Skip Navigation Links

Page One

Campus Computing News

BulkMail To: Address Changed

New CBT Courses in Digital Imaging and Web Development

Save Travel Dollars With Videoconferencing

Coming Next Week to a Convention Center Near You!

The Unintended Consequences of Making Music

Today's Cartoon

RSS Matters

SAS Corner

The Network Connection

Link of the Month

WWW@UNT.EDU

Short Courses

IRC News

Staff Activities

Subscribe to Benchmarks *Online*

Research and Statistical Support
University of North Texas

RSS Matters

The previous issue in this series can be found in the December, 2002 issue of Benchmarks Online: Interactive Graphics in R

Interactive Graphics in R (Part II - cont.): Kernel Density Estimation in One and Two Dimensions

By Dr. Rich Herrington, Research and Statistical Support Services Manager

This month we continue our discussion of graphics in R. This month we examine histogram generation, 1-D and 2-D kernel density estimation. The GNU S language, "R" is used to implement this procedure. R is a statistical programming environment that utilizes the S and S-Plus language developed at Lucent Technologies. In the following document we illustrate the use of a GNU Web interface to the R engine on the "rss" server (http://rss.acs.unt.edu/cgi-bin/R/Rprog). This GNU Web interface is a derivative of the "Rcgi" Perl scripts available for download from the CRAN Website (http://www.cran.r-project.org), the main "R" Website. Scripts can be submitted interactively, edited, and then be re-submitted with changed parameters by selecting the hypertext link buttons that appear below the figures. For example, clicking the "Run Program" button below creates a vector of 100 random normal deviates; creates a histogram of the random numbers, and then overlays a nonparametric density estimate over the histogram. To view any text output, scroll to the bottom of the browser window. To view any graphical output, select the "Display Graphic" link. The script can be edited and resubmitted by changing the script in the form window and then selecting "Run the R Program". Selecting the browser "back page" button will return the reader to this document.

```
x<-rnorm(100)
hist(x, prob=T)
lines(density(x))
```

Run Program

Introduction to Histograms

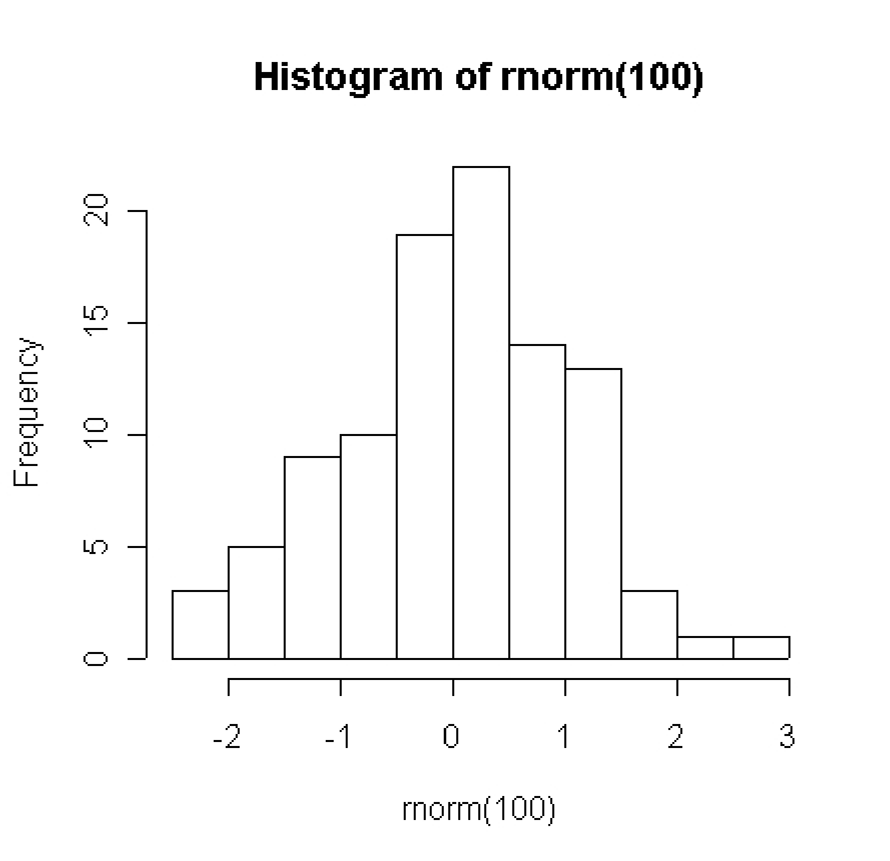
A histogram is a graphical method of representing a probability distribution over an interval of the real number line. First, we discuss the formal representation of a histogram and follow this up with an informal discussion. We assume that one has n data points from a particular probability distribution, $x_1 \ldots\ldots x_n$ over an interval [a, b]. To obtain histograms, we partition the interval [a, b] into m equally sized intervals called bins. Bin j is then:

$$B_j = [a + \frac{(b-a)\cdot(j-1)}{m}, \; , \; , \; a + \frac{(b-a)\cdot j}{m}) \text{ for j=1,....m}$$

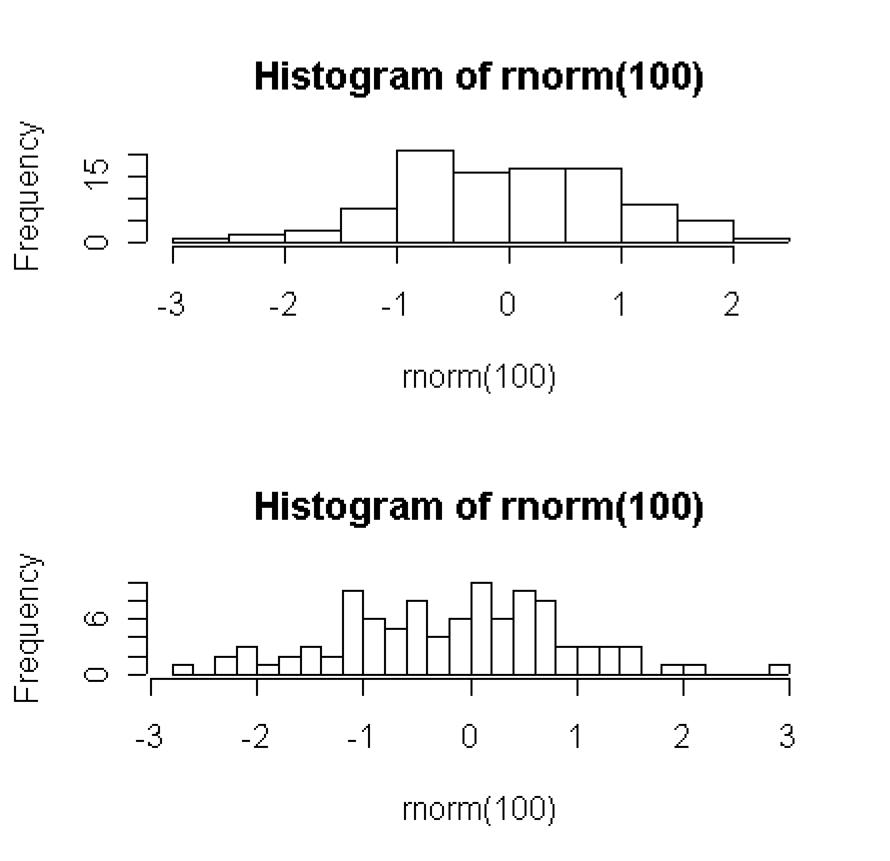
The endpoints a and b are usually taken to be the minimum and maximum of the data set. The number of data points in bin j is n_j. Of course it will turn out that

$$\sum_{1}^{j} n_j = n \cdot$$

Essentially, a histogram is the simplest non-parametric density estimator and is the that is used by most researchers. The figure below depicts a histogram of 100 pseudo random numbers from a normal distribution with a mean of zero and standard deviation of one:



This histogram is constructed by dividing the interval covered by the data values into equal sub-intervals called bins. Each time a data point falls into a particular bin, then the bin is incremented by 1. The choice of endpoints and the choice of the number of sub-intervals can have marked effects on the shape of the histogram. Data can look bimodal when represented with a particular number of bins and bin width, but can appear uni-modal when represented with, for example, less bins and a wider bin width (see below):



Thus, histograms have a few drawbacks: 1) they are not smooth, 2) the depend greatly on the end points of the bins, and 3) they depend on the width of the bins. These first two problems can be addressed by a histogram smoothing technique called "kernel density estimation". Before we look at kernel density estimation, we want to illustrate a method for simulating correlated bivariate and multivariate data sets. We will use a simulated data set with a known population correlation to illustrate kernel density estimation in two dimensions.

Simulating Data with a Known Covariance Matrix

A random vector having a multivariate normal distribution with a mean vector μ and a variance-covariance matrix V can be simulated by using the following procedure. First, form the "Cholesky Decomposistion" of the matrix V, that is, find the lower triangular matrix L such that:

$$V = L \cdot L^T$$

In Splus and R the "chol" function performs this operation. Thus, V is reproduced from the multiplication of L and the transpose of L. Next, simulate a vector z with a normal distribution whose mean is zero and standard deviation is one. A simulated vector from the required multivariate distribution is given by:

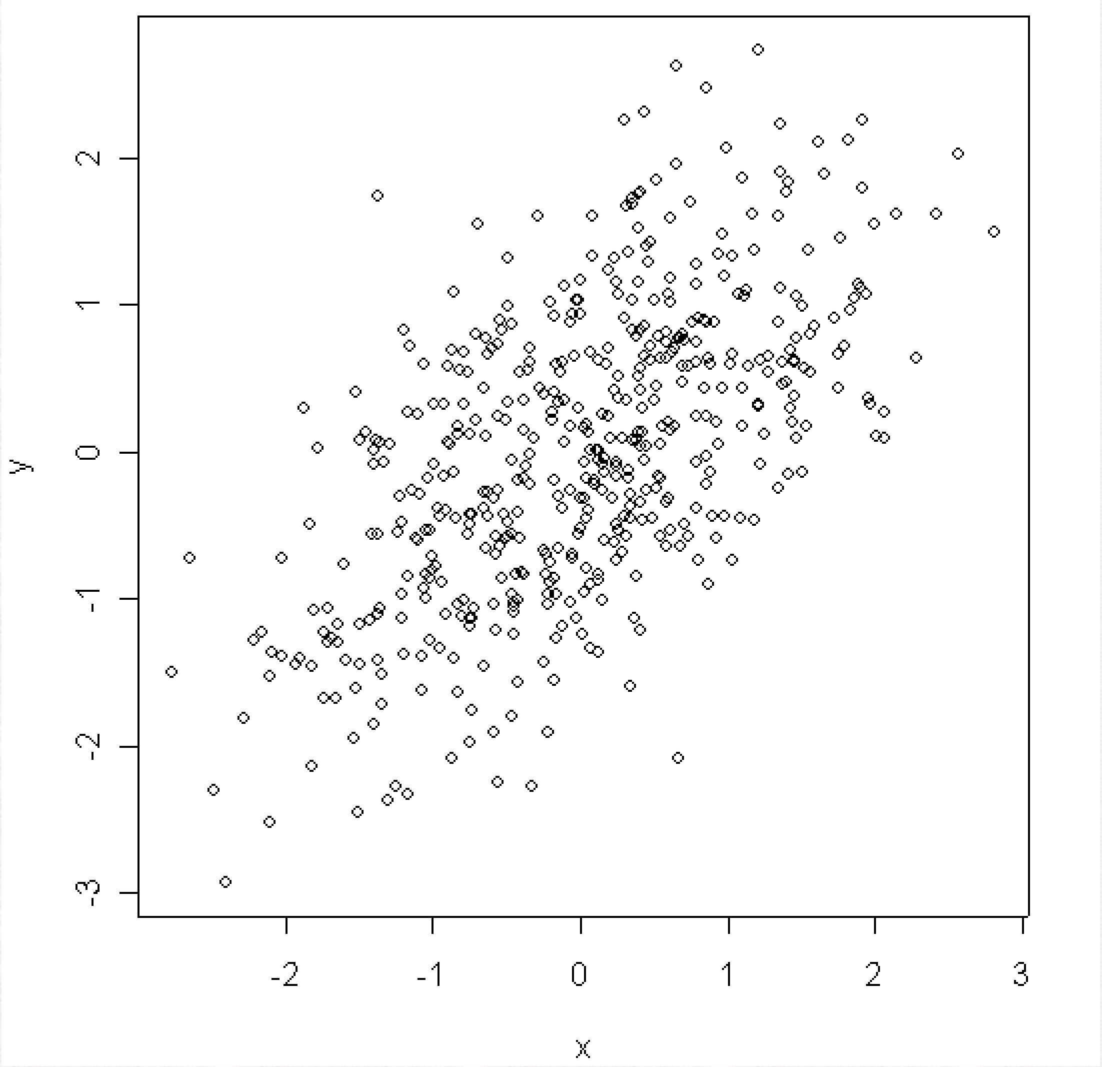
$$\mu + L \cdot z$$

Splus and R both include the MASS library. The "mvrnorm" function in the MASS library utilizes the cholesky decompostion algorithm to generate correlated data sets with a specified mean, variance and correlation structure. In the R code below, two vectors are simulated with a mean of zero, variance of one, and a correlation of .60 (as an aside, it is interesting to replace the last data point in one of the two simulated data vectors with a relatively large value. It is interesting to see how one data point can adversely effect the size of the correlation coefficient - this is left as an exercise for the reader. Furthermore, does the impact of this one large data point have the same impact on the correlation coefficient as the sample size increases?). The first example uses the MASS library to generate two correlated vectors. The "empirical=T" option allows the user to generate a data set where the correlation in the data set is exactly .60. The second example illustrates how to write a function that implements a bivariate or multivariate simulation using cholesky decomposition algorithm. In this example however, because of sampling variability, the sample correlation produced will be approximately equal to .60.

```
### Using the MASS library to generate a bivariate
#   correlated data set

library(MASS)
set.seed(321)
x.y<-mvrnorm(n=500, mu=c(0,0), Sigma=matrix(c(1,.60,.60,1),
           ncol=2), empirical=T)

x<-x.y[,1]
y<-x.y[,2]
cor(x,y)

plot(x,y)

### An example of how to create a function to create
#   a correlated bivariate data set with the cholesky
#   decomposition algorithm

rmvn<-function(n, mu=0, V=matrix(1))
    {p<-length(mu)
     D<-chol(V)
     matrix(rnorm(n*p), ncol=p)%*%D+
           rep(mu, rep(n,p))
}

x.y<-rmvn(n=500, mu=c(0,0), V=matrix(c(1,.6, .6, 1), ncol=2))
cor(x.y)
```

Run Program

The scatterplot below gives a graphical depiction of the relationship between the two simulated vectors:



Histograms and One-Dimensional Kernel Density Estimation

The essential idea behind nonparametric density estimation is to relax the parametric assumptions about the data. Usually these assumptions are replaced by ones that refer to the smoothness of the density. A histogram, being the most common nonparametric density estimator, assumes that the underlying density function is fairly smooth as determined by the bin widths. This estimate is made by binning the data and displaying the frequency or proportion of points in each bin. The "kernel density estimator" is related to the histogram, but produces a smooth estimate of the density. The kernel density estimator creates an estimate of the density by placing a "bump" at each data point and then sum the bumps up:

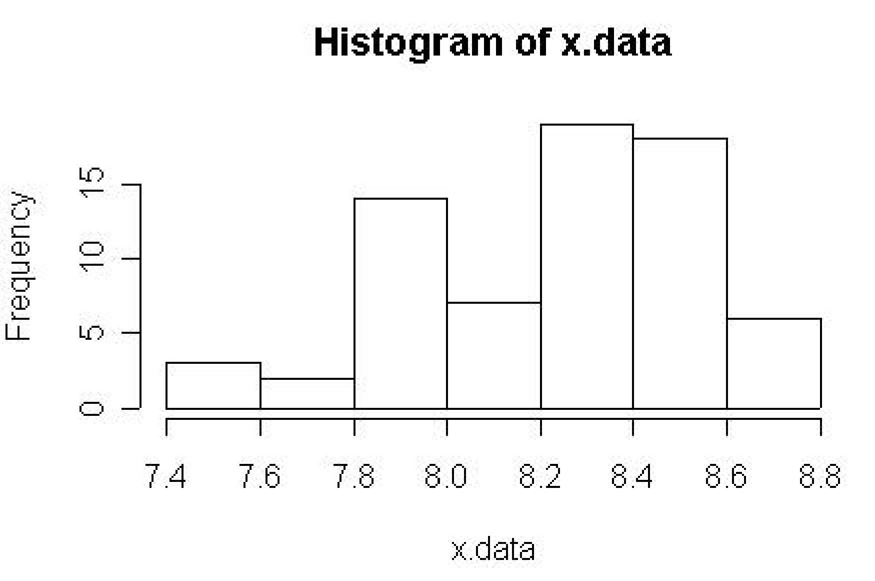
$$\hat{f}_x = \frac{1}{n \cdot h} \sum_{i=1}^{n} k\left(\frac{x - x_i}{h}\right)$$

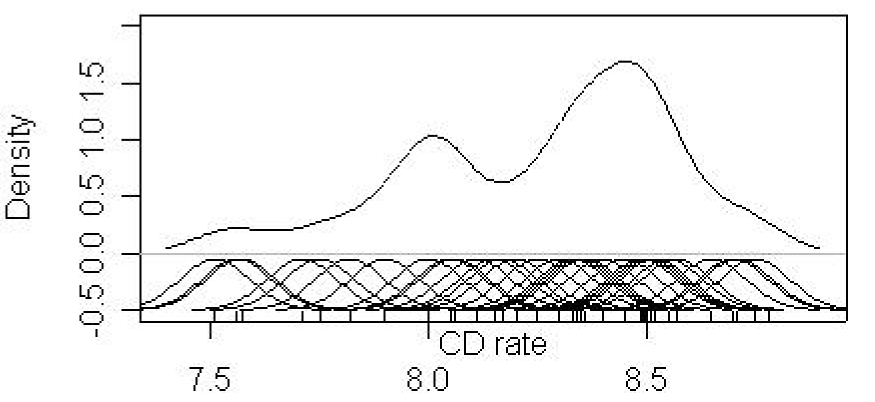
The "bump" function k( ) is called a kernel and the parameter h is the kernel width. A kernel should always be non-negative and integrate to one. The following example is taken from *Smoothing Methods in Statistics* by Jeffrey S. Simonoff (1996). The data are three-month CD rates for 69 Long Island banks and thrifts in August 1989.

```
### Example from "Smoothing Methods in Statistics"
#   by Jeffrey S. Simonoff, page 42

x.data<-read.table
("http://pages.stern.nyu.edu/~jsimonof/SmoothMeth/Data/ASCII/cdrate.dat")

x.data<-x.data$V1

par(mfrow=c(2,1))

hist(x.data, xlab="CD rate")

plot(density(x.data, kernel="gaussian", bw=.08, n=200,
    from=7.4, to=8.9), ylim=c(-.5, 2))

axis(2,at=c(0, .5, 1, 1.5))
mtext("CD rate", side=1)

for(i in 1:69)
{
lines(c(0:100)/200-.25 + x.data[i*1-1], dnorm(c(0:100)/200-.25 +
x.data[i*1-1], mean=x.data[i*1-1], sd=.08)*0.09-0.5)
}

rug(jitter(x.data,factor=.3))
```

Run Program

The figure below depicts the unsmoothed histogram and a kernel density estimate of the CD data. The kernel estimate of these data uses a Gaussian density for k, with h =0.08. The curve in the bottom of the panel is smooth, and has a tri-modal form with modes at 7.5%, 8.0%, and 8.5%. The curves along the bottom of the plot illustrate the additive form of $\hat{f}_x(x)$. The density estimate at any point in the large curve is an average of the Gaussian densities centered at each observation (the smaller curves).





The degree to which the data are smoothed has a large effect on the appearance of the density estimate. The setting of the bandwidth parameter, h, determines the degree of smoothing for the data. The simplest way to choose the bandwidth h is by choosing a value for h that minimizes our error in accuracy (minimize AMISE - asymptotic mean integrated squared error) as compared to some reference distribution (e.g. assuming that the true density is Gaussian). For example, if the reference distribution is Gaussian, and a Gaussian kernel K is used, then:

$$h_0 = 1.059 \cdot \sigma \cdot n^{-\frac{1}{5}}$$

Substituting an estimate for σ into this estimate of h gives a data-based rule for selecting h. Kernel density estimation is not without problems Boundary bias, lack of local adaptivity, and the tendency to flatten out peaks and valleys are potential difficulties with this method (Simonoff, 1996).

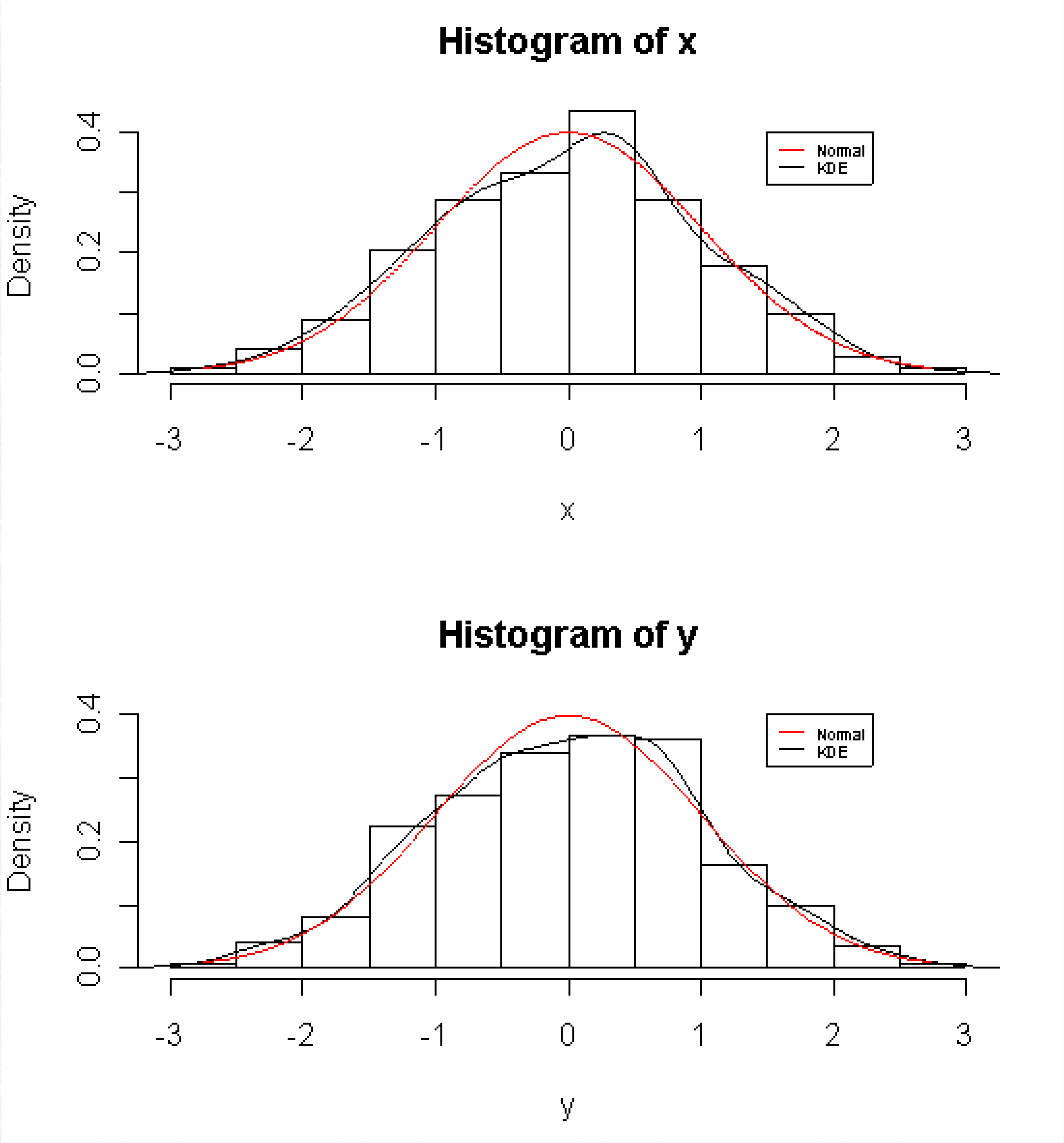
A Simulation Example of One and Two Dimensional Kernel Density Estimation

In the following example we simulate a bivariate normal distribution with a mean of zero, standard deviation of one, and a correlation of .60. The first part of the example is concerned with comparing a univariate kernel density estimate to a Gaussian (normal) distribution for each of the marginal (univariate) distributions. Since the data are randomly sampled from Gaussian distributions, the kernel density estimate and Gaussian curve should be close. The Gaussian curve is fit by estimating the mean and standard deviation of the data. Then assuming that the data is truly Gaussian, a Gaussian probability distribution is fit with the estimated mean and standard deviation as parameters. The second part of the example graphically depicts the two univariate histograms in the margins of a bivariate scatterplot for the two variables. The purpose for this graphical arrangement is to build our intuition about the joint density function which characterizes the joint variation of our two variables while still viewing the marginal distributions. While this graphical arrangement allows us to view the two marginal distributions simultaneously, it appears that little insight is gained bout the joint density function.

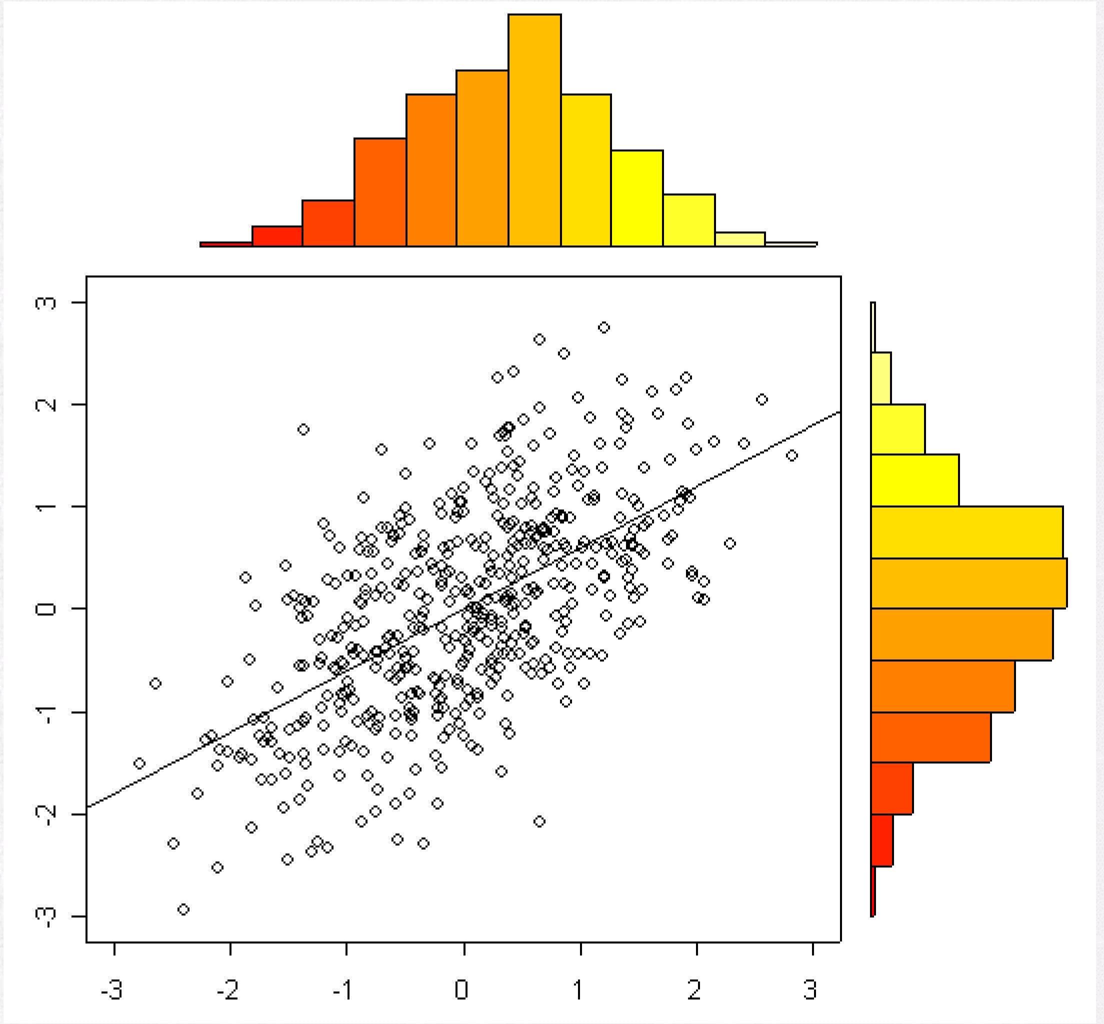
```
### Part I of the Example

par(mfrow=c(2,1))

xhist <- hist(x, prob=T, ylim=c(0,.42))

lines(density(x))

seq.x<-seq(min(xhist$mids), max(xhist$mids),.01)
lines(seq.x, dnorm(seq.x), col="red")

legend(1.5,.4,c("Normal", "KDE"),lty=1,cex=.6,
          col=c(2,1))

yhist <- hist(y, prob=T, ylim=c(0,.40))

lines(density(y))

seq.y<-seq(min(yhist$mids), max(yhist$mids),.01)
lines(seq.y, dnorm(seq.y), col="red")

legend(1.5,.4,c("Normal", "KDE"),lty=1,cex=.6,
         col=c(2,1))

### Part II of the example

top <- max(c(xhist$counts, yhist$counts))

xrange <- c(-3,3)
yrange <- c(-3,3)

layout(matrix(c(2,0,1,3),2,2,byrow=TRUE), c(3,1), c(1,3), TRUE)

plot(x, y, xlim=xrange, ylim=yrange, xlab="", ylab="")
abline(lm(y~x))

par(mar=c(0,3,1,1))

barplot(xhist$counts, axes=FALSE, ylim=c(0, top),
           space=0)

par(mar=c(3,0,1,1))

barplot(yhist$counts, axes=FALSE, xlim=c(0, top),
           space=0, horiz=TRUE)
```

Run Program

The following figure depicts the marginal distributions fit with both a Gaussian distribution (Normal) and a kernel density estimate.



The "layout" function in Splus/R allows us to define three regions for the graphics plot. The "horiz=T" option for the "barplot" function allows us to depict the y marginal distribution's y axis horizontally in the right portion of the graph layout.



Regions of high density in the scatterplot will be where the high density portions of each marginal distribution intersect. It is somewhat difficult to discern where the areas of high density in the scatterplot are because each scatterplot point gets overlaid on previously plotted points.

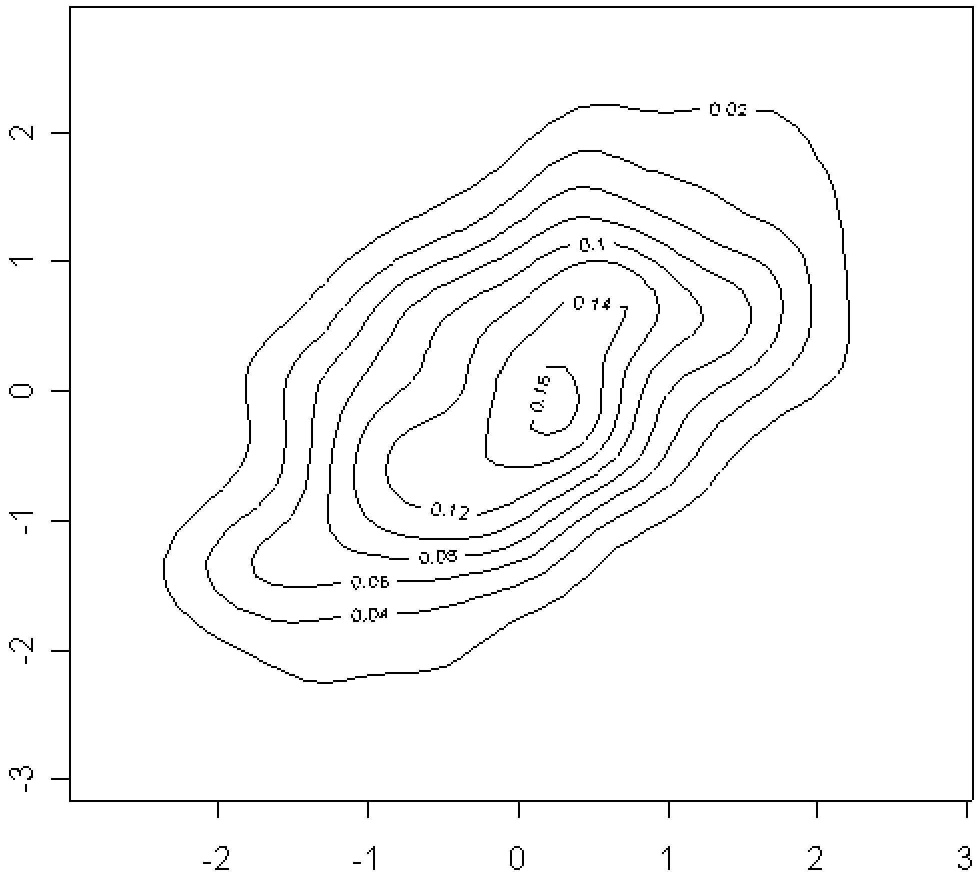
Contour and Perspective Plots: Two Dimensional Kernel Density Estimation

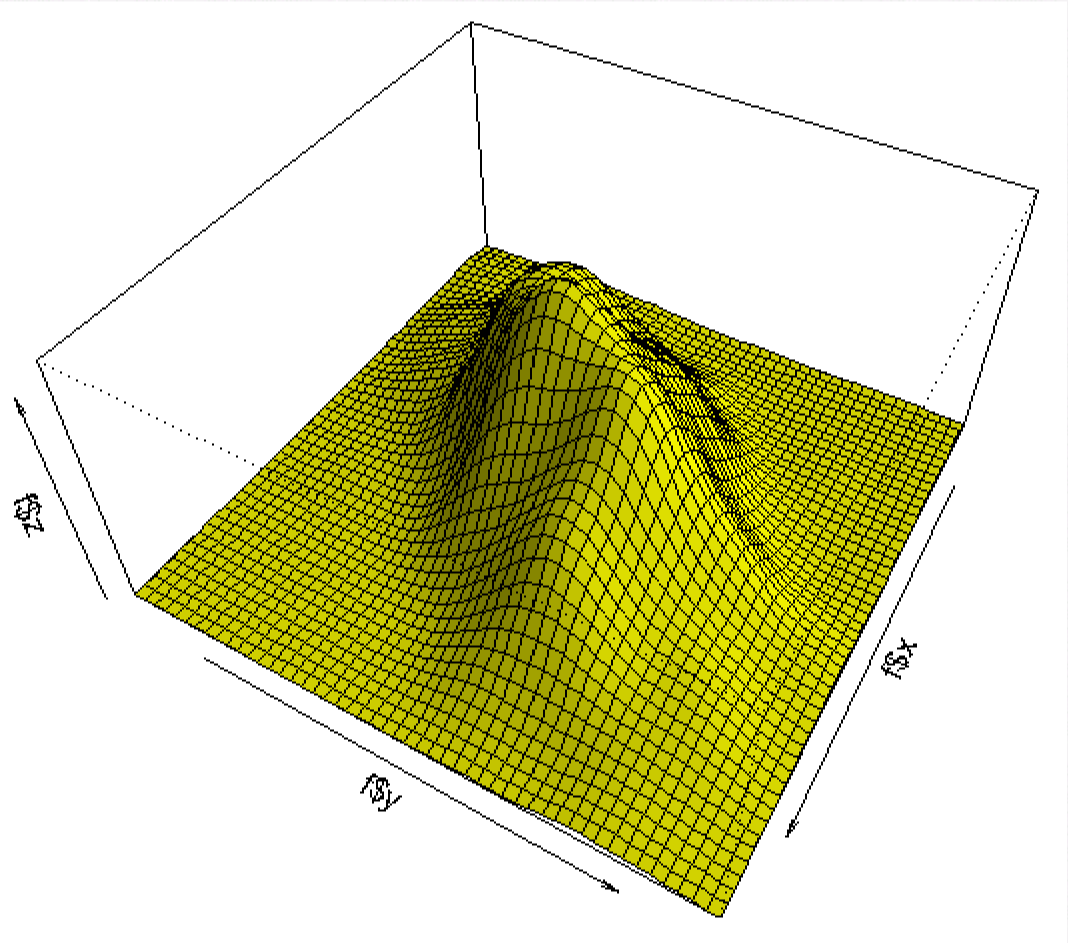
The MASS library provides the function "kde2d" for extending univariate kernel density estimation to two dimensions (Venables and Ripley, 2002). We defer the details of multivariate kernel density estimation to Simonoff (1996). In the following program, we continue to use the simulated bivariate data set with mean of zero, standard deviation of one, and correlation of .60.

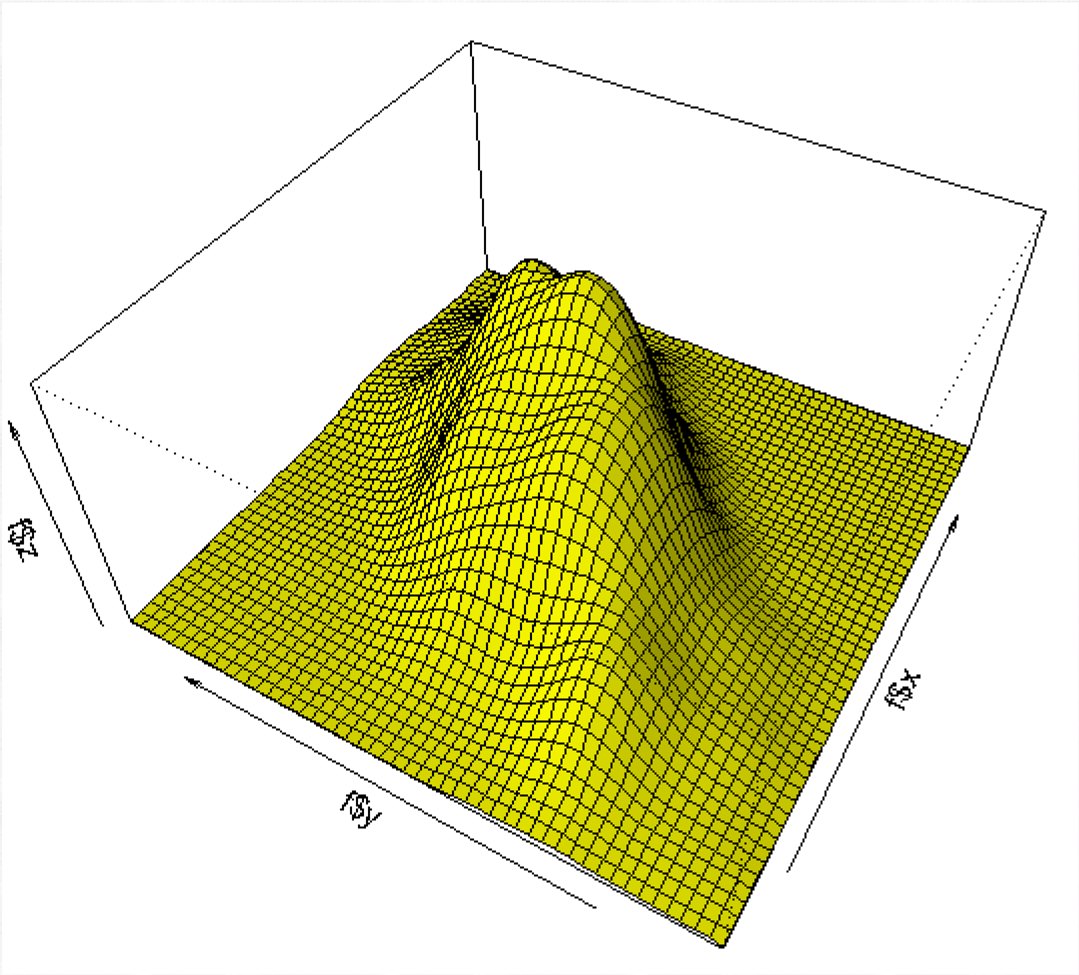
```
par(mfrow=c(2,1))

plot(x,y)

f<-kde2d(x,y,n=50)

contour(f$x,f$y,f$z)

persp(f$x,f$y,f$z,shade=.3, theta = 120, phi = 45,
     expand = 0.5, col = "yellow")

persp(f$x,f$y,f$z,shade=.3, theta = 300, phi = 45,
     expand = 0.5, col = "yellow")

par(mfrow=c(2,2))

plot(x,y)

f<-kde2d(x,y,n=50)

contour(f$x,f$y,f$z)

persp(f$x,f$y,f$z,shade=.3, theta = 120, phi = 45,
     expand = 0.5, col = "yellow")

persp(f$x,f$y,f$z,shade=.3, theta = 300, phi = 45,
     expand = 0.5, col = "yellow")
```

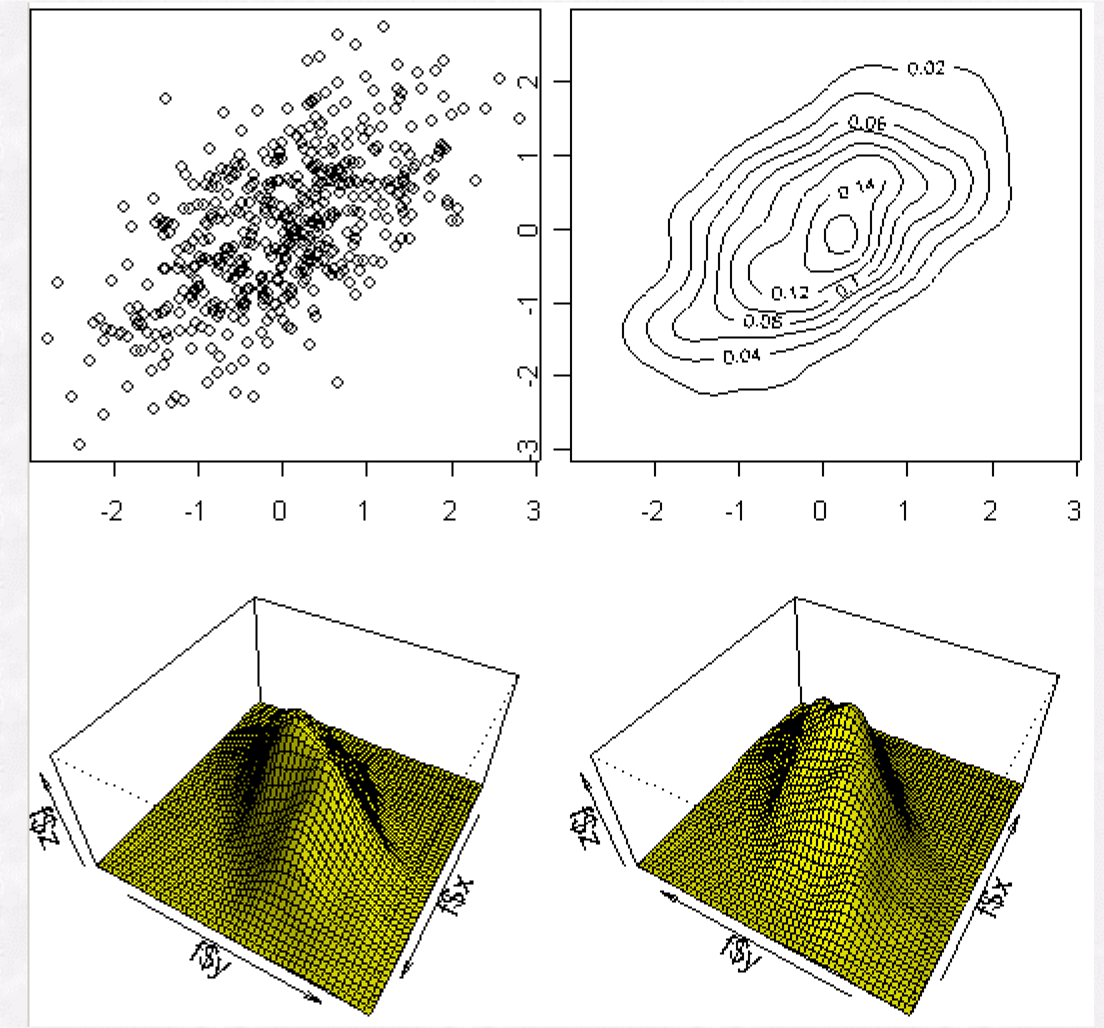
Run Program

The following graphical output is produced:


0.02
0.1
0.14
0.16
0.12
0.08
0.06
0.04
2
1
0
-1
-2
-3
-2
-1
0
1
2
3


f$z
f$y
f$x


f$z
f$y
f$x



Conclusions

There are a number of ways in which kernel density smoothing can aid in analysis and inference - here we only review a few. First, kernel density estimation provides an exploratory method for potentially highlighting important structure in the data. For example, in the CD rate data, an initial hypothesis of a Gaussian distribution might be reasonable. However, examining the kernel density estimate of the data reveals the possible existence of two, possibly three subgroups in the data. Secondly, the smooth curves of the kernel density estimate can be used to test the adequacy of fit of a hypothesized model. For example, the difference between the assumed Gaussian model and the nonparametric kernel density estimate curves can be used to define a test of the goodness of fit for the Gaussian distribution. Tests constructed this way can be more powerful than those based on the empirical distribution alone (Simonoff, 1996). Lastly, standard methodologies can be modified using smoothed density estimates by substituting the density estimate for either the empirical or parametric density function. For example, the bootstrap is a methodology that is improved by substituting the empirical distribution function by a smoothed version of it (see RSS Matters Oct. 2001). These examples are only a few of the ways in which smoothing methods can useful. Readers are encouraged to review Simonoff's 1996 summary of smoothing methods in statistics.

Next Time

Next time we return to Part II of our series on multilevel modeling using the NLME (linear and nonlinear mixed effects) functions in R and S-Plus.

References

Krause, A. and Olson, M. (2000). *The Basics of S and S-Plus, 2nd Edition.* Springer Verlag: New York.

Simonoff, Jeffrey S. (1996). *Smoothing Methods in Statistics.* Springer Verlag: New York.

Venables, W.N. & Ripley, B.D. *Modern Applied Statistics with S, 4th Edition*. Springer Verlag: New York.