Multivariate kernel regression

- Nonparametric regression. Suppose that (X_1, Y_1), …, (X_n, Y_n) are IID data and
$$Y_i = f(X_i) + \varepsilon_i \tag{1}$$
for $i = 1, \ldots, n$, where $(\varepsilon_1, \ldots, \varepsilon_n)$ is independent of $(X_1, \ldots, X_n)$, $E(\varepsilon_1) = 0$ and $Var(\varepsilon_1) = \sigma^2$. The problem of interest is to estimate m based on (X_1, Y_1), …, (X_n, Y_n).

- Kernel function on R^d. A kernel function k on R^d usually satisfies the usual constraints:
  (a) $k \geq 0$.
  (b) $\int k(s)ds = 1$.
  (c) $\int s_j k(s_1, \ldots, s_d)d(s_1, \ldots, s_d) = 0$ for $j = 1, \ldots, d$.
  (d) $\int \|s\|^2 k(s)ds < \infty$, where $\|(s_1, \ldots, s_d)\|^2 = \sum_{j=1}^d s_j^2$.

- Example of a kernel function on R^d. Let k_0 be the density for $N(0, 1)$. Define
$$k(x_1, \ldots, x_d) = k_0(x_1) \cdots k_0(x_d) \tag{2}$$
for $(x_1, \ldots, x_d) \in R^d$. Then k is a kernel function on R^d.

- Kernel regression estimator. Suppose that $(X_1, \ldots, X_n)$ is a random sample and X_i takes values in R^d for $i = 1, \ldots, n$. The kernel regression estimator for $f(x)$ with kernel k (defined on R^d) and bandwidth h is
$$\hat{f}(x) = \frac{\displaystyle\sum_{i=1}^n Y_i k\left(\frac{x - X_i}{h}\right)}{\displaystyle\sum_{i=1}^n k\left(\frac{x - X_i}{h}\right)}. \tag{3}$$

- Monte Carlo integration. To evaluate $\int_{[0,1]^d} f(x_1, \ldots, x_d)d(x_1, \ldots, x_d)$, we can generate U_1, …, U_L from the uniform distribution on $[0, 1]^d$ for a large L, and then use
$$\frac{1}{L}\sum_{j=1}^L f(U_j)$$
L to approximate $\int_{[0,1]^d} f(x_1, \ldots, x_d)d(x_1, \ldots, x_d)$.

- Example 1. Compute $\int_{[0,1]^2} e^{x^2+y^2} d(x, y)$ using Monte Carlo integration.

```
f <- function(x,y){ exp(x^2+y^2) }
L=10000
ans <- f(runif(L), runif(L))
mean(ans); sd(ans)

f1 <- function(x){ exp(x^2)}
```

```
integrate(f1,0,1)$value^2

tem1 <- function(x){
  tem2 <- function(y){ f(x,y) }
  vtem2 <- Vectorize(tem2)
  return(integrate(vtem2, 0, 1)$value)
}
vtem1 <- Vectorize(tem1)
integrate(vtem1, 0,1)$value
```

- Exercise 1. Write a function using R with the following input and output:

  Input:

  - data matrix `X` whose i-th row is X_i,
  - data vector $\mathtt{y} = (Y_1, \ldots, Y_n)$,
  - bandwith h, and
  - evaluation point x_0.

  Output: $\hat{f}(x_0)$ based on (3) with the kernel function k in (2).

- Example 2. Let `ker.est` be the function in Exercise 1. We will compute the kernel estimator $\hat{f}$ with kernel k in (2) and bandwidth $h = 0.05$ based on the data generated below. We will plot the estimated regression function and compute the ISE.

  Generate data as follows.

```
set.seed(1)
f <- function(x1,x2){ dnorm(x1-0.5, sd=0.2)*dnorm(x2-0.5, sd=0.2) }
n <- 1000
X <- matrix(runif(n*2), n,2)
y <- f(X[,1],X[,2]) + rnorm(n,sd=0.4)
```

  Compute $\hat{f}(x0, y0)$ and plot $\hat{f}(x_0, y_0)$ and $f(x_0, y_0)$ for $(x_0, y_0) \in \{(x, y) : x \in \{1/21, 2/21, \ldots, 20/21\}, y \in \{1/11, 2/11, \ldots, 10/11\}\}$, and then compute the ISE.

```
h0=0.05
xlist <- (1:20)/21;  ylist <- (1:10)/11
n1 <- length(xlist); n2 <- length(ylist)
zm <- matrix(0, n1, n2)
for (i in 1:n1){  for (j in 1:n2){  zm[i,j] <- f(xlist[i], ylist[j])  } }
f.persp <- persp(xlist, ylist, zm, theta=20)

for (i in 1:n1){
   xi <- xlist[i]
   fhat.xi <- rep(0, n2)
   for (j in 1:n2){
    fhat.xi[j] <- ker.est(X, y, h0, c(xi,ylist[j]))
   }
   lines(trans3d(xi, ylist, fhat.xi, pmat=f.persp), col=2)
```

```
}

dif1 <- function(u, v){ (f(u,v)-ker.est(X, y, h0, c(u,v)))^2 }
tem1 <- function(u){
  tem2 <- function(v){ dif1(u,v)}
  vtem2 <- Vectorize(tem2)
  return(integrate(vtem2, 0, 1)$value)
}
vtem1 <- Vectorize(tem1)
integrate(vtem1, 0,1)$value

#h0 = 0.05 => ISE:  0.009570552
```

- Compute the ISE for the univariate case.

```
#data generation
set.seed(1)
f <- function(x1){ dnorm(x1-0.5, sd=0.2) }
n <- 1000
X <- runif(n)
y <- f(X) + rnorm(n,sd=0.4)
curve(f, 0,1)

#compute estimated regression function values
h0=0.05
fhat.x <- rep(0, n1)
for (i in 1:n1){
    fhat.x[i] <- ker.est(X, y, h0, xlist[i])
}
lines(xlist, fhat.x, col=2)

#compute ISE
tem1 <- function(u){ (f(u)-ker.est(X, y, h0, u))^2 }
vtem1 <- Vectorize(tem1)
integrate(vtem1, 0,1)$value

#h0 = 0.05 => ISE:  0.002296325
```

Exercise 2. Consider the $\hat{f}$ in Example 2 based on the data generated below:

```
set.seed(1)
f <- function(x1,x2){ dnorm(x1-0.5, sd=0.2)*dnorm(x2-0.5, sd=0.2) }
n <- 1000
X <- matrix(runif(n*2), n,2)
y <- f(X[,1],X[,2]) + rnorm(n,sd=0.4)
```

(a) Find $\hat{f}(0.5, 0.5) - f(0.5, 0.5)$.

(b) Compute an approximate ISE for $\hat{f}$ using Monte Carlo integration with $L = 10000$. Give an approximate 95% C.I. for the approximate ISE.

Exercise 3. Compute the ISE for the kernel estimator $\hat{f}$ with kernel k in (2) and bandwidth $h = 0.05$ based on the data generated below.

```
set.seed(1)
f <- function(x1,x2,x3){
  dnorm(x1-0.5, sd=0.2)*dnorm(x2-0.5, sd=0.2)* dnorm(x3-0.5, sd=0.2)
}
n <- 1000
X <- matrix(runif(n*3), n,3)
y <- f(X[,1],X[,2],X[,3]) + rnorm(n,sd=0.4)
```

(a) Find $\hat{f}(0.5, 0.5, 0.5) - f(0.5, 0.5, 0.5)$.

(b) Compute an approximate ISE for $\hat{f}$ using Monte Carlo integration with $L = 10000$. Give an approximate 95% C.I. for the approximate ISE. Can we conclude that the ISE for $\hat{f}$ is larger than the ISE in Example 2 at level 0.05?