```
#################################
##第十一回「ノンパラメトリック回帰」
#################################

#####################################
## old-faithful data
# eruptions: 間欠泉が噴き出るまでに掛かった時間（分）
# waiting: 次の吹き出しまでの時間
library(FNN)

# データの読み込み
oldfaithful.data <- read.table(file = "oldfaithfuldata.txt",header=TRUE)
attach(oldfaithful.data)

## Nadaraya-Watson推定量
# デフォルトの設定でNadaraya-Watson推定量 (h=0.5)
oldfaithful.reg.default <- ksmooth(eruptions, waiting, kernel="normal")
plot(eruptions, waiting); lines(oldfaithful.reg.default)

# a smaller bandwidth (h=0.1)
oldfaithful.reg.smallbw <- ksmooth(eruptions, waiting, kernel="normal", bandwidt
h=0.1);
lines(oldfaithful.reg.smallbw,col='2')  #良い推定結果とは言いにくい

# A larger bandwidth (h=0.2)
oldfaithful.reg.largebw <- ksmooth(eruptions, waiting, kernel="normal", bandwidt
h=2);
lines(oldfaithful.reg.largebw,col='3')  #悪くない

## lm, 線形回帰
oldfaithful.lm <- lm(waiting ~ eruptions)
plot(eruptions, waiting); lines(eruptions,predict(oldfaithful.lm))

## knn
# k=50
oldfaithful.knn <- knn.reg(eruptions,y=waiting,k=50)
sind = order(eruptions)
plot(eruptions, waiting); lines(eruptions[sind],oldfaithful.knn$pred[sind],col='
2')

# k=5
oldfaithful.knn.smallk <- knn.reg(eruptions,y=waiting,k=5)
lines(eruptions[sind],oldfaithful.knn.smallk$pred[sind],col='3')
legend(-3, 0.5, legend = c("k=50","k=5"), lty = c(1,1), col=c(2,3),title = "k")


# 各種手法を比較
plot(eruptions, waiting);
lines(oldfaithful.reg.default,col='2')
lines(oldfaithful.reg.largebw,col='3')
lines(eruptions,predict(oldfaithful.lm),col='4')
lines(eruptions[sind],oldfaithful.knn$pred[sind],col='5')


# estimating the test error
n <- length(eruptions)

err.ks.default <- c(1:10)
err.ks.lw <- c(1:10)
err.knn <- c(1:10)
err.lm < -c(1:10)


for(cv_iter in 1:10){
    indtrain <- sample(n,floor(0.8*n))
    train.erupt <- eruptions[indtrain]
    train.waiting <- waiting[indtrain]
    train = data.frame(erupt = eruptions[indtrain], waiting = waiting[indtrain])

    test = data.frame(erupt = eruptions[-indtrain], waiting = waiting[-indtrain]
)
```

```
    oldfaithful.lm <- lm(waiting ~ erupt, data=train)
    oldfaithful.lm.predict_new <- predict(oldfaithful.lm,test)
    err.lm[cv_iter] = mean((oldfaithful.lm.predict_new - test$waiting)^2)

    oldfaithful.reg.default.pred <- ksmooth(train$erupt,train$waiting, kernel="n
ormal", x.points = test$erupt)
    sind <- order(test$erupt) #ksmoothの予測はxの値を勝手にソートするので, 整列
し直す.
    err.ks.default[cv_iter] = mean((oldfaithful.reg.default.pred$y - test$waitin
g[sind ])^2)

    oldfaithful.reg.lw.pred <- ksmooth(train$erupt,train$waiting, kernel="normal
", bandwidth=2, x.points = test$erupt)
    err.ks.lw[cv_iter] = mean((oldfaithful.reg.lw.pred$y - test$waiting[sind ])^
2)

    oldfaithful.knn <- knn.reg(train=train$erupt,test=as.matrix(test$erupt,ncol=
1),y=train$waiting,k=50)
    err.knn[cv_iter] = mean((oldfaithful.knn$pred - test$waiting)^2)
}

c(mean(err.lm),mean(err.ks.lw),mean(err.ks.default),mean(err.knn)) # どれが一番
良いであろうか？


#####################################
## オゾンデータでスプライン

lm.model<-gam(Ozone^0.1~Temp,data=airquality)        #  線形回帰
gam.model<-gam(Ozone^0.1~s(Temp),data=airquality)    #  平滑化スプライン

new<-data.frame(Temp=seq(min(airquality$Temp),max(airquality$Temp),0.1))
lm.pred<-predict(lm.model,new)
gam.pred<-predict(gam.model,new)

plot(airquality$Ozone^0.1~airquality$Temp,xlab="Temp",ylab="Ozone",
    main="推定結果",cex.main=2,cex.lab=1.5)
lines(lm.pred~as.matrix(new),col=2,lwd=2)
lines(gam.pred~as.matrix(new),col=4,lwd=2)
legend("topleft",lwd=2,col=c(2,4),legend=c("線形 回帰","平滑化スプライン"))

sp<-seq(from=0.001,to=0.1,by=0.001)
GCV<-numeric()
for(i in 1:length(sp)){
    g.m<-gam(Ozone^0.1~s(Temp),sp=sp[i],data=airquality)
    GCV[i]<-g.m$gcv.ubre
}
plot(GCV,type='l')

g.lm<-gam(Ozone^0.1 ~ s(Temp),sp=100000,data=airquality)
g.lm.pred<-predict(g.lm,new) #ほぼlmと同じ.


######################
## 人工データ: smooth
y18 <- c(1:3, 5, 4, 7:3, 2*(2:5), rep(10, 4))
artdata = data.frame(x=1:18,y=y18)
xx  <- seq(1, length(y18), len = 201)
(s2  <- smooth.spline(artdata)) # GCV
(s02  <- smooth.spline(artdata, spar = 0.02))
(s03  <- smooth.spline(artdata, spar = 1))

plot(y18, main = "cubic spline fitting", col.main = 2)
lines(predict(s2, xx), col = 2, lwd=2)
lines(predict(s02, xx), col = 3)
lines(predict(s03, xx), col = 4)
legend(14, 3, legend = c("CV", "small","Large"), lty=c(1), col=c(2,3,4),title =
"lambda")

plot(y18, main = "cubic spline fitting", col.main = 2)
lines(predict(s2, xx), col = 2)
lines(predict(s03, xx), col = 4)
```

```
legend(14, 3, legend = c("spline", "linear"), lty=c(1), col=c(2,3,4),title = "me
thod")
```