

Stat 471: Lecture 31

Kernel Density Estimation.

Moo K. Chung
mchung@stat.wisc.edu

December 10, 2003

- Let us consider the problem of estimating a probability density f of observations $y_1, \dots, y_n$. The most simple approach would be to use histogram which count the number of observations that belong to m disjoint bins. But it gives a discrete estimation. Suppose $S = \bigcup_{i=1}^m A_i$ is the partition of S .

$$\hat{f}(x) = \frac{\# \text{ in the } i\text{-th interval}}{n}.$$

```
x=50:250;
h=hist(intensity,x)/19660;
plot(h,'.');
```

- But we know the true density to be continuous. Using the Kernel density estimator of Rosenblatt (1956) and Parzen (1962),

$$\hat{f}_\sigma(x) = \frac{1}{n} \sum_{i=1}^n K_\sigma(x - x_i)$$

where the proper choice of kernel K_σ would be that it should be symmetric probability density. Some other boring conditions are assumed from time to time. It can be shown that $\hat{f}_\sigma$ is a probability density as well - it is related with a mixture model. We will use the discrete version of $K_\sigma \sim N(0, \sigma^2)$ (see lecture 28-30).

```
kernel=inline('exp(-x.^2/sigma^2)/
sum(exp(-x.^2/sigma^2))','x','sigma')
sigma=1, %1, 5, 15
fhat=zeros(1,length(x));
for i=1:19660 %number of sample
    fhat = fhat + kernel(x-intensity(i),
                        ,sigma)/19660;
end;
hold on; plot(fhat);
```

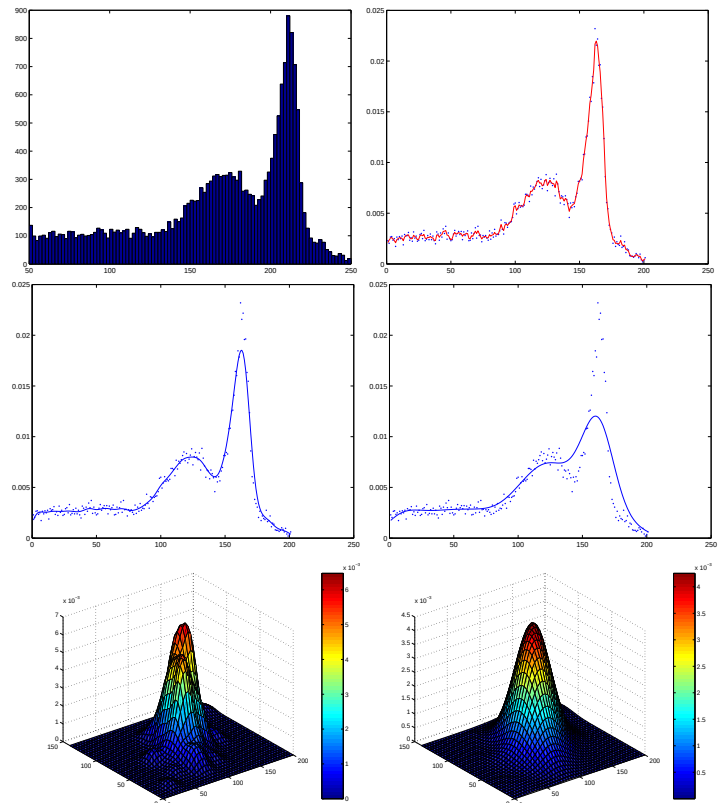
- How good is our density estimator and how we choose σ ? We can try to minimize the mean integrated squared error (σ):

$$MISE(\sigma) = \mathbb{E} \int [\hat{f}_\sigma - f(x)]^2 dx.$$

See Ruppert, Wand and Carroll's semiparametric regression.

- Bivariate kernel density estimate of observations $z_i = (x_i, y_i)'$ is given by

$$\begin{aligned} \hat{f}_\sigma(x, y) &= \frac{1}{n} \sum_{i=1}^n K_\sigma(x - x_i) K_\sigma(y - y_i) \\ &= \frac{1}{n} \sum_{i=1}^n \mathbf{K}_\sigma(z - z_i) \end{aligned}$$



where $\mathbf{K}_\sigma$ is the discrete version of bivariate normal $N(0, \sigma^2 I)$. For strength.data,

```
[x,y]=meshgrid(0:4:200,0:4:140);
kernel=inline('exp((-x.^2-y.^2)/sigma^2)/
sum(sum(exp((-x.^2-y.^2)/sigma^2)))','x','y','sigma')
sigma=10, %10, 15, 20
fhat=zeros(size(x));
for i=1:147
    fhat=fhat+kernel(x-grip(i),
                    y-arm(i),sigma)/147;
end;
>> sum(sum(fhat))
ans =
    1.0000
surf(x,y,fhat); colorbar
```

Our discrete version gives a probability function rather than a probability density such that $\hat{f}_\sigma$ sum over all possible grid points should be 1, i.e.

$$\sum_{i,j} \hat{f}_\sigma(x_i, y_j) = 1.$$