#### code for Figure: The response of different K

```r
filterweight=function(w,h){
  if(h==0){
    return(w/pi)
  }else{
    return(sin(h*w)/(h*pi))
  }
}

frequency_response_function2=function(w,K){
  aw=0
  for (h in -K:K) {
    b_h=filterweight(2*pi/32,h)
    theta=1/(2*K+1)
    for (h1 in -K:K) {
      theta=theta-filterweight(2*pi/32,h1)/(2*K+1)
    }
    a_h=b_h+theta
    aw=aw+a_h*exp(-1i*w*h)
  }
  aw2=0
  for (h in -K:K) {
    b_h2=filterweight(2*pi/6,h)
    theta2=1/(2*K+1)
    for (h1 in -K:K) {
      theta2=theta2-filterweight(2*pi/6,h1)/(2*K+1)
    }
    a_h2=b_h2+theta2
    aw2=aw2+a_h2*exp(-1i*w*h)
  }
  return(aw2-aw)
}


a=seq(from=0,to=pi,length.out=100)
par(mfrow=c(2,2),mar=c(3,3,2,1),cex=.8)
plot(a,frequency_response_function2(a,4),ylab = 'frequency response',xlab= 'w',main="Truncated Filter K=4")
abline(h=1,lty=3)
abline(v=2*pi/32,lty=3)
abline(v=2*pi/6,lty=3)
abline(h=0,lty=3)
plot(a,frequency_response_function2(a,8),ylab = 'frequency response',xlab= 'w',main="Truncated Filter K=8")
abline(h=1,lty=3)
abline(v=2*pi/32,lty=3)
abline(v=2*pi/6,lty=3)
abline(h=0,lty=3)
plot(a,frequency_response_function2(a,12),ylab = 'frequency response',xlab= 'w',main="Truncated Filter K=12")
abline(h=1,lty=3)
abline(v=2*pi/32,lty=3)
abline(v=2*pi/6,lty=3)
abline(h=0,lty=3)
plot(a,frequency_response_function2(a,16),ylab = 'frequency response',xlab= 'w',main="Truncated Filter K=16")
abline(h=1,lty=3)
abline(v=2*pi/32,lty=3)
abline(v=2*pi/6,lty=3)
abline(h=0,lty=3)

```

#### code for Figure: BK(6,32) on real world data

```r
library("data.table")
library(mFilter)
GNP=as.data.frame(fread("data/data_new/GNP.csv",header = T))
GNP_data_log=ts(log(GNP$GNP[1:202]),frequency = 4,start = 1947)
GNP_data_log.bk=bkfilter(GNP_data_log,pl=6,pu=32,nfix = 12)
par(mfrow=c(2,1))
#plot(GNP_data_log,main="")
plot(GNP_data_log.bk$trend,
     main="Trending term form BK filter vs raw data , log(GNP)",
     col=2,lwd=3, ylab="",)
lines(GNP_data_log,col=1,lwd=1)
legend("bottomright",cex = 0.8,legend=c("BP12(6,32)", "raw data"), col=2:1,lwd=3:1)
plot(100*GNP_data_log.bk$cycle,
     main="cycle term from BK filter , log(GNP)",
     col=2,lwd=3, ylab="",)
nep=as.data.frame(fread("data/data_new/NetExport.csv",header = T))
nep_data_log=ts(nep$NETEXP[1:202],frequency = 4,start = 1947)
nep_data_log.bk=bkfilter(nep_data_log,pl=6,pu=32,nfix = 12)


par(mfrow=c(2,1))
plot(nep_data_log.bk$trend,
     main="Trending term form BK filter vs raw data , Net Export",
     col=2,lwd=3, ylab="",)
lines(nep_data_log,col=1,lwd=1)
```

~~~
legend("bottomleft",cex = 0.8,legend=c("BP12(6,32)", "raw data"), col=2:1,lwd=3:1)
plot(nep_data_log.bk$cycle,
     main="First Difference vs. Band-Pass filter , Net Export",
     col=2,lwd=3, ylab="",)
lines(diff(nep_data_log,differences = 1,lag = 1),col=1)
legend("topleft",cex = 0.6,legend=c("BP12(6,32)", "Diff1"), col=2:1, lty=rep(1,2), ncol=1)
```



#### code for Section 3: HP(1600) on real world data

```r
obj <- function(x,lambda=1500){
  lambda_vec = rep(lambda, length(x))
  lambda_vec[1]= lambda_vec[length(x)] =  3*lambda
  lambda_vec[2]= lambda_vec[length(x)-1] =  3/2*lambda
  y = sum((GNP_data_log-x)^2/lambda)  +  sum((diff(x, lag = 1, differences =2))^2)
  return(y)
}

GNP=as.data.frame(fread("/Users/xiaolongluo/Desktop/Programming_2021/TS/Project/data_new/GNP.csv",header = T))
GNP_data_log=ts(log(GNP$GNP[1:202]),frequency = 4,start = 1947)
GNP_data_log.bk=bkfilter(GNP_data_log,pl=6,pu=32,nfix = 12)

#result = optim(par = GNP_data_log, fn = obj_raw, method = "CG", lambda = 100)
#ts_result_raw = GNP_data_log -result$par

result = optim(par = GNP_data_log, fn = obj, method = "CG", lambda = 1600)
ts_result = GNP_data_log -result$par
plot(result$par)
plot(100*GNP_data_log.bk$cycle,
     main="Hodrick-Prescott vs. Band-Pass filter , log(GNP)",
     col=2,lwd=3, ylab="",)
lines(100*ts_result,col=4)
#lines(100*GNP_data_log.hp$cycle,col=4)
#lines(100*ts_result_raw,col=4)
legend("bottomright",cex = 0.8,legend=c("BP12(6,32)", "HP1600"), col=2:1, lty=rep(1,2), ncol=1)

png(file="seq1_lxl.png")
par(mfrow=c(2,1))
plot(result$par,
     main="Trending term form Modified HP filter vs raw data , log(GNP)",
     col=2,lwd=3, ylab="",)
lines(GNP_data_log,col=1,lwd=1)
legend("bottomright",cex = 0.8,legend=c("Modified HP (1600)", "raw data"), col=2:1,lwd=3:1)
plot(100*ts_result,
     main="cycle term from Modified HP filter, log(GNP)",
     col=2,lwd=3, ylab="",)
dev.off()


nep=as.data.frame(fread("/Users/xiaolongluo/Desktop/Programming_2021/TS/Project/data_new/NetExport.csv",header = T))
nep_data_log=ts(nep$NETEXP[1:202],frequency = 4,start = 1947)
nep_data_log.bk=bkfilter(nep_data_log,pl=6,pu=32,nfix = 12)

result = optim(par = nep_data_log, fn = obj, method = "CG", lambda = 1600)
ts_result = nep_data_log -result$par

png(file="seq4_lxl.png")
par(mfrow=c(2,1))
plot(result$par,
     main="Trending term form Modified HP filter vs raw data , Net Export",
     col=2,lwd=3, ylab="",)
lines(nep_data_log,col=1,lwd=1)
legend("bottomleft",cex = 0.8,legend=c("Modified HP (1600)", "raw data"), col=2:1,lwd=3:1)
plot(ts_result,
     main="Modified HP filter vs. Band-Pass filter , Net Export",
     col=2,lwd=3, ylab="",)
lines(nep_data_log.bk$cycle,col=1)
legend("topleft",cex = 0.6,legend=c("Modified HP (1600)", "BP12(6,32)"), col=2:1, lty=rep(1,2), ncol=1)
dev.off()
```



#### code for Section 4: Robust Filters

```r
---

```{r}
library(extRC)
library(MASS)
Soft_thre <- function(x,rho){
  ifelse(abs(x)<=rho,0,x-rho*sign(x))
}
terminate <- function(eps_abs,eps_rel,D,t,z,u,z0,rho,N,...){
  res_r <- norm(D%*%t-z,type="2")
  res_s <- rho * norm(t(D)%*%(z-z0),type="2")
  eps_pri <- sqrt(2*N-3)*eps_abs + eps_rel * max(norm(D%*%t,type = "2"),norm(z,type = "2"))
  eps_dual <- sqrt(N)%*%eps_abs + eps_rel * norm(rho*t(D)%*% u,type="2")
~~~

~~~
  return((res_r<eps_pri)&& (res_s < eps_dual))
}
huber_filter <- function(y,l1=1,l2=1,rho=0.1,gamma = 0.1,eps_abs = 5e-3,eps_rel = 5e-3,episode = 100,...){
  N <- length(y)
  t <- numeric(N)
  A <- matrix(data=0, ncol = N,nrow = N)

  D1 <- dfm(N)
  D2 <- -cbind(D1,matrix(0,ncol=1,nrow =N-1))+cbind(matrix(0,ncol=1,nrow =N-1),D1)
  D3 <- D2[-N+1,-N-1]
  D <- rbind(D1*l1,D3*l2)
  p <- nrow(D)
  u <- numeric(p)

  t <- mean(y)*rep(1,N)
  z <- D%*%t
  S <- rho * t(D)%*%D
  Dy <- D%*%y
  for (k in 1:episode){
    r <- y - t
    huber_rate <- ifelse(abs(r)>gamma, gamma*sign(r)/(r),1)
    A <- diag(as.vector(huber_rate))
    t <- y - rho * solve(A + S)%*%t(D)%*%(u-z+Dy)
    z0 <- z
    z <- Soft_thre(D%*%t + u,rho)
    u <- u + D%*%t - z
    if (terminate(eps_abs,eps_rel,D,t,z,u,z0,rho,N))
      break

  }
  return(t)
}

```

Simulation
```{r}
set.seed(123)
syn.data0 <- rep(0,500)
syn.data0[51:200] <- sin(1:150/75*pi)
syn.data0[201:250] <- 0.02*1:50
syn.data0[251:300] <- 1-0.02*1:50
syn.data0[351:400] <- -1
syn.data0[401:450] <- 1
plot(syn.data0,type = "l")
syn.data1 <- syn.data0 + rnorm(500,sd = 0.2)
plot(syn.data1,type = "l")
for (number in c(20)){
  syn.data <- syn.data1
  index <- sample(1:500,number)
  syn.data[index] <- syn.data[index] + rnorm(number,sd= 1)
  plot(syn.data,type = "l")
}
pdf("1-orign.pdf")
plot(syn.data0,type = "l",main = "Orignal Signal",ylab = "")
dev.off()
pdf("1-4per.pdf")
plot(syn.data,type = "l",main = "4 Percent Outliers",ylab = "")
dev.off()
```

```{r}
library(tvR)
set.seed(123)
number <- 20
l1s <- c(0.6,4,40)
l2s <-  c(0.4,4,40)
rhos <- c(0.8)
gams <- c(0.1,0.4,4,40)
syn.data <- syn.data1
  index <- sample(1:500,number)
syn.data[index] <- syn.data[index] + rnorm(number,sd= 2)
data_y <- syn.data
for (m in 1:length(rhos)){
  pre0_rb <- huber_filter(y = syn.data,l1=l1s[1],l2=l2s[1],rho=rhos[m],gamma =gams[1])
  plot(data_y,type = "l")
  lines(pre0_rb,col = "red")
  pre0_l1 <- huber_filter(y = syn.data,l1=1,l2=0,rho=rhos[m],gamma =Inf)
  plot(data_y,type = "l")
  lines(pre0_l1,col = "red")
  pre0_HP <- huber_filter(y = syn.data,l1=0,l2=1,rho=rhos[m],gamma =Inf)
  plot(data_y,type = "l")
  lines(pre0_HP,col = "red")
  pre0_TVD <- denoise1(syn.data, method = "TVL2.MM")
plot(data_y,type = "l",xlab = "Day",ylab = "",main = "RobustFilter",col = rgb(0, 0, 0, 160, maxColorValue=255))
lines(pre0_TVD,col = "red")
}

```

```{r}
library(tvR)
pdf("1-pred-rob.pdf")
~~~

~~~
#layout(matrix(1:4, 4, 1,byrow=T))
#pre0_rb <- huber_filter(y = syn.data,l1=0.6,l2=0.4,rho=0.8,gamma =0.1)
#pre0_l1 <- huber_filter(y = syn.data,l1=1,l2=0,rho=0.8,gamma =Inf)
#pre0_HP <- huber_filter(y = syn.data,l1=0,l2=1,rho=0.8,gamma =Inf)
#pre0_TVD <- denoise1(syn.data, method = "TVL2.MM")
plot(data_y,type = "l",ylab = "",main = "RobustFilter",col = rgb(0, 0, 0, 160, maxColorValue=255))
lines(pre0_rb,col = "red")
legend("bottomleft",col = c(rgb(0, 0, 0, 160, maxColorValue=255),"red"),legend = c("data","RobustFilter"),lty = c(1,1),cex=1)
dev.off()
pdf("1-pred-l1.pdf")
plot(data_y,type = "l",ylab = "",main = "l1 Filter",col = rgb(0, 0, 0, 160, maxColorValue=255))
lines(pre0_l1,col = "red")
legend("bottomleft",col = c(rgb(0, 0, 0, 160, maxColorValue=255),"red"),legend = c("data","l1 Filter"),lty = c(1,1),cex=1)
dev.off()
pdf("1-pred-HP.pdf")
plot(data_y,type = "l",ylab = "",main = "HP Filter",col = rgb(0, 0, 0, 160, maxColorValue=255))
lines(pre0_HP,col = "red")

legend("bottomleft",col = c(rgb(0, 0, 0, 160, maxColorValue=255),"red"),legend = c("data","HP Filter"),lty = c(1,1),cex=1)
dev.off()
pdf("1-pred-TVD.pdf")
plot(data_y,type = "l",ylab = "",main = "TVD",col = rgb(0, 0, 0, 160, maxColorValue=255))
lines(pre0_TVD,col = "red")
legend("bottomleft",col = c(rgb(0, 0, 0, 160, maxColorValue=255),"red"),legend = c("data","TVD"),lty = c(1,1),cex=1)
dev.off()
```



MSE/MAE Table

```{r}
cat("Filters\tMSE\tMAE")
library(tvR)
set.seed(123)
er <- function(pred){
  mse <- sum((pred- syn.data0)**2)/length(syn.data)
  mae <- sum(abs(pred- syn.data0))/length(syn.data)
  return(c(mse,mae))
}
for (number in c(20,50,100)){
  syn.data <- syn.data1
  index <- sample(1:500,number)
  syn.data[index] <- syn.data[index] + rnorm(number,sd= 2)
  pre0_rb <- huber_filter(y = syn.data,l1=0.6,l2=0.4,rho=0.8,gamma =0.1)
  pre0_l1 <- huber_filter(y = syn.data,l1=1,l2=0,rho=0.8,gamma =Inf)
  pre0_HP <- huber_filter(y = syn.data,l1=0,l2=1,rho=0.8,gamma =Inf)
  pre0_TVD <- denoise1(syn.data, method = "TVL2.MM")
  cat(number/5," Percent Outliers\n","RobustFilter",er(pre0_rb))
  cat("\nl1 Filter",er(pre0_l1))
  cat("\nHP Filter",er(pre0_HP))
  cat("\nTVD", er(pre0_TVD))
  cat("\n")
}

```

Covid-19 Data

```{r}
covid.italy <- read.csv("covid/italy.csv")$newcase
data_y <- covid.italy+1
data_y <- data_y[200:500]

pdf("2-11.pdf")
layout(matrix(1:2, 2, 1,byrow=T))
plot(covid.italy[200:500],type = "l",col = c(rgb(0, 0, 0, 250, maxColorValue=255)),,xlab = "Day",ylab = "New Cases",main = "Covid-19 New Cases Curve in Italy")
dev.off()
pdf("2-12.pdf")
plot(log(data_y),type = "l",col = c(rgb(0, 0, 0, 250, maxColorValue=255)),,xlab = "Day",ylab = "New Cases (Log Transformed)",main = "Covid-19 New Cases Curve in Italy")
dev.off()
```




```{r}
pdf("2-Robust.pdf")
library(tvR)
l1s <- c(0.2,4,40)
l2s <-  c(0.4,4,40)
rhos <- c(10)
gams <- c(0.1,0.4,4,40)
covid.italy <- read.csv("covid/italy.csv")$newcase
data_y <- covid.italy+1
data_y <- data_y[200:500]
data_y <- log(data_y)
layout(matrix(1:2, 2, 1,byrow=T))
for (m in 1:length(rhos)){
  pre_rob <- huber_filter(y = data_y,l1=l1s[1],l2=0.4,rho=10,gamma =0.1)
  plot(data_y,type = "l",xlab = "Day",ylab = "New Cases (Log Transformed)",col = rgb(0, 0, 0, 160, maxColorValue=255),main = "RobustFilter")
  lines(pre_rob,col = "red")
  legend("bottomleft",col = c(rgb(0, 0, 0, 160, maxColorValue=255),"red"),legend = c("data","RobustFilter"),lty = c(1,1),cex=1)
~~~

~~~
  plot(data_y - pre_rob,ylim = c(-10,3),col = "orange",type="l",xlab= "Day",ylab =" Residual( Involve Cycle)",main = "RobustFilter Residual")
  lines(rep(1,300)*sd(data_y - pre_rob),lty = 3,col = "red")
  lines(-rep(1,300)*sd(data_y - pre_rob),lty = 3,col = "red")
}
legend("bottomleft",col = c("orange","red"),legend = c("residua (include cycle)","3-SD Line"),lty = c(1,1),cex=1)
dev.off()
pre_rob <- huber_filter(y = data_y,l1=l1s[1],l2=0.4,rho=10,gamma =0.1)
  plot(data_y,type = "l",xlab = "Day",col = rgb(0, 0, 0, 160, maxColorValue=255),main = "RobustFilter")
  lines(pre_rob,col = "red")
  legend("bottomleft",col = c(rgb(0, 0, 0, 160, maxColorValue=255),"red"),legend = c("data","trend-Robust"),lty = c(1,1),cex=1)
  plot(data_y - pre_rob,ylim = c(-10,3),col = "orange",type="l",xlab= "Day",ylab =" Residual( Involve Cycle)")
  lines(rep(1,300)*sd(data_y - pre_rob),lty = 3,col = "red")
  lines(-rep(1,300)*sd(data_y - pre_rob),lty = 3,col = "red")
```

Residual Model
```{r}
set.seed(123)
library(forecast)
sdd <- sd(r)
r <- data_y - pre_rob
data_e <- ifelse(abs(r)>3*sdd,0,r)
model <- auto.arima(data_e)
model
Box.test(model$residuals)

```
```{r}
pdf("2-l1.pdf")
layout(matrix(1:2, 2, 1,byrow=T))
for (m in 1:length(rhos)){
  pre_l1 <- huber_filter(y = data_y,l1=1,l2=0,rho=5,gamma =Inf)
  plot(data_y,type = "l",xlab = "Day",main = "l1 Filter",col = rgb(0, 0, 0, 160, maxColorValue=255),ylab="")
  lines(pre_l1,col = "red")
  legend("bottomleft",col = c(rgb(0, 0, 0, 160, maxColorValue=255),"red"),legend = c("data","l1 Filter"),lty = c(1,1),cex=1)
  plot(data_y - pre_l1,ylim = c(-10,3),col = "orange",type="l",xlab= "Day",main = "l1 Filter Residual",ylab="")
  lines(rep(1,300)*sd(data_y - pre_l1),lty = 3,col = "red")
  lines(-rep(1,300)*sd(data_y - pre_l1),lty = 3,col = "red")
}
legend("bottomleft",col = c("orange","red"),legend = c("residua (include cycle)","3-SD Line"),lty = c(1,1),cex=1)
dev.off()
```

```{r}
pdf("2-HP.pdf")
layout(matrix(1:2, 2, 1,byrow=T))
for (m in 1:length(rhos)){
  pre_HP <- huber_filter(y = data_y,l1=0,l2=1,rho=5,gamma =Inf)
  plot(data_y,type = "l",xlab = "Day",main = "HP Filter",col = rgb(0, 0, 0, 160, maxColorValue=255),ylab = "New Cases (Log Transformed)")
  lines(pre_HP,col = "red")
  legend("bottomleft",col = c(rgb(0, 0, 0, 160, maxColorValue=255),"red"),legend = c("data","HP Filter"),lty = c(1,1),cex=1)
  plot(data_y - pre_HP,ylim = c(-10,3),col = "orange",type="l",xlab= "Day",ylab =" Residual( Include Cycle)",main = "HP Filter Residual")
  lines(rep(1,300)*sd(data_y - pre_HP),lty = 3,col = "red")
  lines(-rep(1,300)*sd(data_y - pre_HP),lty = 3,col = "red")
}
legend("bottomleft",col = c("orange","red"),legend = c("residua (include cycle)","3-SD Line"),lty = c(1,1),cex=1)
dev.off()
```

```{r}
pdf("2-TVD.pdf")
layout(matrix(1:2, 2, 1,byrow=T))
for (m in 1:length(rhos)){
  pre_TV <- denoise1(data_y, method = "TVL2.MM")
  plot(data_y,type = "l",xlab = "Day",main = "TVD Filter",col = rgb(0, 0, 0, 160, maxColorValue=255),ylab="")
  lines(pre_TV,col = "red")
  legend("bottomleft",col = c(rgb(0, 0, 0, 160, maxColorValue=255),"red"),legend = c("data","TVD Filter"),lty = c(1,1),cex=1)
  plot(data_y - pre_TV,ylim = c(-10,3),col = "orange",type="l",xlab= "Day",ylab ="",main = "TVD Filter Residual")
  lines(rep(1,300)*sd(data_y - pre_TV),lty = 3,col = "red")
  lines(-rep(1,300)*sd(data_y - pre_TV),lty = 3,col = "red")
}
legend("bottomleft",col = c("orange","red"),legend = c("residua (include cycle)","3-SD Line"),lty = c(1,1),cex=1)
dev.off()
```

```
~~~