Module 2-1: Multivariate Data Visualization

Weixing Song

August 15, 2018

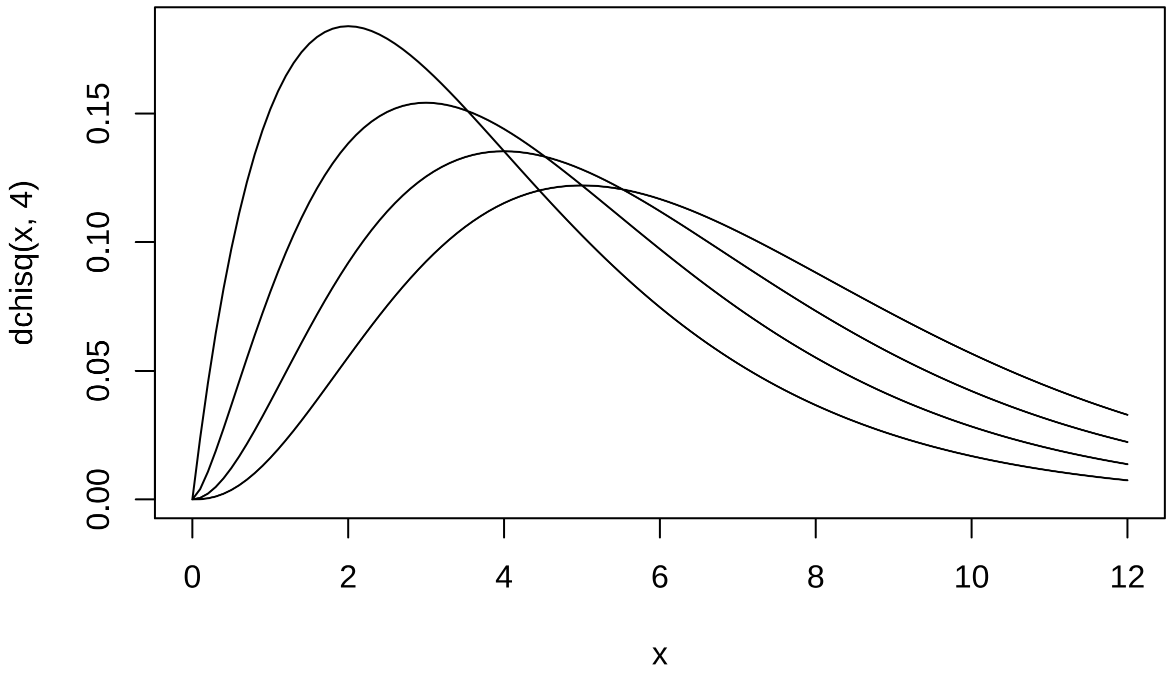
First, we use the following R-codes to install all necessary packages.

```r
list.of.packages=c("aplpack","graphics","lattice","MASS","MVA","boot","rlang","plotly")
if(length(which(!list.of.packages %in% installed.packages()))){
  install.packages(list.of.packages[!list.of.packages %in% installed.packages()])}
```

The following R codes draw a series χ^2-density plots.

1. Line plots

```r
x=seq(0,12,by=0.1)
plot(x, dchisq(x,4),type="l")
lines(x,dchisq(x,5))
lines(x,dchisq(x,6))
lines(x,dchisq(x,7))
```



```r
pdf(file="chisquare.pdf")
plot(x, dchisq(x,4),type="l",ylab="Chi-squared density")
```

```
lines(x,dchisq(x,5))
lines(x,dchisq(x,6))
lines(x,dchisq(x,7))
text(c(3,4.2,5.2,6.2),c(0.18,0.153,0.137,0.123),labels=c("4 df", "5 df","6 df", "7 df"))
dev.off()
```

```
## pdf
##   2
```

2. Histogram, Stem-and-Leaf Plot.

```
require(boot)
```

```
## Loading required package: boot
```

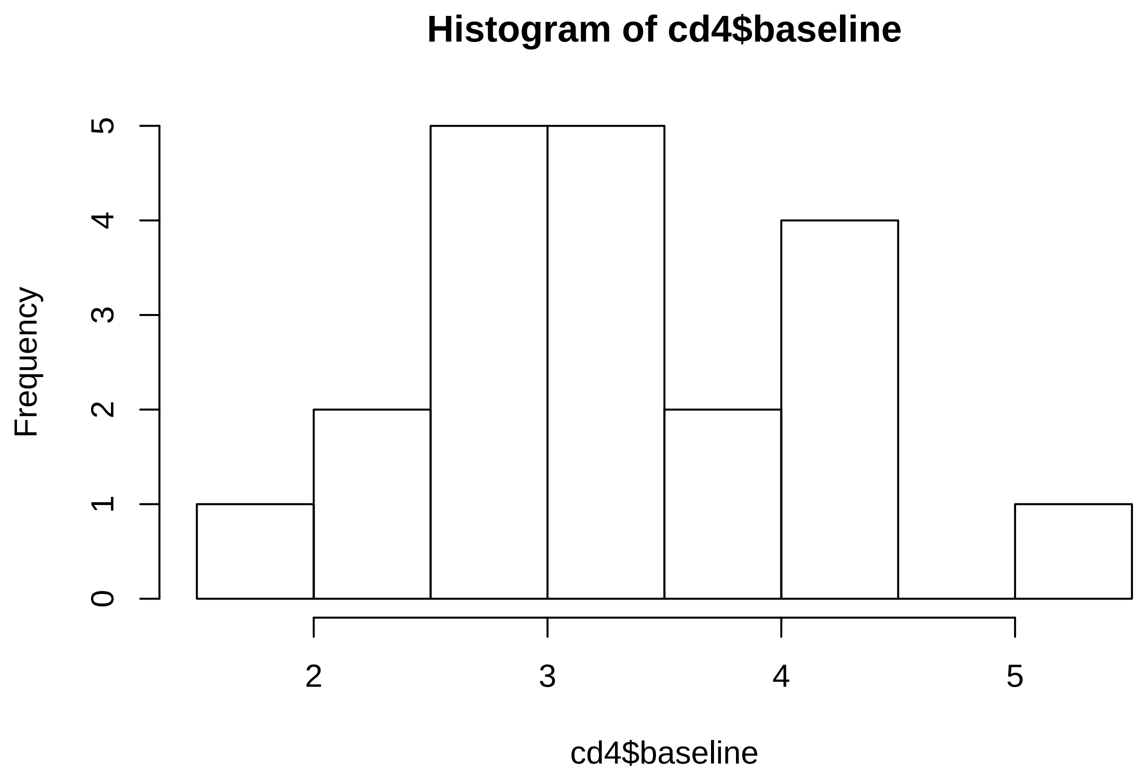
```
cd4
```

```
##    baseline oneyear
## 1      2.12    2.47
## 2      4.35    4.61
## 3      3.39    5.26
## 4      2.51    3.02
## 5      4.04    6.36
## 6      5.10    5.93
## 7      3.77    3.93
## 8      3.35    4.09
## 9      4.10    4.88
## 10     3.35    3.81
## 11     4.15    4.74
## 12     3.56    3.29
## 13     3.39    5.55
## 14     1.88    2.82
## 15     2.56    4.23
## 16     2.96    3.23
## 17     2.49    2.56
## 18     3.03    4.31
## 19     2.66    4.37
## 20     3.00    2.40
```

```
head(cd4)
```

```
##   baseline oneyear
## 1     2.12    2.47
## 2     4.35    4.61
## 3     3.39    5.26
## 4     2.51    3.02
## 5     4.04    6.36
## 6     5.10    5.93
```

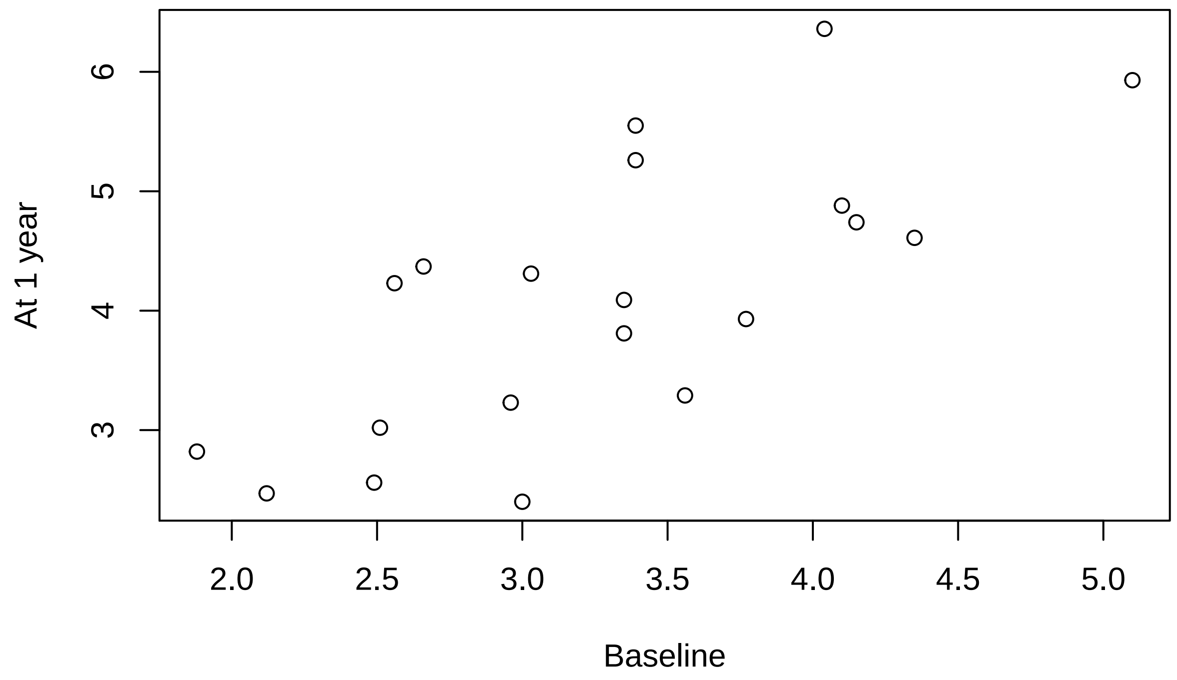
```
hist(cd4$baseline)
```



```r
stem(cd4$baseline)
```

```
##
##   The decimal point is at the |
##
##   1 | 9
##   2 | 15567
##   3 | 000444468
##   4 | 0124
##   5 | 1
```

```r
plot(cd4$baseline,cd4$oneyear,xlab="Baseline",ylab="At 1 year")
```



A fancier back-to-back stem-and-leaf plot

```r
library(aplpack)
```

```
## Loading required package: tcltk
```

```r
stem.leaf.backback(cd4$baseline,cd4$oneyear)
```

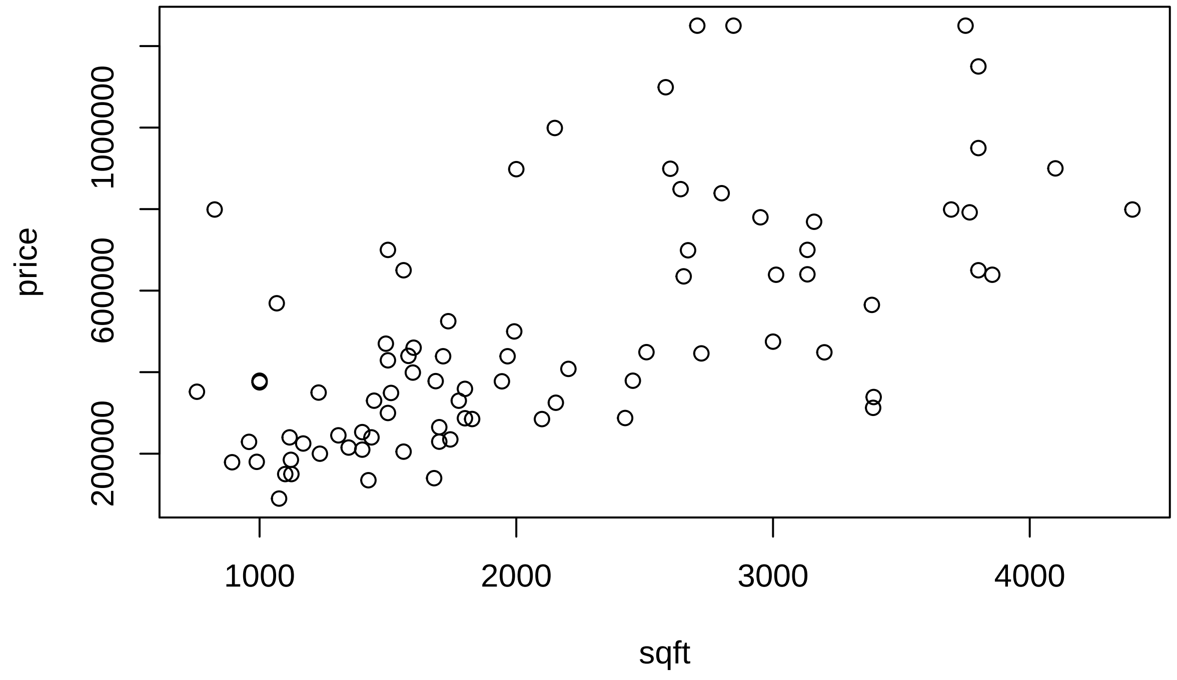
```
## ________________________________
##   1 | 2: represents 1.2, leaf unit: 0.1
##     cd4$baseline      cd4$oneyear
## ________________________________
##                | 1* |
##    1          8| 1. |
##    3         41| 2* |44       2
##    7       9655| 2. |58       4
##   (6)    333300| 3* |022      7
##    7         75| 3. |89       9
##    5       3110| 4* |0233    (4)
##                | 4. |678      7
##    1          1| 5* |2        4
##                | 5. |59       3
##                | 6* |3        1
##                | 6. |
##                | 7* |
## --------------------------------
## n:          20        20
## --------------------------------
```

3. Scatter plot

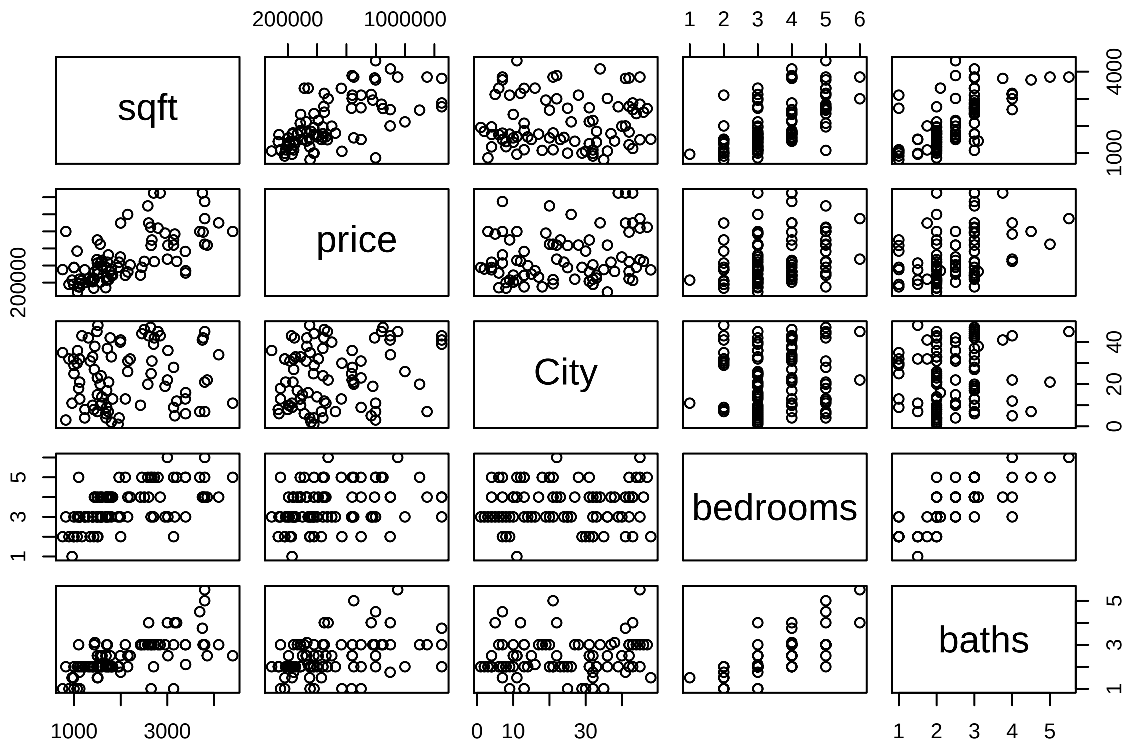
```
house=read.table("http://www.rossmanchance.com/iscam2/data/housing.txt",header=T,sep = "\t")
attach(house)
class(house)
```

```
## [1] "data.frame"
```

```
plot(sqft,price)
```

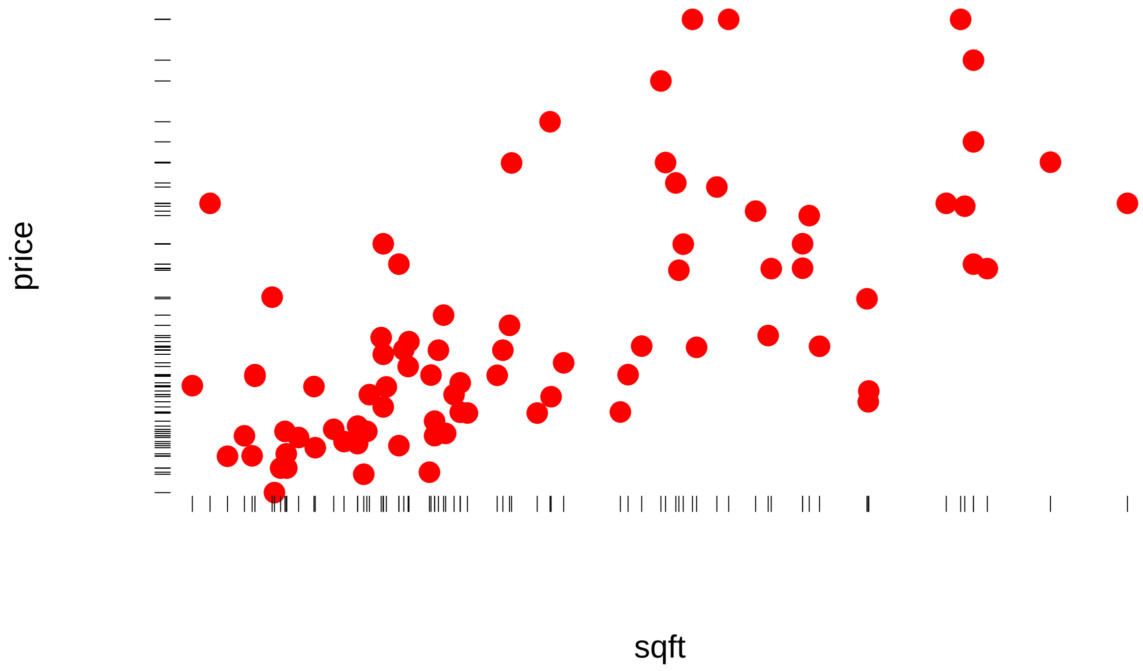


```
plot(house)
```



A fancier scatter plot with rug fringes

```
require(graphics)
plot(sqft,price,axes=F,cex=1.5,pch=16,col="red")
rug(sqft,side=1)
rug(price,side=2)
```



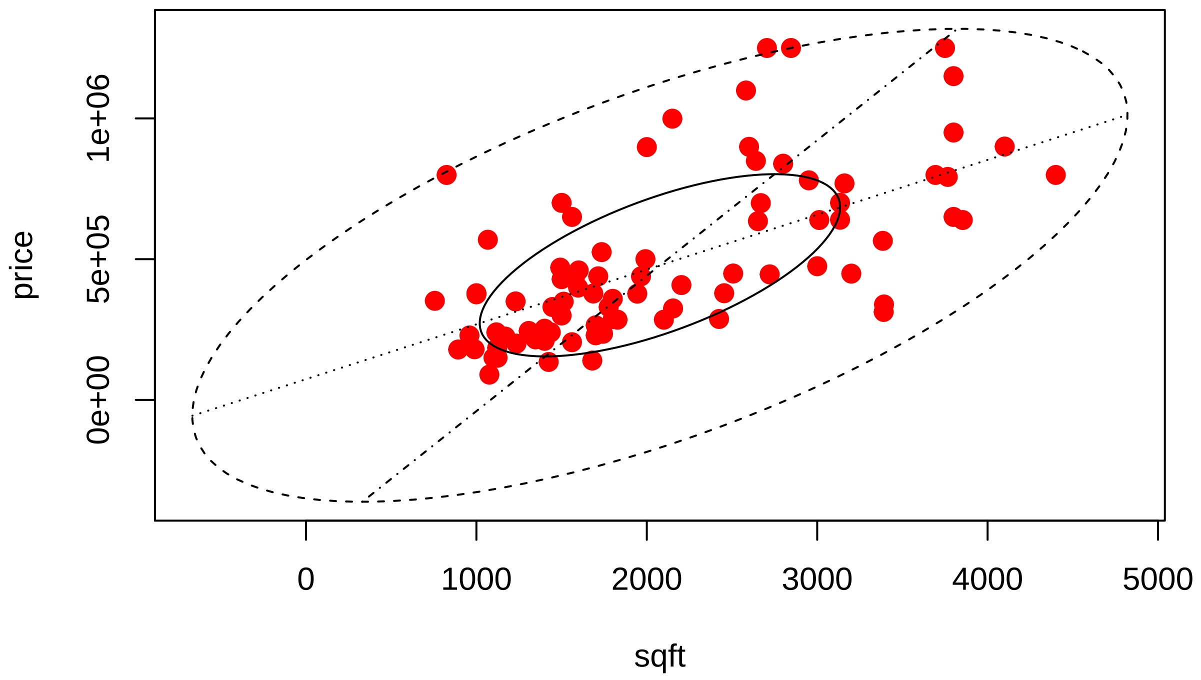
4. Bivariate boxplot

The bivariate boxplot includes a pair of estimated ellipses, the inner one containing approximately 50% of the data and the outer one containing about 95%. The two lines inside the ellipses are estimates of the regression line. The darker line is the usual least squares line using all of the observations. The lighter line is a more robust estimate that reduces the influence of any extreme outlying data values.

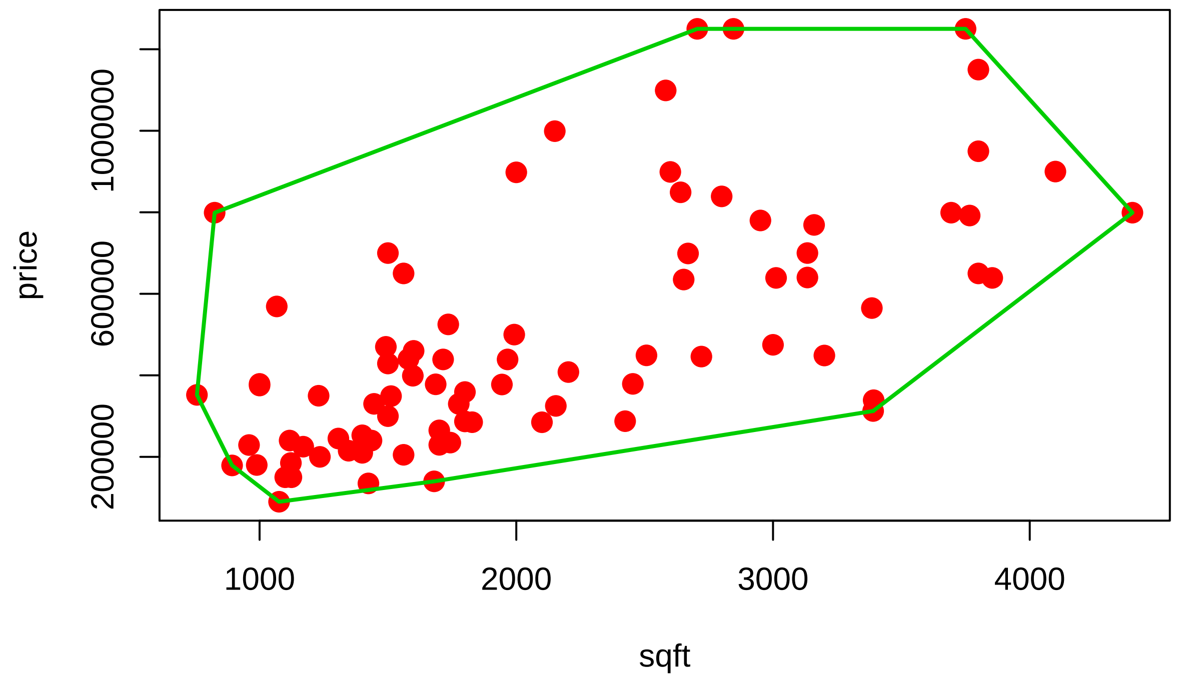
```r
require(MVA)
```

```
## Loading required package: MVA

## Loading required package: HSAUR2

## Loading required package: tools
```

```r
bvbox(cbind(sqft,price),xlab="sqft",ylab="price",pch=19,cex=1.25,col="red")
```



```r
ch=chull(cbind(sqft,price)) # Returns the indices of the data points that are most extreme in each dire
ch=c(ch,ch[1])
plot(sqft,price,cex=1.5,pch=16,col="red")
lines(sqft[ch],price[ch],type="l",col=3,lwd=2)
```



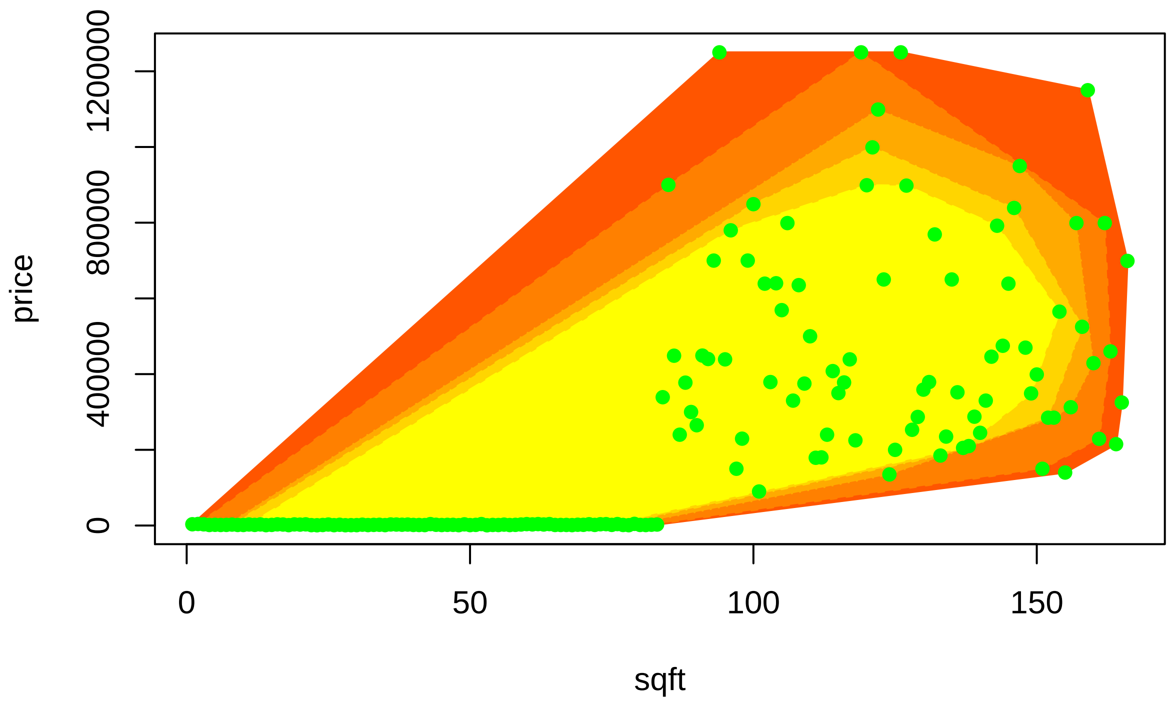
5. Convex hull plot

Mathematically, the convex hull is the smallest convex polygon containing all of the data.

```
library(aplpack)
nlev=5;
colors=heat.colors(9)[3:(nlev+2)]
plothulls(c(sqft,price),n.hull=nlev, col.hull = colors, xlab="sqft",ylab="price",lty.hull=1:nlev,
          density=NA,col=0,main="")
```

```
## [[1]]
##       x.hull  y.hull
##  [1,]    165  325000
##  [2,]    164  215000
##  [3,]    155  139900
##  [4,]     83    2669
##  [5,]     81    1347
##  [6,]     78     959
##  [7,]     53     756
##  [8,]     23     825
##  [9,]     14    1100
## [10,]      4    1436
## [11,]      1    3392
## [12,]     94 1250000
## [13,]    126 1250000
## [14,]    159 1150000
## [15,]    166  699000
##
## [[2]]
##       x.hull  y.hull
##  [1,]    161  229000
##  [2,]    151  150000
##  [3,]     82    2154
##  [4,]     80    1600
##  [5,]     68    1124
##  [6,]     48    1000
##  [7,]     28     893
##  [8,]     18    1076
##  [9,]      6    1500
## [10,]      5    1944
## [11,]      3    3200
## [12,]      2    4100
## [13,]     85  899900
## [14,]    119 1250000
## [15,]    162  799000
## [16,]    163  460000
##
## [[3]]
##       x.hull  y.hull
##  [1,]    124  134900
##  [2,]     79    4400
##  [3,]     77    1500
##  [4,]     50    1122
##  [5,]     29     989
##  [6,]     26    1000
##  [7,]     22    1067
##  [8,]     10    1500
##  [9,]      7    1700
## [10,]    122 1099000
## [11,]    147  949750
## [12,]    157  799000
## [13,]    160  429000
## [14,]    156  312500
## [15,]    153  285000
##
## [[4]]
##       x.hull y.hull
##  [1,]    138 209900
##  [2,]    133 184900
##  [3,]     76   3800
##  [4,]     75   1735
##  [5,]     66   1512
##  [6,]     57   1307
##  [7,]     30   1117
##  [8,]      9   1580
##  [9,]      8   2507
## [10,]    100 849000
## [11,]    121 999000
## [12,]    146 839000
## [13,]    158 525000
## [14,]    152 284900
##
## [[5]]
##       x.hull y.hull
##  [1,]    137 205000
##  [2,]    101  89900
##  [3,]     74   3694
##  [4,]     72   1680
##  [5,]     65   1492
##  [6,]     42   1235
##  [7,]     35   1170
##  [8,]     32   1230
##  [9,]     12   1715
## [10,]     11   2705
## [11,]     96 779900
## [12,]    120 899000
## [13,]    127 898000
## [14,]    143 792000
## [15,]    154 565000
## [16,]    149 349000
```

```r
points(c(sqft,price),pch=16,cex=1,col="green")
```



6. Bubble plot

A Bubble Chart is a multivariate graph combining the features of a scatterplot and a proportional area chart. It uses a Cartesian coordinate system to plot points along a grid where the X and Y axis are separate variables. Each plotted point represents a third variable by the area of its circle.

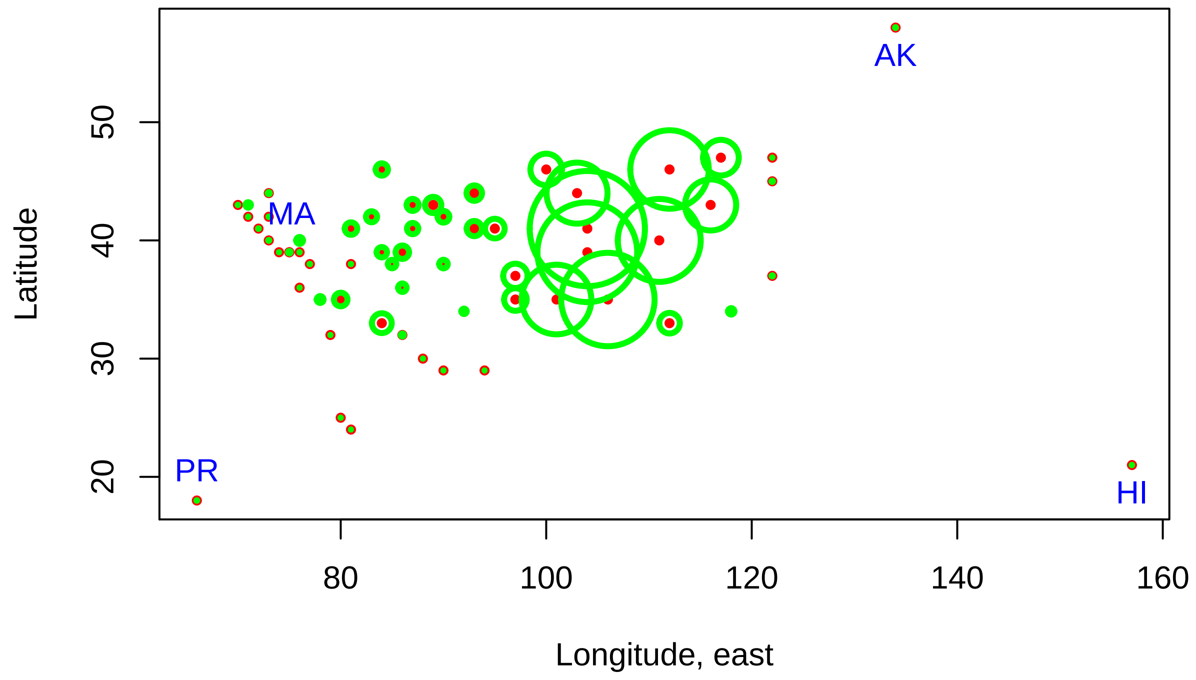
```
# Bubble plot of altitude of US cities
JanTemp=read.table("S:/Workshop/Data/janTemp.txt",header=T)
require(MASS)
```

```
## Loading required package: MASS
```

```
attach(JanTemp)
```

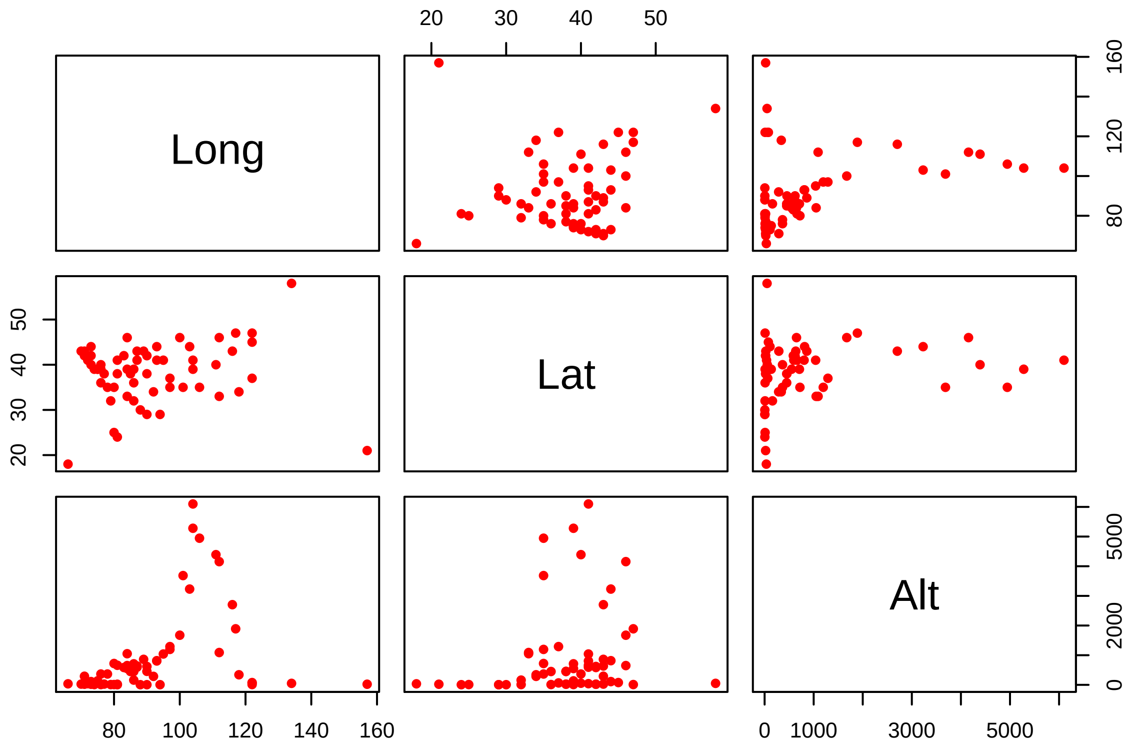
```
## The following object is masked from house:
##
##     City
```

```
plot(Long,Lat,pch=16,cex=0.7,xlab="Longitude, east",ylab="Latitude",col="red")
with(JanTemp,symbols(Long,Lat,circles=Alt,inches=0.3,add=T,lwd=3,fg="green"))
landmarks=c("AK","HI","MA","PR")
lmi=match(landmarks,State)
text(Long[lmi],Lat[lmi],labels=landmarks,pos=c(1,1,4,3),col="blue")
```



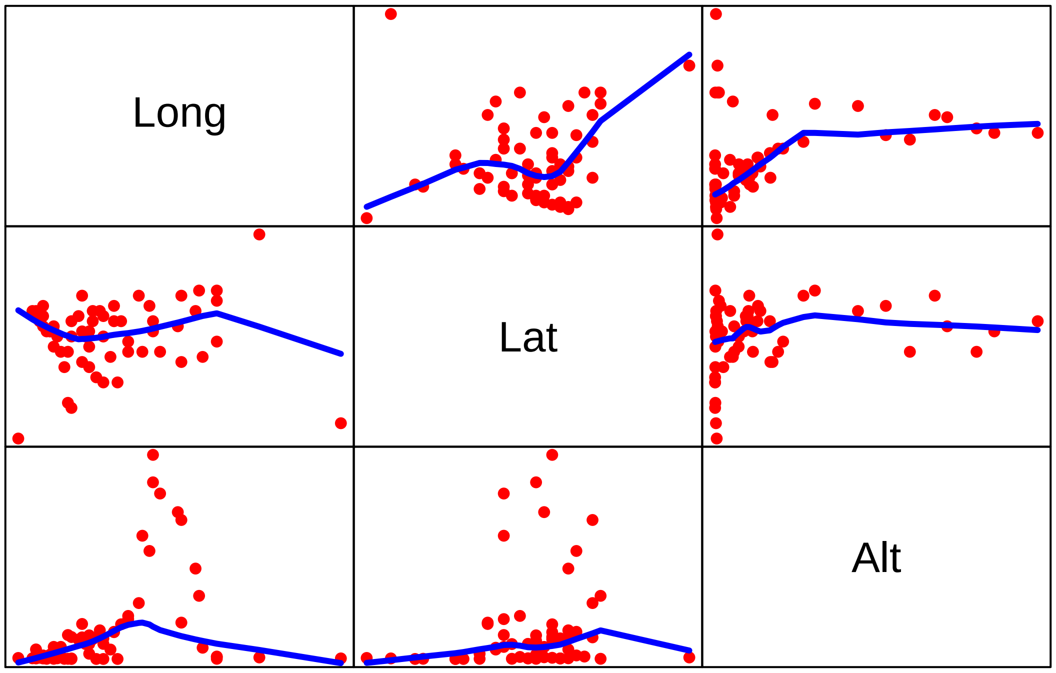
# 7. Scatter plot matrix

```
# Scatter plot matrix

pairs(~Long+Lat+Alt,pch=16,col="red")
pairs(cbind(Long,Lat,Alt),pch=16,col="red")
```



```
pairs(cbind(Long,Lat,Alt),lwd=3,pch=16,cex=1.25,col="red",gap=0,xaxt="n",yaxt="n",panel=panel.smooth,co
```

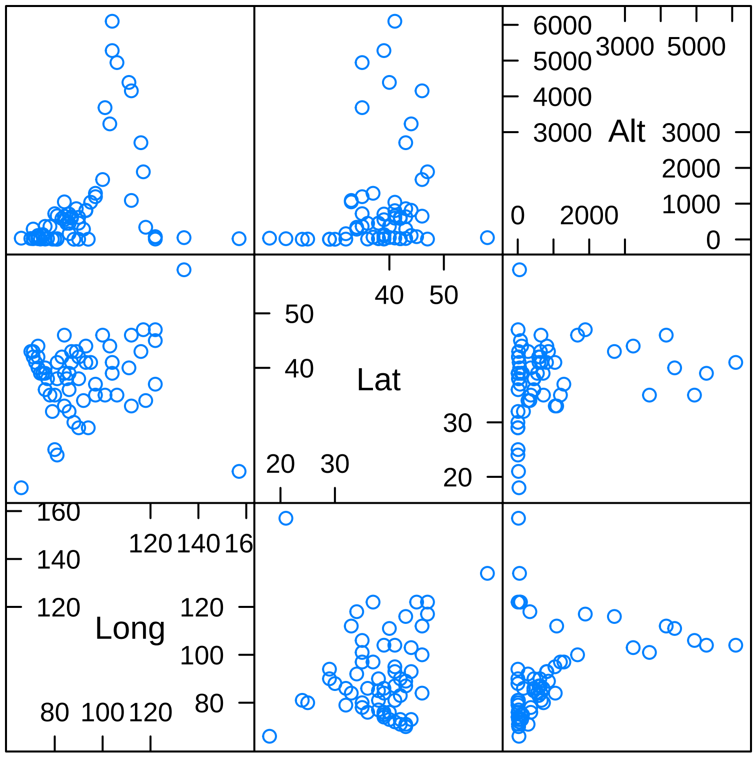
```
## Warning in plot.window(...): "col.smooth" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "col.smooth" is not a graphical
## parameter

## Warning in title(...): "col.smooth" is not a graphical parameter

## Warning in plot.window(...): "col.smooth" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "col.smooth" is not a graphical
## parameter

## Warning in title(...): "col.smooth" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "col.smooth"
## is not a graphical parameter

## Warning in plot.window(...): "col.smooth" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "col.smooth" is not a graphical
## parameter

## Warning in title(...): "col.smooth" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "col.smooth"
## is not a graphical parameter

## Warning in plot.window(...): "col.smooth" is not a graphical parameter
## Warning in plot.xy(xy, type, ...): "col.smooth" is not a graphical
## parameter

## Warning in title(...): "col.smooth" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "col.smooth"
## is not a graphical parameter

## Warning in plot.window(...): "col.smooth" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "col.smooth" is not a graphical
## parameter

## Warning in title(...): "col.smooth" is not a graphical parameter

## Warning in plot.window(...): "col.smooth" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "col.smooth" is not a graphical
## parameter

## Warning in title(...): "col.smooth" is not a graphical parameter

## Warning in plot.window(...): "col.smooth" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "col.smooth" is not a graphical
## parameter

## Warning in title(...): "col.smooth" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "col.smooth"
## is not a graphical parameter

## Warning in plot.window(...): "col.smooth" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "col.smooth" is not a graphical
## parameter

## Warning in title(...): "col.smooth" is not a graphical parameter

## Warning in plot.window(...): "col.smooth" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "col.smooth" is not a graphical
## parameter

## Warning in title(...): "col.smooth" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "col.smooth"
## is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "col.smooth"
## is not a graphical parameter
```



```r
# Scatter plot using lattice package

library(lattice)
```

```
##
## Attaching package: 'lattice'

## The following object is masked from 'package:boot':
##
##     melanoma
```

```r
splom(cbind(Long,Lat,Alt))
```

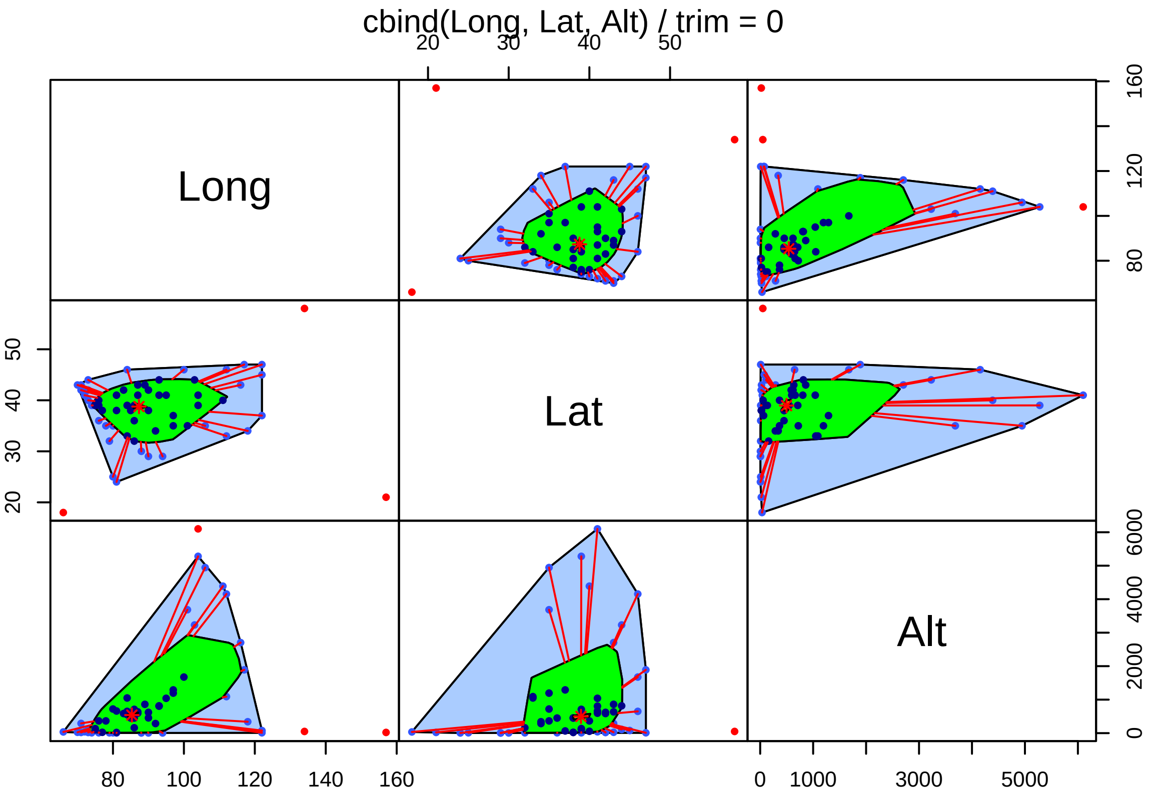


Scatter Plot Matrix

# 8. Bagplot

The bagplot.pairs routine combines several features we have covered: the bivariate boxplot, the convex hull, and the pairs scatterplot. The bagplot contains a scatterplot of bivariate data. The outer polygon of each bagplot is a convex hull, excluding extreme outliers. The inner polygon (the “bag”) of the bagplot contains the central 50% of the data. Data points between the two convex hulls are connected with lines to the central, bivariate median.

```
library(aplpack)
bagplot.pairs(cbind(Long,Lat,Alt),gap=0,col.baghull="green")
```

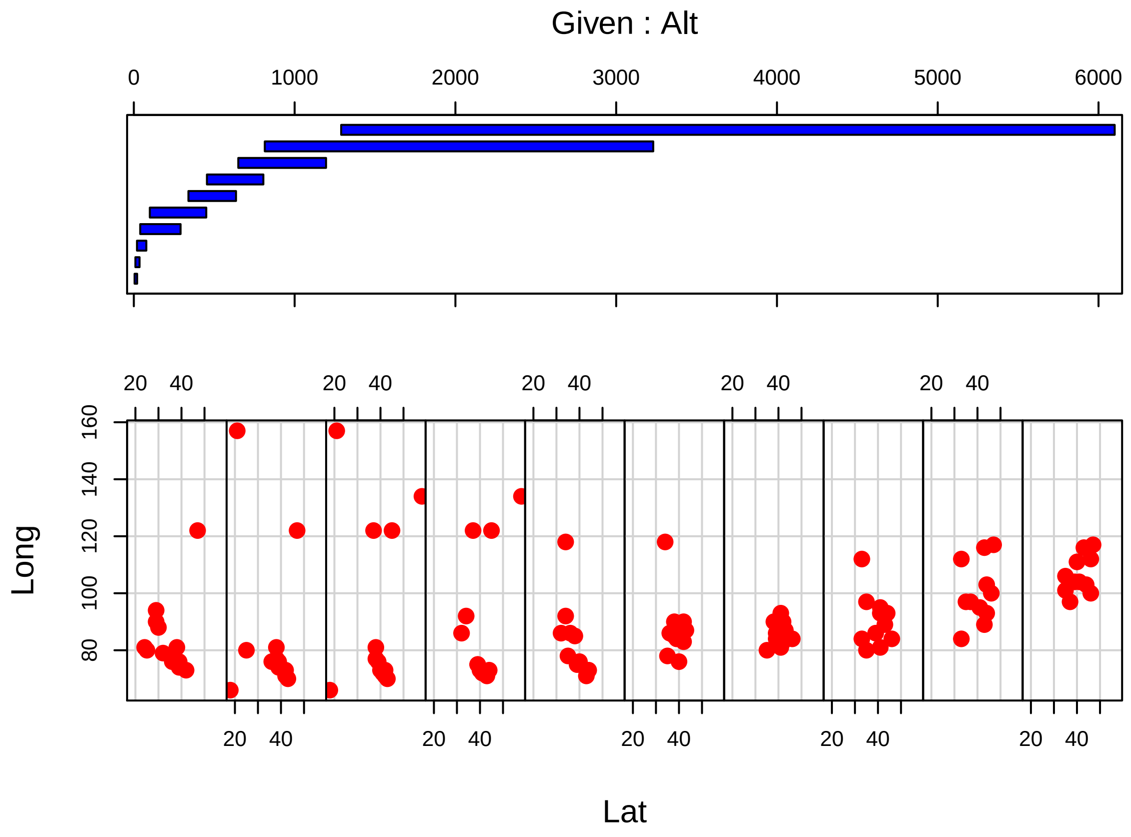


```
##       Long Lat  Alt
##  [1,]   88  30    5
##  [2,]   86  32  160
##  [3,]  134  58   50
##  [4,]  112  33 1090
##  [5,]   92  34  286
##  [6,]  118  34  340
##  [7,]  122  37   65
##  [8,]  104  39 5280
##  [9,]   72  41   40
## [10,]   75  39  135
## [11,]   77  38   25
## [12,]   81  38   20
## [13,]   81  24    5
## [14,]   80  25   10
## [15,]   84  33 1050
## [16,]  157  21   21
## [17,]  116  43 2704
## [18,]   87  41  595
## [19,]   86  39  710
## [20,]   93  41  805
## [21,]   90  42  620
## [22,]   97  37 1290
## [23,]   85  38  450
## [24,]   90  29    5
## [25,]   70  43   25
## [26,]   76  39   20
## [27,]   71  42   21
## [28,]   83  42  585
## [29,]   84  46  650
## [30,]   93  44  815
## [31,]   90  38  455
## [32,]  112  46 4155
## [33,]   95  41 1040
## [34,]   71  43  290
## [35,]   74  39   10
## [36,]  106  35 4945
## [37,]   73  42   20
## [38,]   73  40   55
## [39,]   80  35  720
## [40,]   78  35  365
## [41,]  100  46 1674
## [42,]   84  39  550
## [43,]   81  41  660
## [44,]   97  35 1195
## [45,]  122  45   77
## [46,]   76  40  365
## [47,]   75  39  100
## [48,]   79  32    9
## [49,]  103  44 3230
## [50,]   86  36  450
## [51,]  101  35 3685
## [52,]   94  29    5
## [53,]  111  40 4390
## [54,]   73  44  110
## [55,]   76  36   10
## [56,]  122  47   10
## [57,]  117  47 1890
## [58,]   89  43  860
## [59,]   87  43  635
## [60,]  104  41 6100
## [61,]   66  18   35
```

9. Conditioning plot

A conditioning plot, also known as a coplot or subset plot, is a plot of two variables conditional on the value of a third variable (called the conditioning variable). The conditioning variable may be either a variable that takes on only a few discrete values or a continuous variable that is divided into a limited number of subsets.

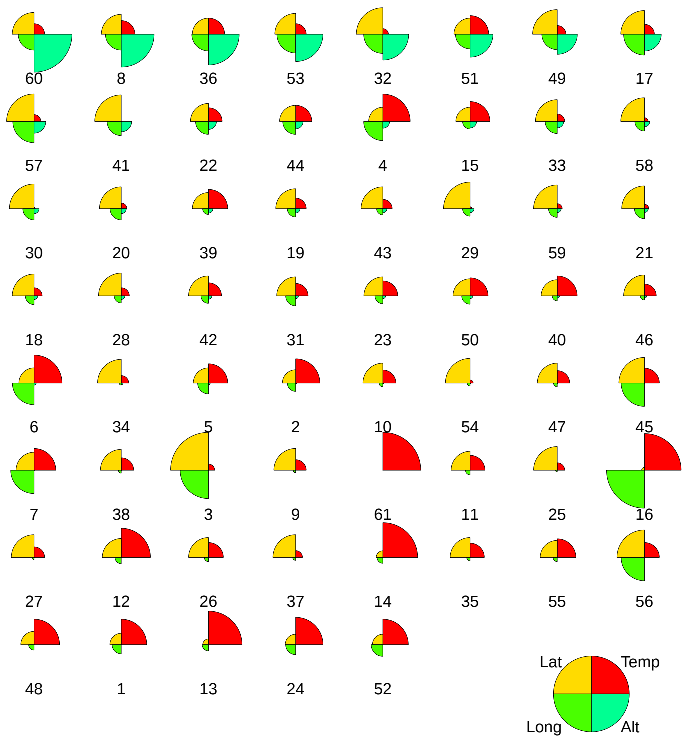
```r
library(graphics)
coplot(Long~Lat|Alt,rows=1,pch=16,cex=1.75,col="red",bar.bg=c(num="blue",fac=gray(0.95)),number=10)
```



10. Star Plot

The star plot summarizes each data item as a circle with different sized wedges. Every star is an individual (row) and every wedge on the star represents a different variable (column) in the data. The radius of each wedge in the star indicates the size of the scaled values for that individual, relative to all other measurements for that variable. These variables are listed in different colors and directions. The key for this is given in the lower right corner of the figure. The radius of each wedge is scaled and can be used to compare the relative magnitude of each variable values across all individuals.

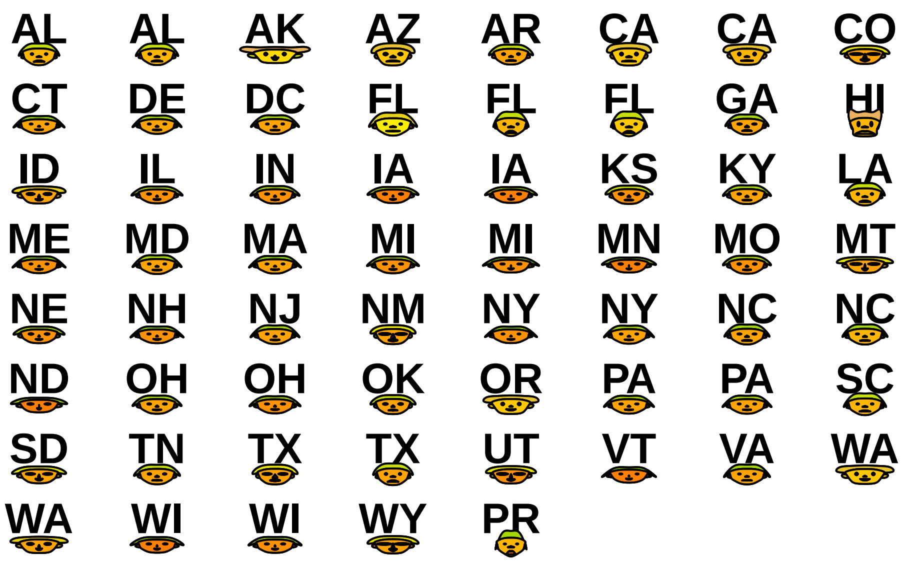
```
library(graphics) # for the stars program
palette(rainbow(7)) # set colors
alts=JanTemp[order(-JanTemp$Alt), ]# order by Alt, decreasing
stars(alts[,-c(1,2)], len=1, cex=0.5, key.loc=c(17, 1),
       labels=row.names(alts), draw.segments=TRUE)
```



11. Face plot

It is also called Chernoff face plot. In such a plot, each variable in the dataset is used to represent a feature of the face. Chernoff used 18 variables to represent different facial features such as head, nose, eyes, eyebrows, mouth, and ears.

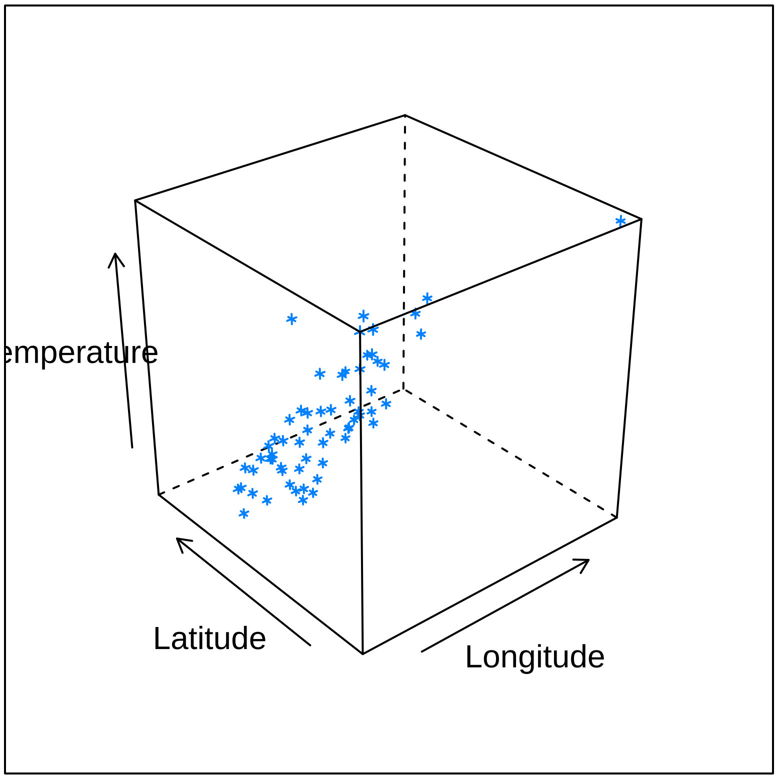
```
library(aplpack)
faces(JanTemp[,-c(1,2)], face.type=1, scale=TRUE,
      labels=JanTemp$State, plot.faces=TRUE, nrow.plot=8,
      ncol.plot =8)
```



```
## effect of variables:
##  modified item       Var
##  "height of face   " "Temp"
##  "width of face    " "Lat"
##  "structure of face" "Long"
##  "height of mouth  " "Alt"
##  "width of mouth   " "Temp"
##  "smiling          " "Lat"
##  "height of eyes   " "Long"
##  "width of eyes    " "Alt"
##  "height of hair   " "Temp"
##  "width of hair   "  "Lat"
##  "style of hair   "  "Long"
##  "height of nose  "  "Alt"
##  "width of nose   "  "Temp"
##  "width of ear    "  "Lat"
##  "height of ear   "  "Long"
```

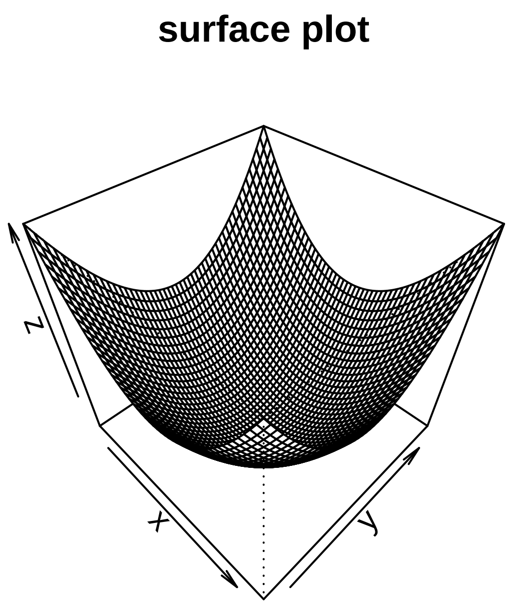
12. Three-dimensional scatter plot

```
cloud(Temp~Long+Lat,xlab="Longitude",ylab="Latitude",zlab="Temperature")
```

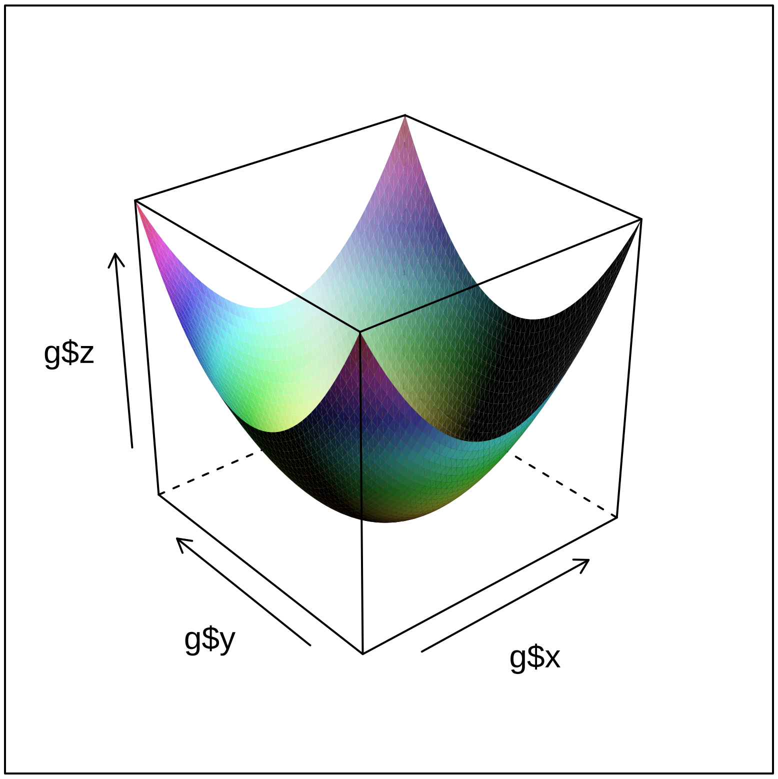


13. Three-dimensional surface plot

```
x=seq(-5,5,length=50);
y=seq(-5,5,length=50);
z=matrix(kronecker(x^2,y^2,"+"),nrow=50);
# Surface plot using base
persp(x,y,z,phi=45,theta=45,xlab="x",ylab="y",main="surface plot")
```

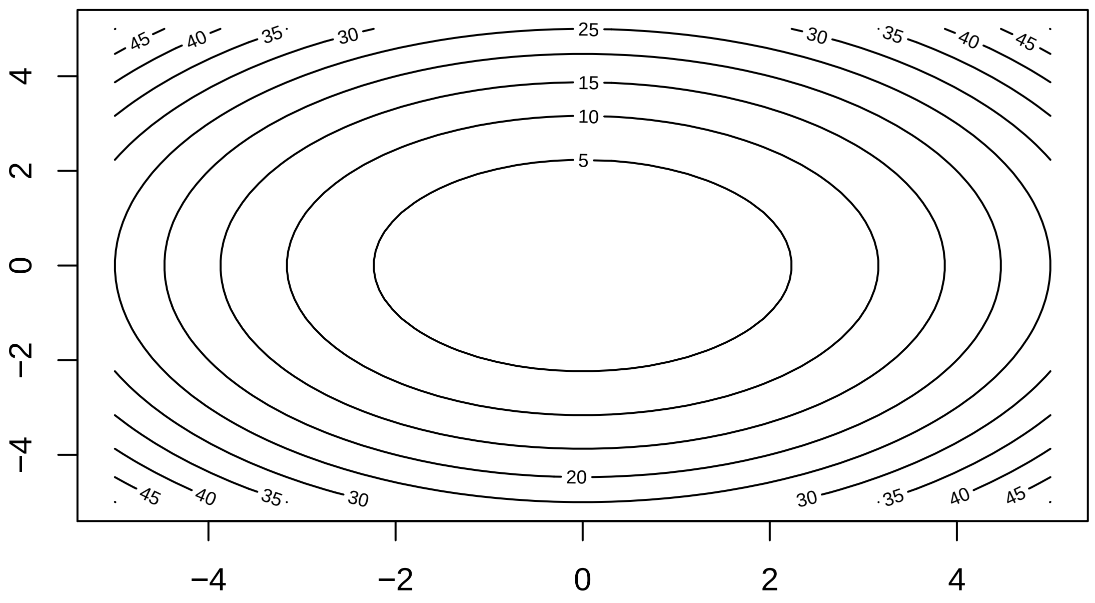


```
# Surface plot using wireframe from lattice package
library(lattice)
g=expand.grid(x=seq(-5,5,length=50),y=seq(-5,5,length=50),gr=1);
g$z=((g$x^2 + g$y^2));
wireframe(g$z~g$x*g$y, shade=TRUE, aspect=c(1,1), light.source=c(10,0,10));
```



14. contour plot

```
contour(x,y,z,nlevels=10)
```


5
10
15
20
25
30
35
40
45
−4
−2
0
2
4