

Boxplots

Source

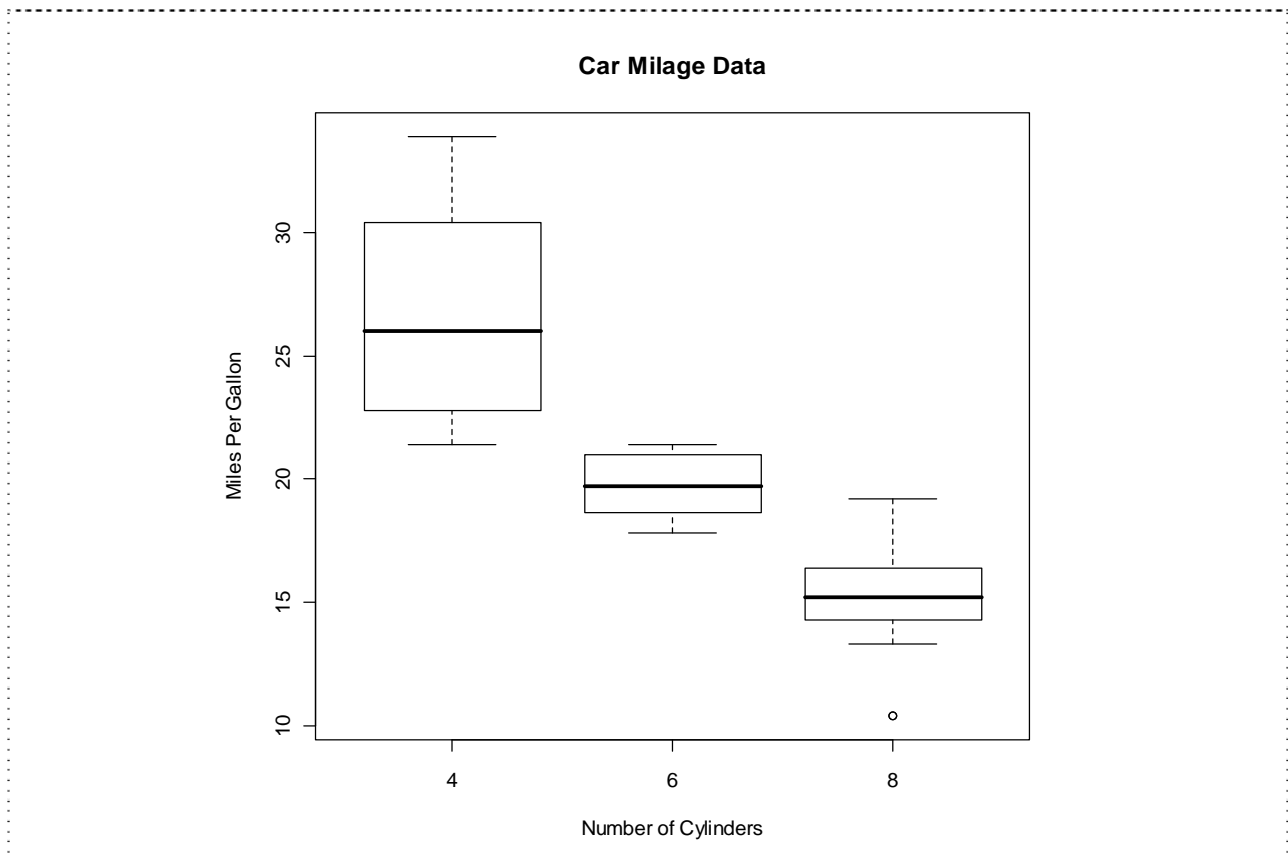
Reprinted with permission from Quick-R (<http://www.statmethods.net>).
© Copyright 2008 Robert I. Kabacoff, Ph.D. All rights reserved.

Background

Boxplots can be created for individual variables or for variables by group. The format is `boxplot(x, data=)`, where `x` is a formula and `data=` denotes the dataframe providing the data. An example of a formula is `y~group` where a separate boxplot for numeric variable `y` is generated for each value of `group`. Add `varwidth=TRUE` to make boxplot widths proportional to the square root of the samples sizes. Add `horizontal=TRUE` to reverse the axis orientation.

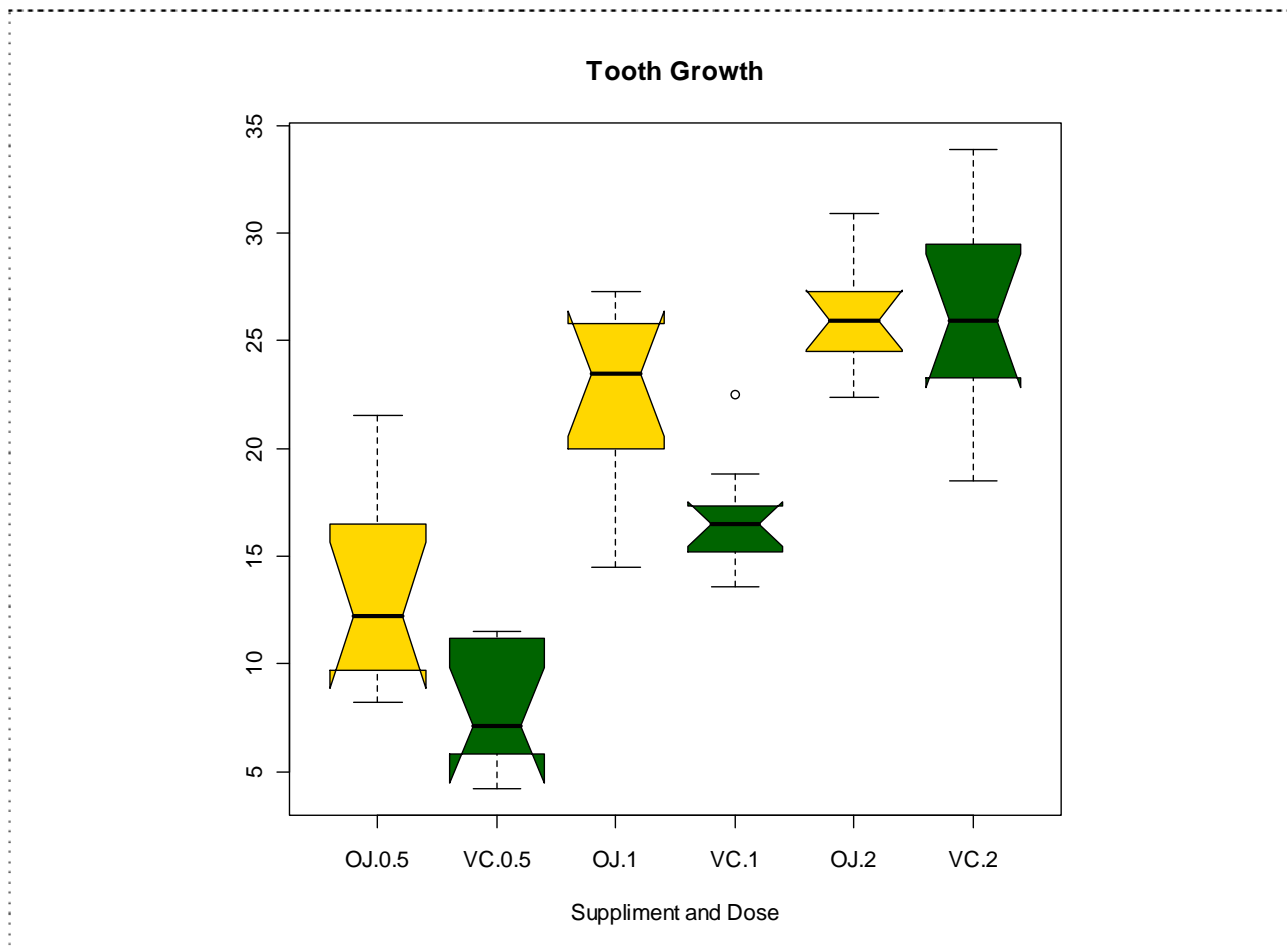
Boxplot of MPG by Car Cylinders

```
boxplot(
  mpg~cyl,
  data=mtcars,
  main="Car Milage Data",
  xlab="Number of Cylinders",
  ylab="Miles Per Gallon"
)
```



Notched Boxplot of Tooth Growth Against 2 Crossed Factors

```
# boxes colored for ease of interpretation
boxplot(
  len~supp*dose,
  data=ToothGrowth,
  notch=TRUE,
  col=(c("gold","darkgreen")),
  main="Tooth Growth",
  xlab="Suppliment and Dose"
)
```



In the notched boxplot, if two boxes' notches do not overlap this is 'strong evidence' their medians differ (Chambers et al., 1983, p. 62).

Colors recycle. In the example above, if I had listed 6 colors, each box would have its own color. Earl F. Glynn has created an easy to use list of colors in PDF format.

Other Options

The `boxplot.matrix()` function in the `sfsmisc` package draws a boxplot for each column (row) in a matrix. The `boxplot.n()` function in the `gplots` package annotates each boxplot

with its sample size. The `bplot()` function in the `Rlab` package offers many more options controlling the positioning and labeling of boxes in the output.

Violin Plots

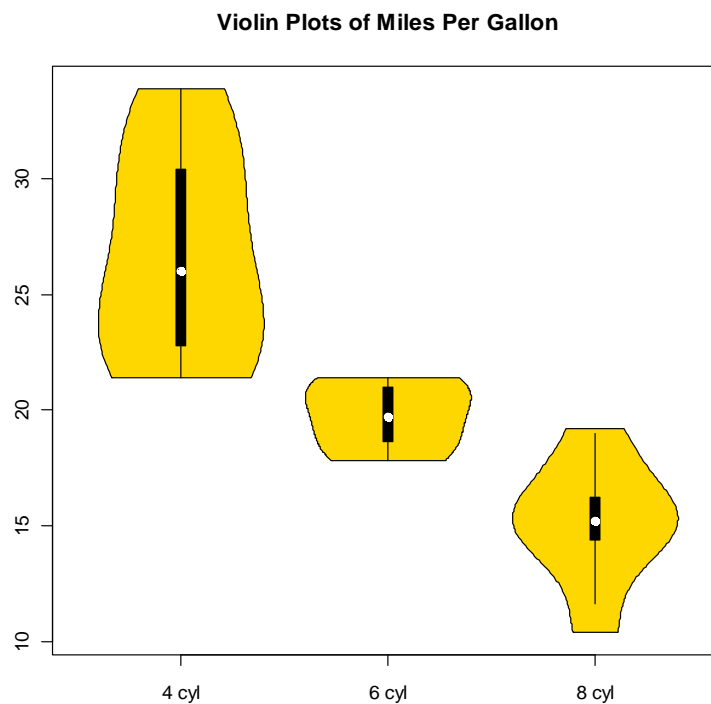
A violin plot is a combination of a boxplot and a kernel density plot. They can be created using the `vioplot()` function from `vioplot` package.

Example Violin Plot

```
x1 <- mtcars$mpg[mtcars$cyl==4]
x2 <- mtcars$mpg[mtcars$cyl==6]
x3 <- mtcars$mpg[mtcars$cyl==8]

show <- function()
{
  vioplot(
    x1,
    x2,
    x3,
    names=c("4 cyl", "6 cyl", "8 cyl"),
    col="gold"
  )
  title("Violin Plots of Miles Per Gallon")
}

show()
```



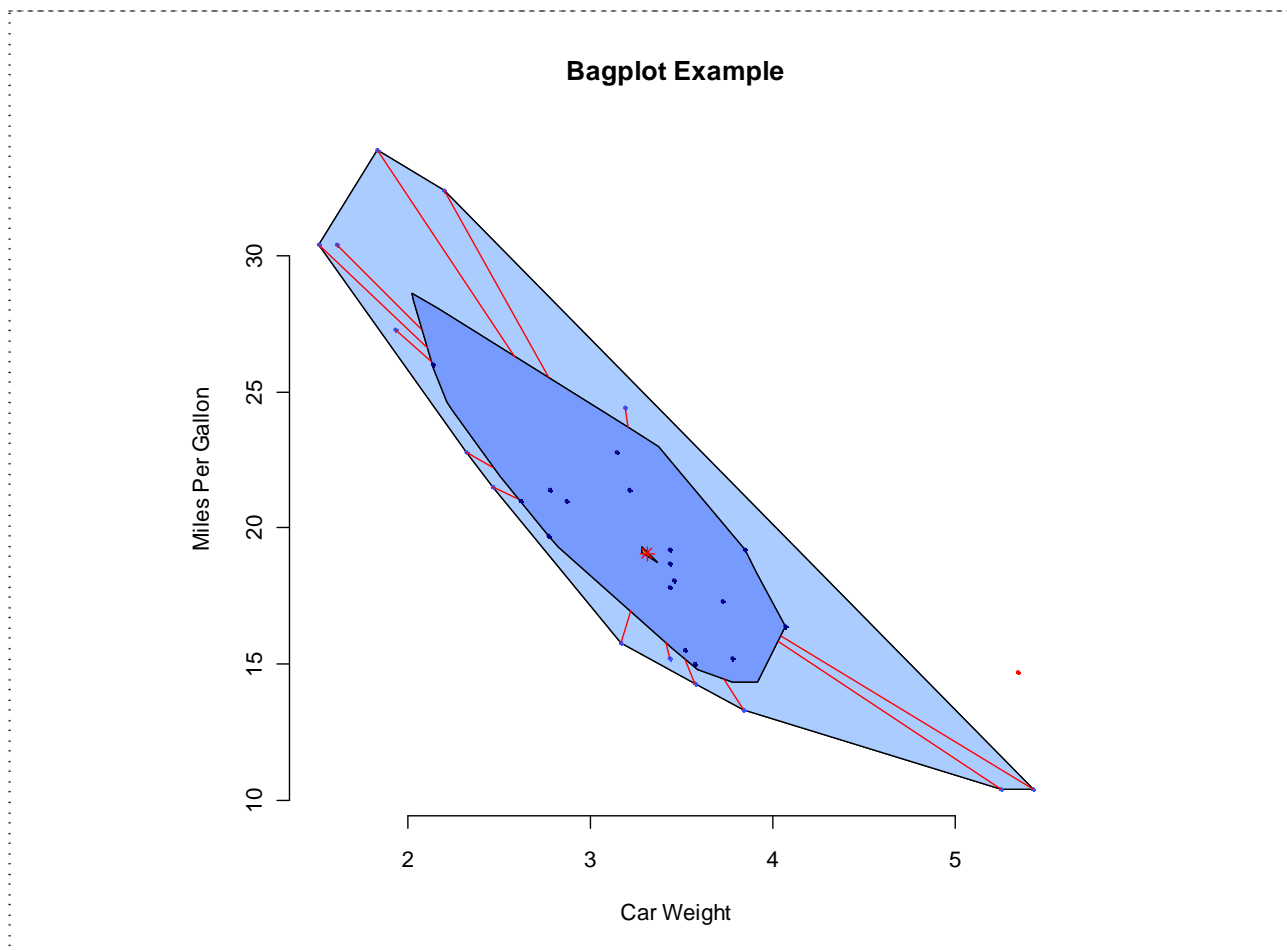
Bagplot - A 2D Boxplot Extension

The `bagplot(x, y)` function in the `aplpack` package provides a bivariate version of the univariate boxplot. The bag contains 50% of all points. The bivariate median is approximated. The fence separates points in the fence from points outside. Outliers are displayed.

Example of a Bagplot

```
attach(mtcars)
```

```
bagplot(
  wt,
  mpg,
  xlab="Car Weight",
  ylab="Miles Per Gallon",
  main="Bagplot Example"
)
```



Scatter Plots

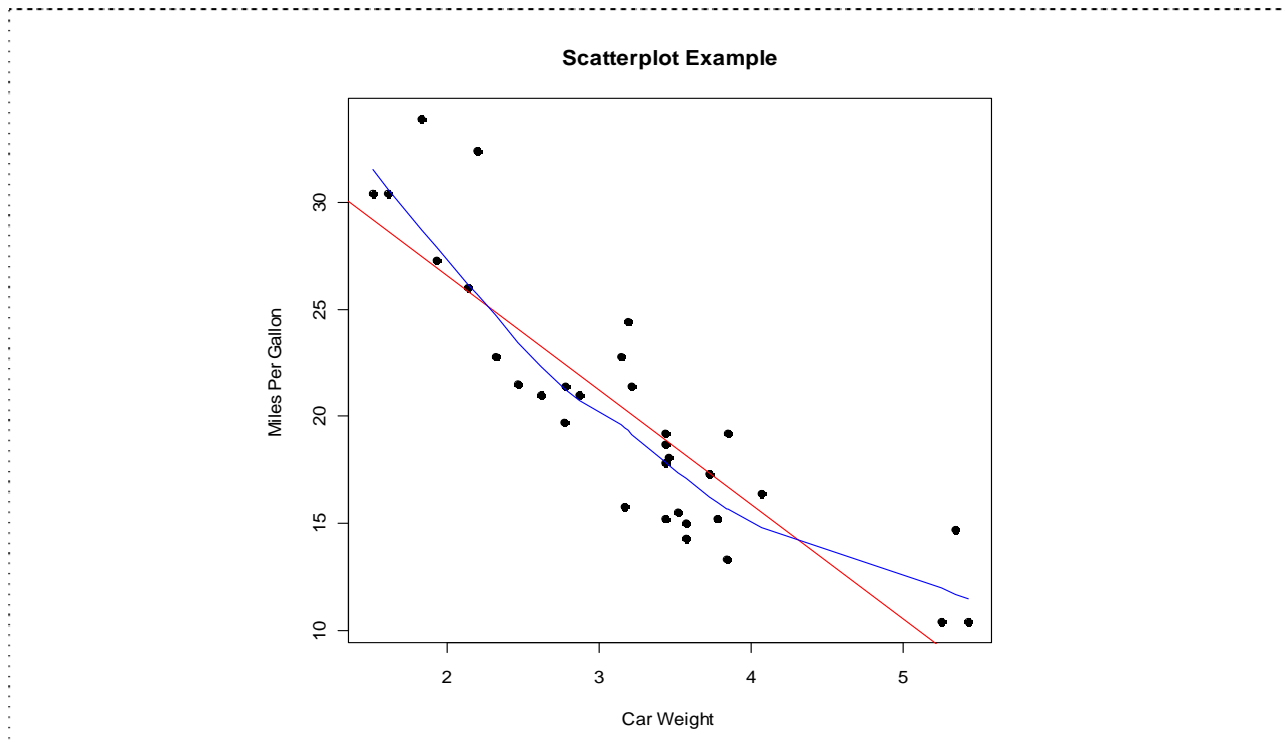
There are many ways to create a scatterplot in R. The basic function is `plot(x, y)`, where `x` and `y` are numeric vectors denoting the (x,y) points to plot. Use `abline()` to add a straight line and `lines()` to add a smoothed line to the plot.

Simple Scatterplot

```
attach(mtcars)

show <- function()
{
  plot(
    wt,
    mpg,
    main="Scatterplot Example",
    xlab="Car Weight ",
    ylab="Miles Per Gallon ",
    pch=19
  )
  # Add fit lines
  # regression line (y~x)
  abline(lm(mpg~wt), col="red")
  # lowess line (x,y)
  lines(lowess(wt,mpg), col="blue")
}

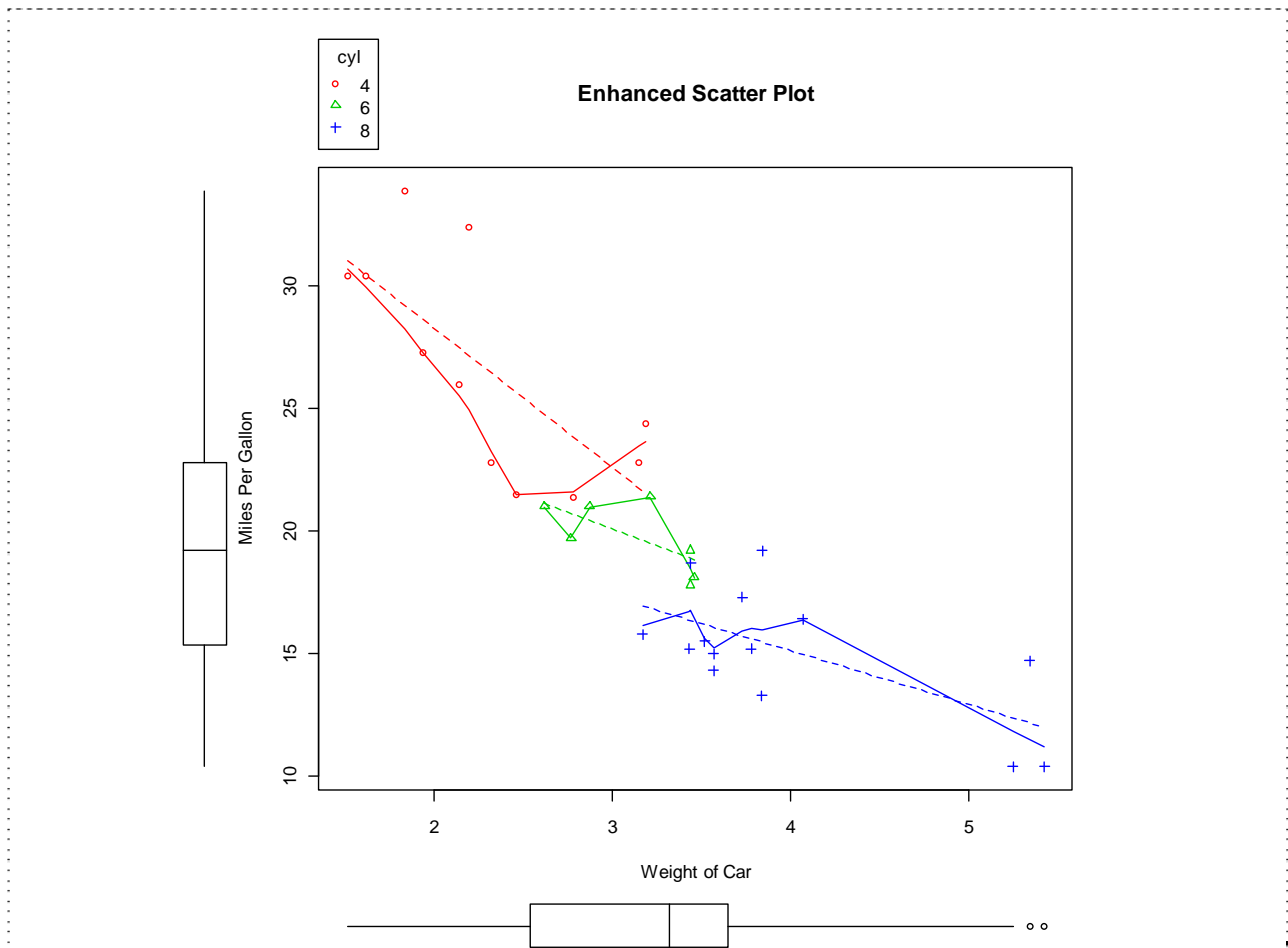
show()
```



The `scatterplot()` function in the `car` package offers many enhanced features, including fit lines, marginal box plots, conditioning on a factor, and interactive point identification. Each of these features is optional.

Enhanced Scatterplot of MPG vs. Weight by Number of Car Cylinders

```
scatterplot(
  mpg ~ wt | cyl,
  data=mtcars,
  xlab="Weight of Car",
  ylab="Miles Per Gallon",
  main="Enhanced Scatter Plot",
  labels=row.names(mtcars)
)
```

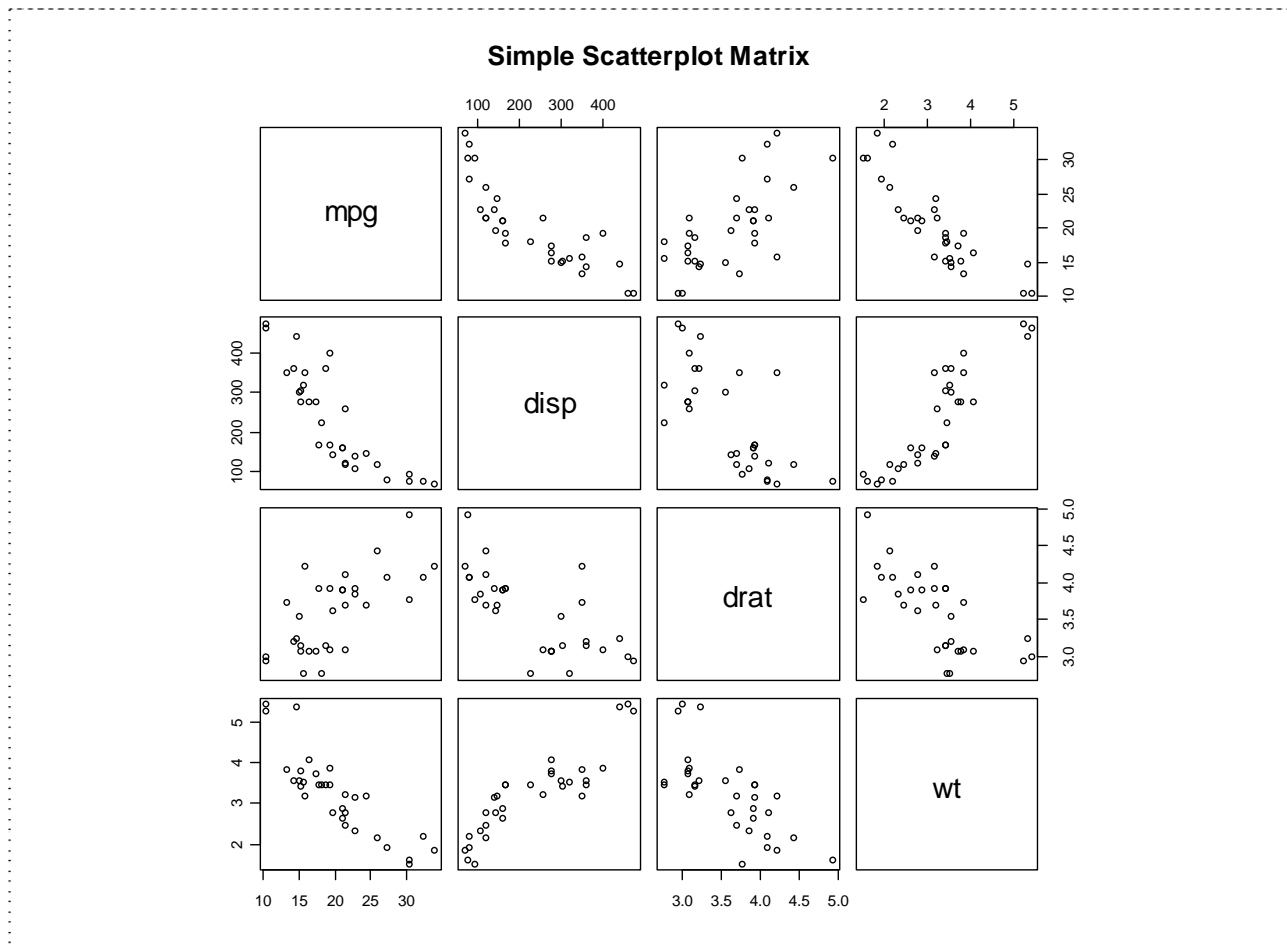


Scatterplot Matrices

There are at least 4 useful functions for creating scatterplot matrices. Analysts must love scatterplot matrices!

Basic Scatterplot Matrix

```
pairs(
  ~mpg+disp+drat+wt,
  data=mtcars,
  main="Simple Scatterplot Matrix"
)
```

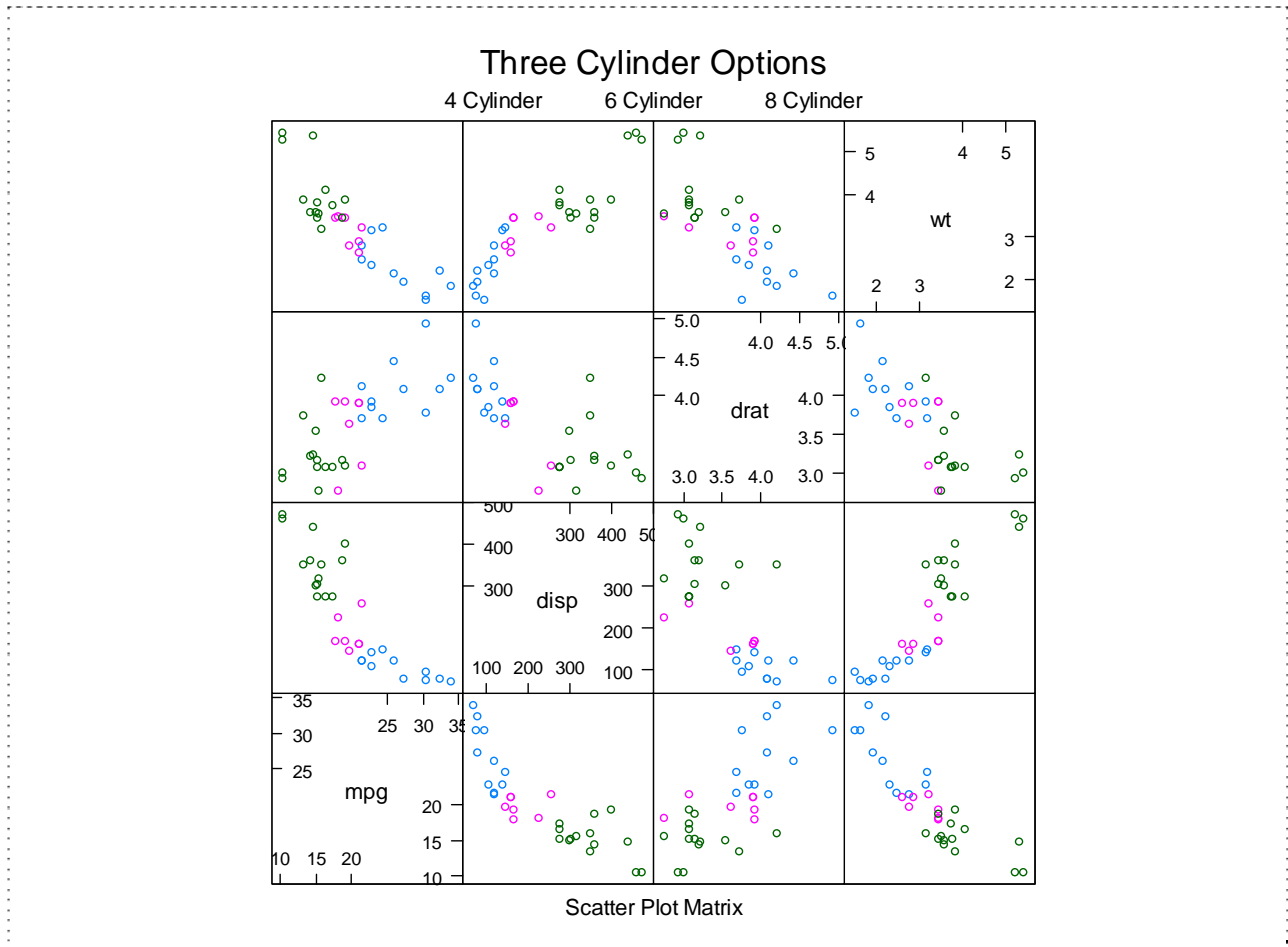


Scatterplot Matrices from the lattice Package

The lattice package provides options to condition the scatterplot matrix on a factor.

```
splom(
  mtcars[c(1,3,5,6)],
  groups=cyl,
  data=mtcars,
  panel=panel.superpose,
```

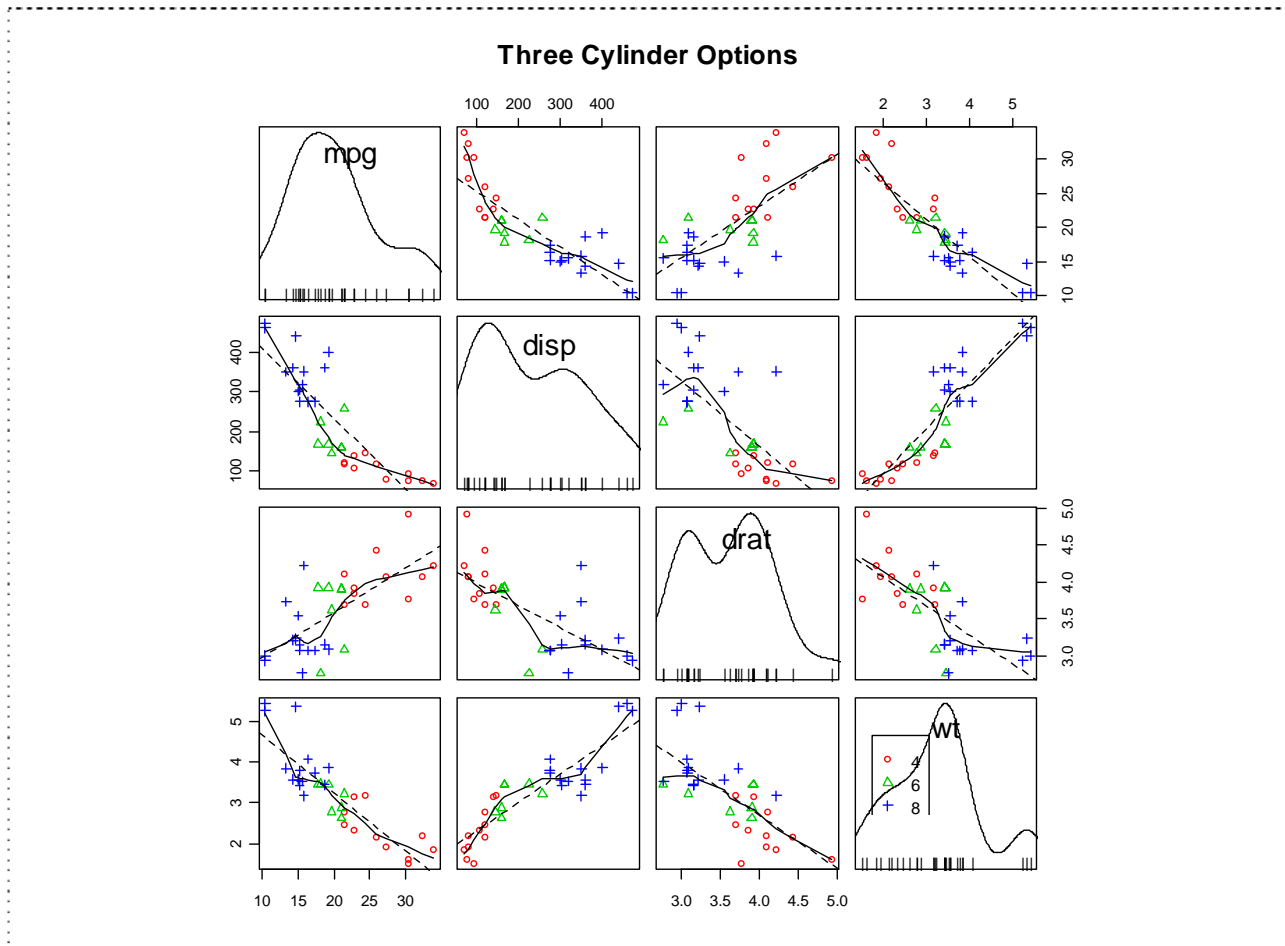
```
key=list(
  title="Three Cylinder Options",
  columns=3,
  #points=list(
    #pch=super.sym$pch[1:3],
    #col=super.sym$col[1:3]
  #),
  text=list(c("4 Cylinder","6 Cylinder","8 Cylinder"))
)
```



Scatterplot Matrices from the car Package

The car package can condition the scatterplot matrix on a factor, and optionally include lowess and linear best fit lines, and boxplot, densities, or histograms in the principal diagonal, as well as rug plots in the margins of the cells.

```
scatterplot.matrix(
  ~mpg+disp+drat+wt|cyl,
  data=mtcars,
  main="Three Cylinder Options"
)
```



Scatterplot Matrices from the `glus` Package

The `glus` package provides options to rearrange the variables so that those with higher correlations are closer to the principal diagonal. It can also color code the cells to reflect the size of the correlations.

```
# get data
dta <- mtcars[c(1,3,5,6)]

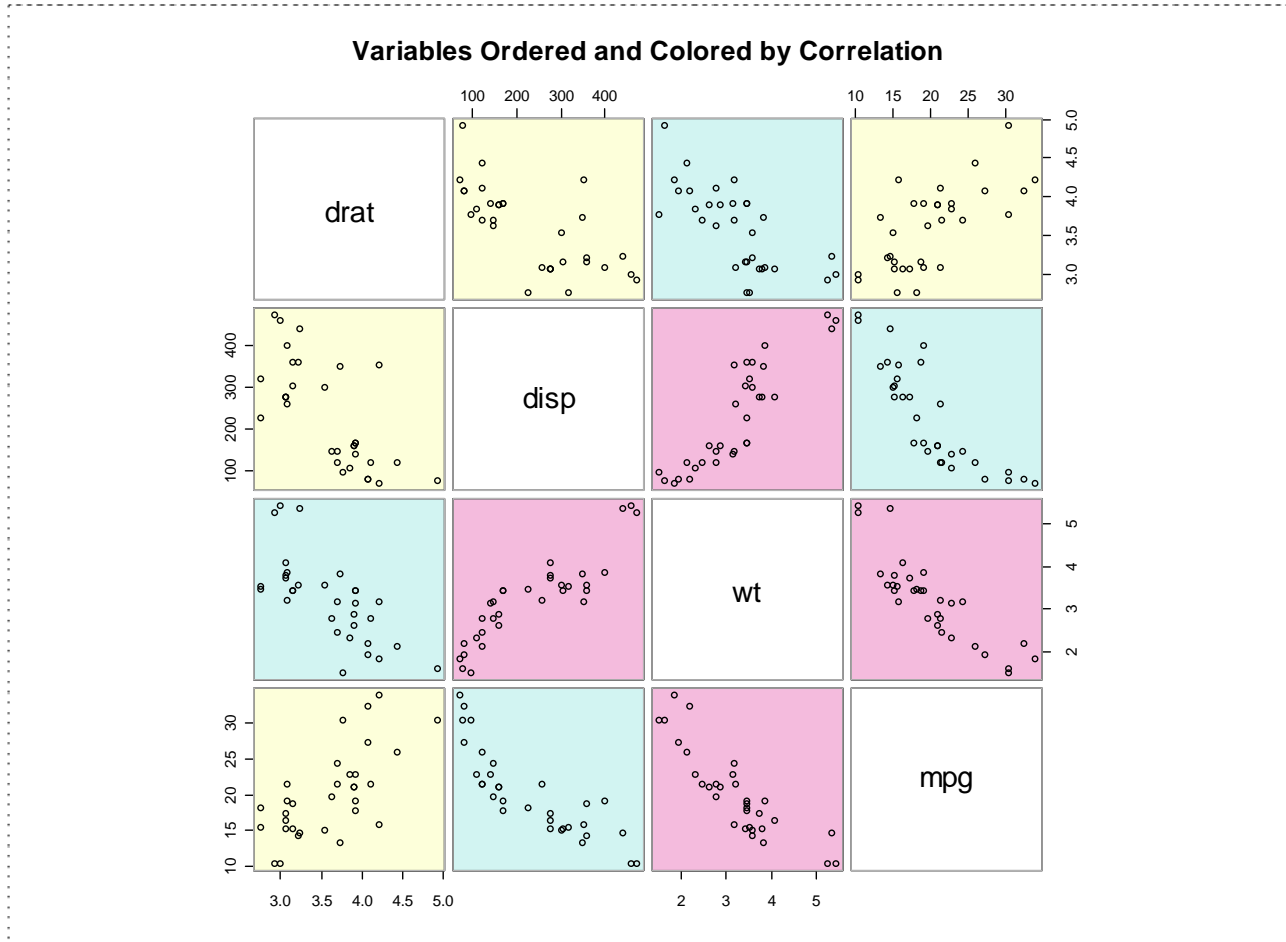
# get correlations
dta.r <- abs(cor(dta))

# get colors
dta.col <- dmat.color(dta.r)

# reorder variables so those with highest correlation are closest to the diagonal
dta.o <- order.single(dta.r)

cpairs(
  dta,
  dta.o,
```

```
panel.colors=dta.col,  
gap=.5,  
main="Variables Ordered and Colored by Correlation"  
)
```



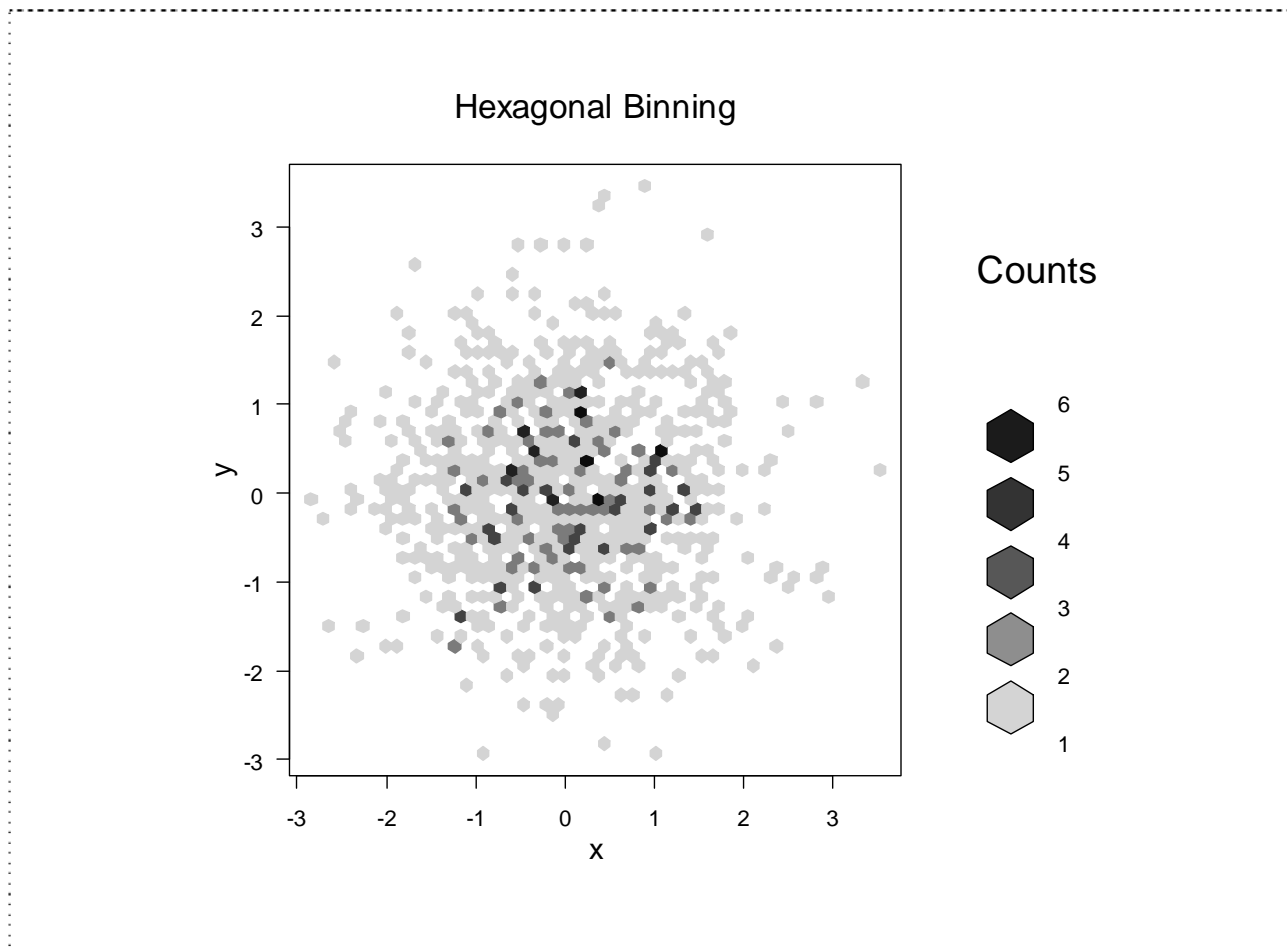
click to view

High Density Scatterplots

When there are many data points and significant overlap, scatterplots become less useful. There are several approaches that be used when this occurs. The `hexbin(x, y)` function in the `hexbin` package provides bivariate binning into hexagonal cells (it looks better than it sounds).

High Density Scatterplot with Binning

```
x <- rnorm(1000)  
y <- rnorm(1000)  
  
bin <- hexbin(x, y, xbins=50)  
plot(bin, main="Hexagonal Binning")
```



Another option for a scatterplot with significant point overlap is the sunflowerplot. See `help(sunflowerplot)` for details.

High Density Scatterplot with Color Transparency

Finally, you can save the scatterplot in PDF format and use color transparency to allow points that overlap to show through (this idea comes from B.S. Everitt in HSAUR).

```
pdf("c:/scatterplot.pdf")
x <- rnorm(1000)
y <- rnorm(1000)
plot(
  x,
  y,
  main="PDF Scatterplot Example",
  col=rgb(0,100,0,50,maxColorValue=255),
  pch=16
)
dev.off()
```

Note: You can use the `col2rgb()` function to get the rgb values for R colors. For example, `col2rgb("darkgreen")` yeilds `r=0, g=100, b=0`. Then add the alpha transparency level as the

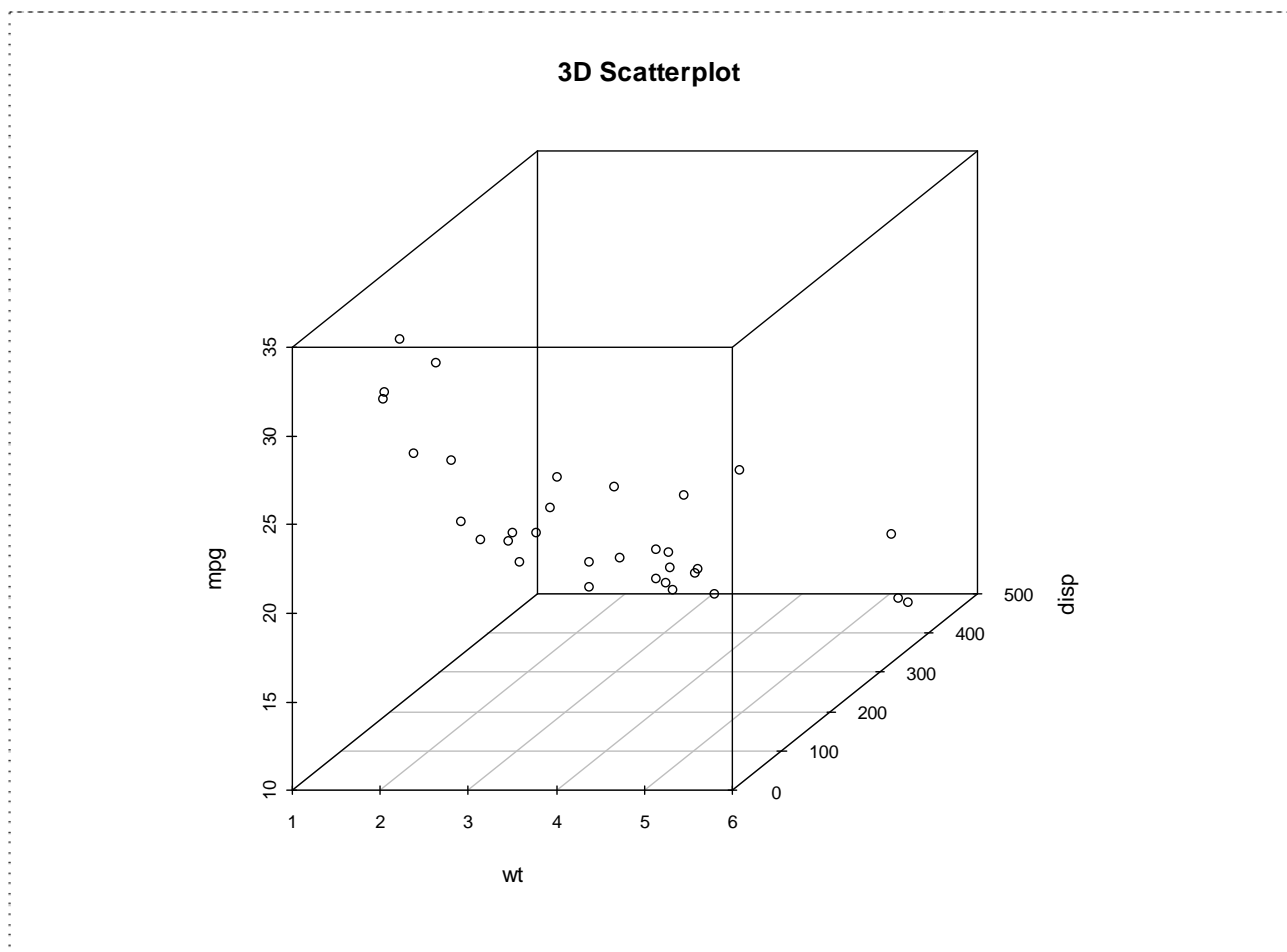
4th number in the color vector. A value of zero means fully transparent. See `help(rgb)` for more information.

3D Scatterplots

You can create a 3D scatterplot with the `scatterplot3d` package. Use the function `scatterplot3d(x, y, z)`.

3D Scatterplot

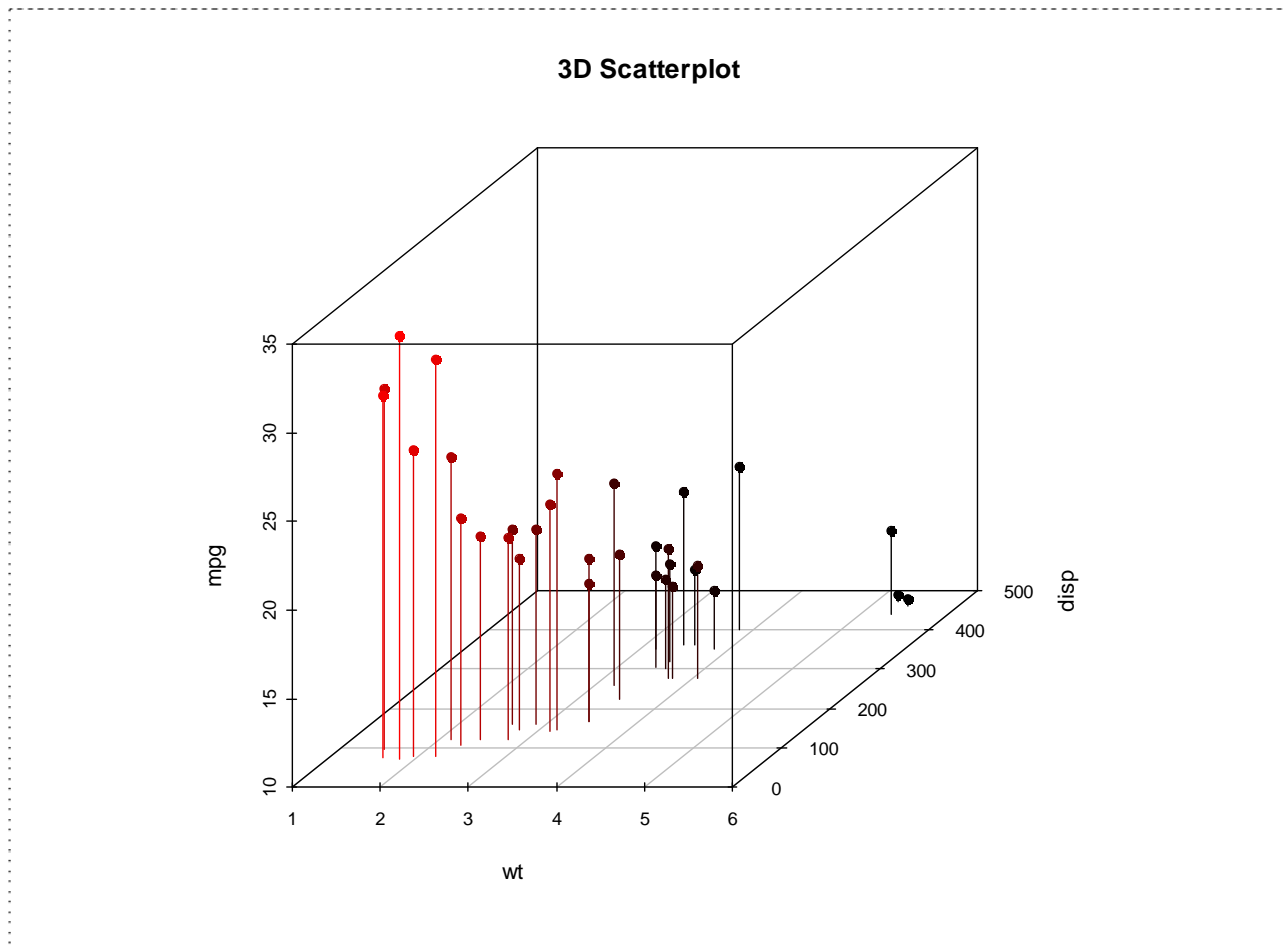
```
attach(mtcars)  
scatterplot3d(wt, disp, mpg, main="3D Scatterplot")
```



3D Scatterplot with Coloring and Vertical Drop Lines

```
attach(mtcars)  
  
scatterplot3d(  
  wt,  
  disp,  
  mpg,
```

```
pch=16,
highlight.3d=TRUE,
type="h",
main="3D Scatterplot"
)
```



3D Scatterplot with Coloring and Vertical Lines and Regression Plane

```
attach(mtcars)

show <- function()
{
  s3d <- scatterplot3d(
    wt,
    disp,
    mpg,
    pch=16,
    highlight.3d=TRUE,
    type="h",
    main="3D Scatterplot"
  )
  fit <- lm(mpg ~ wt+disp)
  s3d$plane3d(fit)
}
```

```
}  
show()
```

