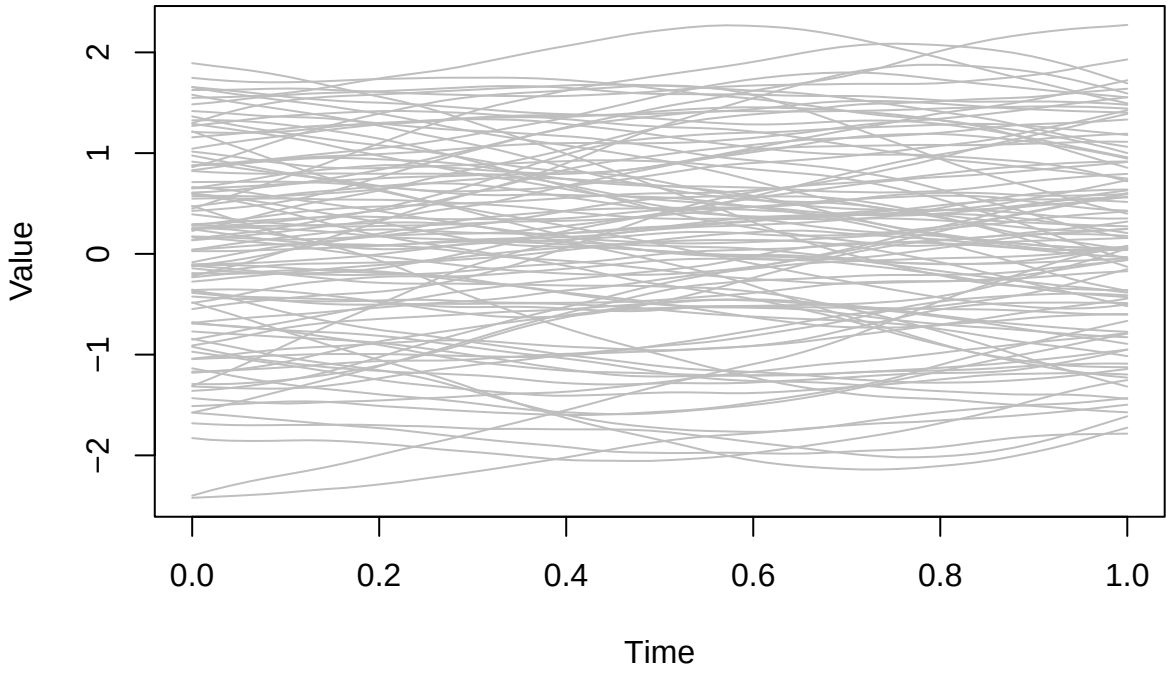
```r
set.seed(1)
# Simulation 100 times

for(nu in nu_values){
  simul_result <- simul(100, time_points, sigma2,nu,rho)
  # (1)
  # fda package
  matplot(time_points,t(simul_result),type='l',col='gray',lty=1,
          xlab='Time',ylab='Value',main=paste('nu =',nu))

  # (2)
  basis <- create.bspline.basis(rangeval=c(0,1), nbasis=5)
  fd_obj <- Data2fd(time_points,t(simul_result), basis)
  pca_result <- pca.fd(fd_obj,nharm=4)
  plot(pca_result$harmonics,lwd=3)

  # (3)
  explained_variance <- cumsum(pca_result$values)/sum(pca_result$values)*100
  plot(explained_variance,type='b')


  # (4)
  fd_cov <- var.fd(fd_obj)
  grid <- seq(0,1,length=50)
  fd_cov_mat <- eval.bifd(grid,grid,fd_cov)
  persp(grid,grid,fd_cov_mat,xlab='s',ylab='t',zlab='C(t,s)')
  contour(grid,grid,fd_cov_mat)

}
```

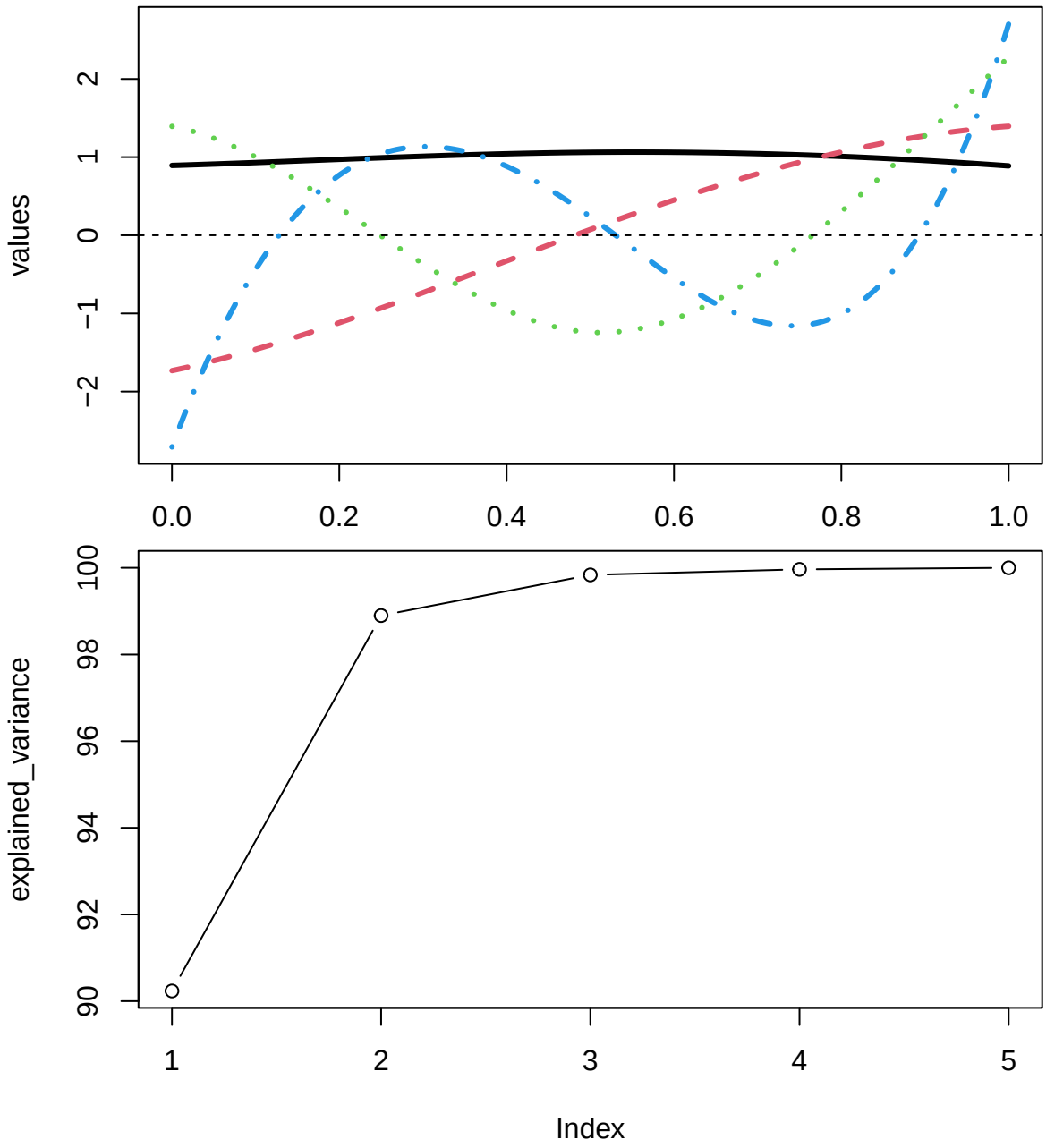

nu = 0.5
Value
2
1
0
−1
−2
−3
0.0
0.2
0.4
0.6
0.8
1.0
Time
values
2
1
0
−1
−2
−3
0.0
0.2
0.4
0.6
0.8
1.0

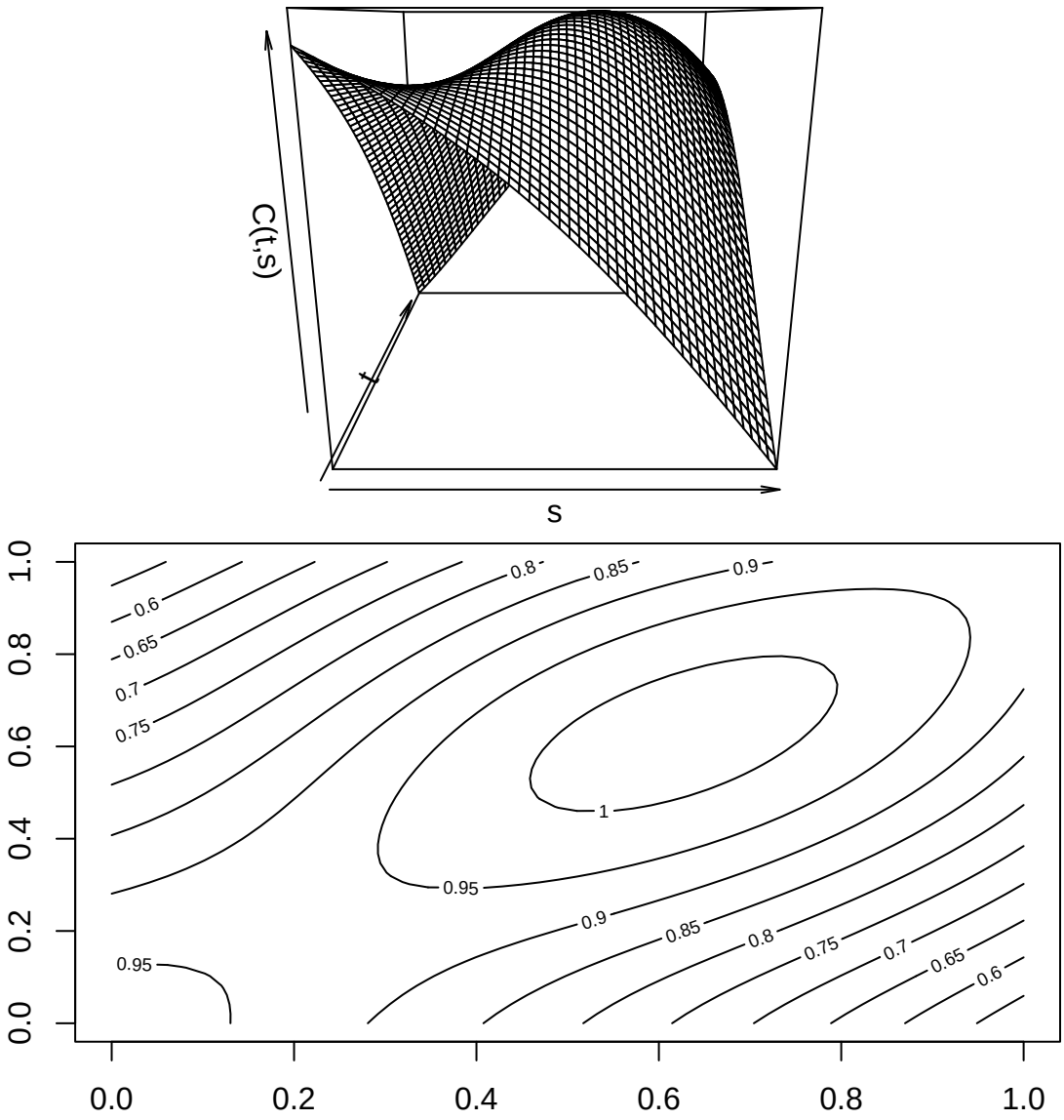

explained_variance
100
95
90
85
80
75
1
2
3
4
5
Index

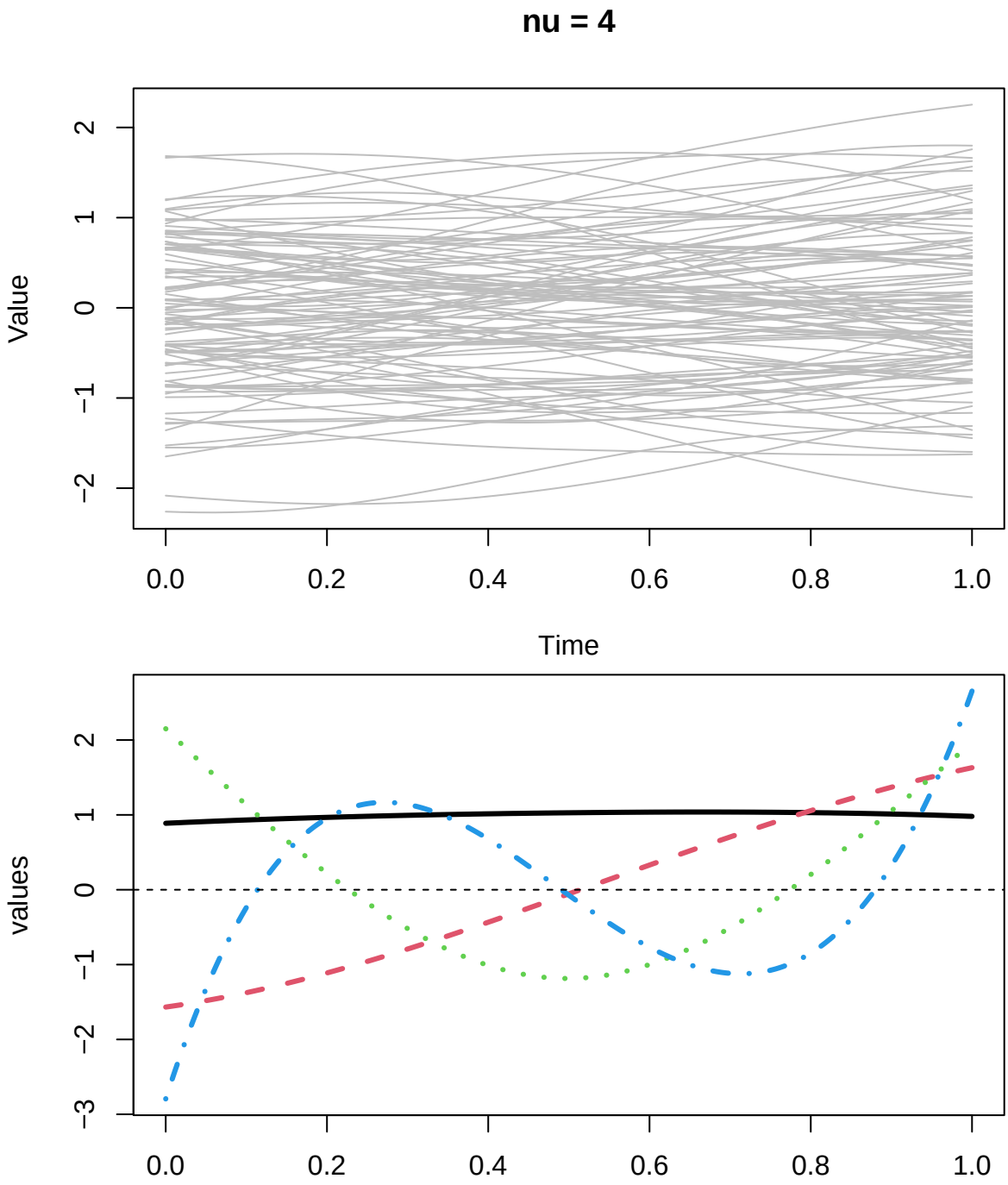

C(t,s)
t
s

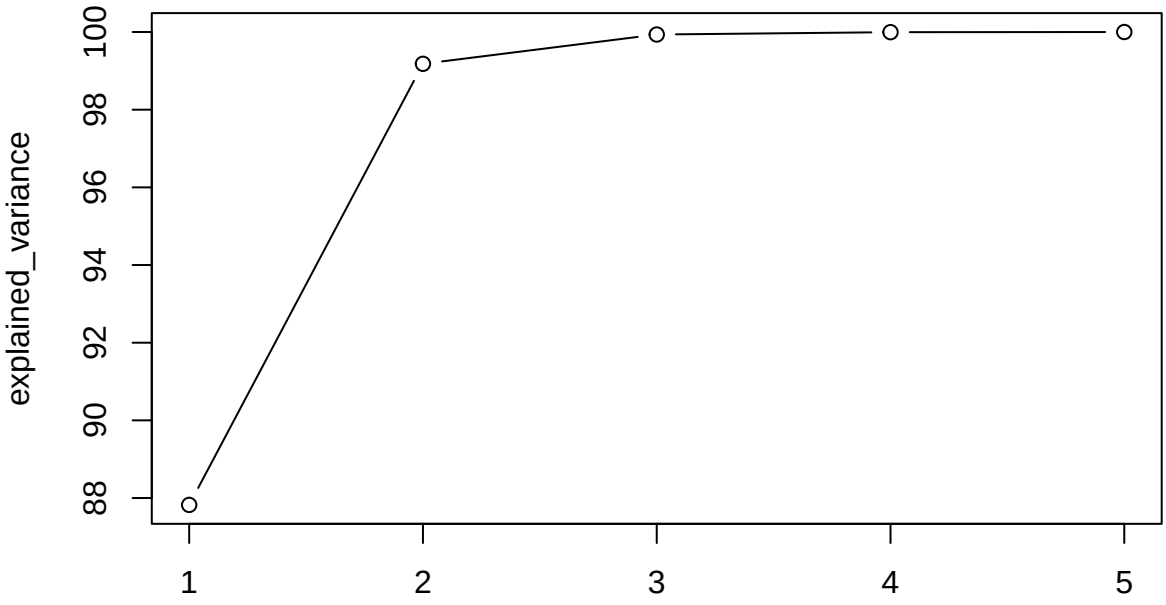

0.4
0.4
0.5
0.8
0.6
0.7
0.8
0.7
0.6
0.5
0.4
0.8
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
nu = 2

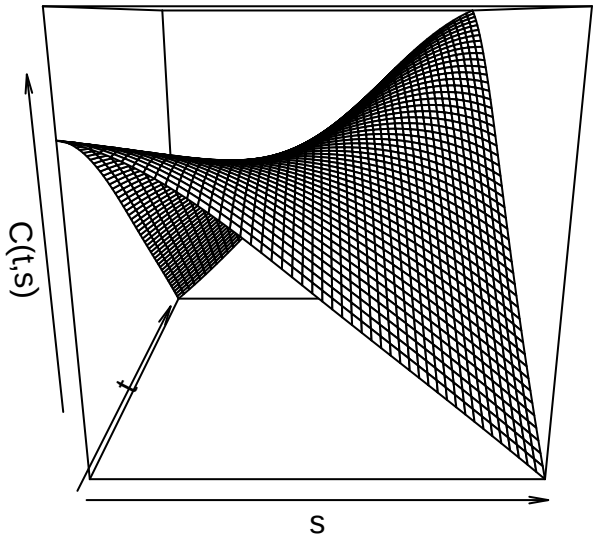

Value
2
1
0
−1
−2
0.0
0.2
0.4
0.6
0.8
1.0
Time

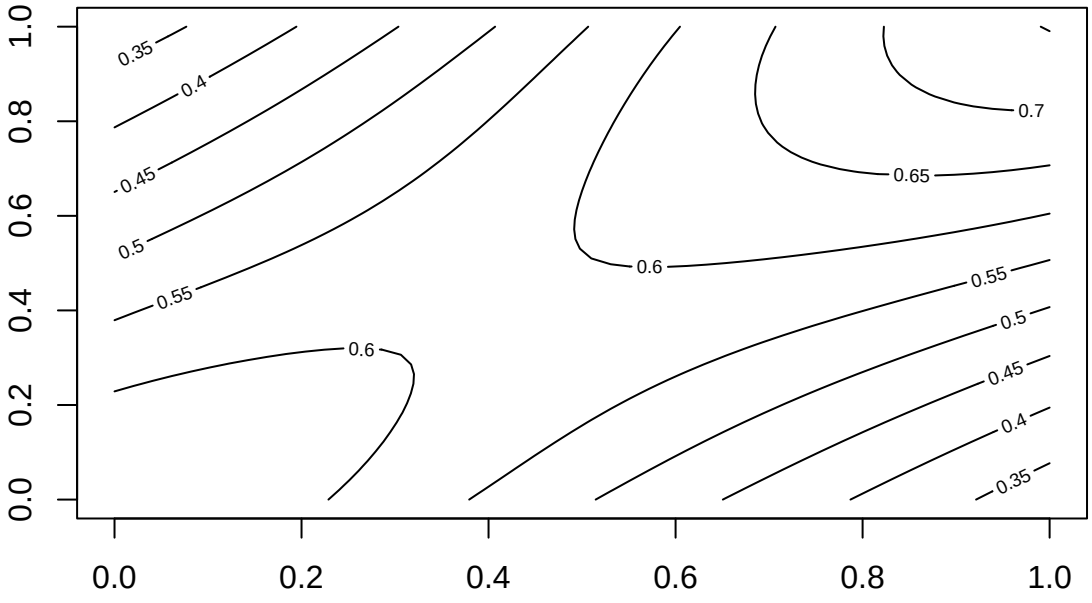

values
2
1
0
−1
−2
0.0
0.2
0.4
0.6
0.8
1.0
explained_variance
100
98
96
94
92
90
1
2
3
4
5
Index


C(t,s)
t
s
0.0
0.2
0.4
0.6
0.8
1.0
0.6
0.65
0.7
0.75
0.8
0.85
0.9
1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.95


nu = 4
Value
2
1
0
−1
−2
0.0
0.2
0.4
0.6
0.8
1.0
Time
values
2
1
0
−1
−2
−3
0.0
0.2
0.4
0.6
0.8
1.0


explained_variance
88
90
92
94
96
98
100
1
2
3
4
5
Index


C(t,s)
t
s


0.35
0.4
0.45
0.5
0.55
0.6
0.6
0.65
0.7
0.55
0.5
0.45
0.4
0.35
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0

(GPT code..)

```r
# Load necessary library for matrix operations
library(MASS)

# Define the covariance function for the Matern process
matern_cov <- function(h, nu, sigma2, rho) {
  if (h == 0) {
    return(sigma2)
  } else {
    factor1 <- (2^(1 - nu)) / gamma(nu)
    factor2 <- (sqrt(2 * nu) * h / rho) ^ nu
    factor3 <- besselK(sqrt(2 * nu) * h / rho, nu)
    return(sigma2 * factor1 * factor2 * factor3)
  }
}

# Parameters
sigma2 <- 1
rho <- 1
nu_values <- c(0.5, 2, 4)
time_points <- seq(0, 1, length.out = 50)

# Function to create covariance matrix
create_cov_matrix <- function(time_points, nu, sigma2, rho) {
  n <- length(time_points)
  cov_matrix <- matrix(0, n, n)
  for (i in 1:n) {
    for (j in 1:n) {
      cov_matrix[i, j] <- matern_cov(abs(time_points[i] - time_points[j]), nu, sigma2, rho)
    }
  }
  return(cov_matrix)
}

# Simulation function
simulate_matern_process <- function(n_sim, time_points, nu, sigma2, rho) {
  n <- length(time_points)
  cov_matrix <- create_cov_matrix(time_points, nu, sigma2, rho)
  simulations <- mvrnorm(n_sim, mu = rep(0, n), Sigma = cov_matrix)
  return(simulations)
}

# Set seed for reproducibility
set.seed(123)

# Plotting function
plot_simulations <- function(simulations, time_points, nu) {
  matplot(time_points, t(simulations), type = 'l', col = 'gray', lty = 1,
          main = paste('Matérn Process Simulations (nu =', nu, ')'),
          xlab = 'Time', ylab = 'Value')
  grid()
}
```

```r
# Simulate and plot for each nu value
par(mfrow = c(3, 1)) # 3 plots in one column
for (nu in nu_values) {
  simulations <- simulate_matern_process(100, time_points, nu, sigma2, rho)
  plot_simulations(simulations, time_points, nu)
}
```